Filomat 35:11 (2021), 3781–3799
https://doi.org/10.2298/FIL2111781T

Published by Faculty of Sciences and Mathematics,
University of Niš, Serbia
Available at: http://www.pmf.ni.ac.rs/filomat

An SOR-Type Algorithm Based on IO Iteration for Solving Coupled Discrete Markovian Jump Lyapunov Equations

Zhaolu Tian[a], Tongyang Xu[b]

[a]*College of Applied Mathematics, Shanxi University of Finance and Economics, Taiyuan 030006, P.R.China*
[b]*School of Information, Shanxi University of Finance and Economics, Taiyuan 030006, P.R.China*

Abstract. In this paper, based on the inner-outer (IO) iteration framework [17], by introducing some tunable parameters, an SOR-type IO (SIO) iteration method is proposed for solving the Sylvester matrix equation and coupled Lyapunov matrix equations (CLMEs) in the discrete-time jump linear systems with Markovian transitions. Fisrtly, the SIO iteration algorithm for solving the discrete Sylvester matrix equation is developed, its convergence property is analyzed and the choices of the parameters are also discussed. Next, the SIO iteration algorithm is used to solve the CLMEs. Moreover, by using the latest estimations, a current-estimation-based SIO (CSIO) iteration algorithms are also constructed for solving the CLMEs, respectively. The boundedness and monotonicity of the iteration sequence derived from the proposed algorithm with zero initial conditions are established. Finally, several numerical examples are implemented to illustrate the superiorities of the proposed iteration algorithms.

1. Introduction

The coupled algebraic Lyapunov matrix equations play an important role in the stability analysis for discrete-time Markovian jump linear systems [2,4,6,10], and the mean square stability of a discrete-time Markovian jump system can be equivalent to the existence of positive definite solutions of such matrix equations. In [13], based on the coupled algebraic Lyapunov matrix equations, a necessary and sufficient condition of the stochastic stability was given for Markovian jump systems.

In the last decades, some numerical methods have been proposed to solve the coupled algebraic Lyapunov equations for discrete-time Markovian jump linear systems due to its broad applications. In [2], a formula for computing the exact solutions was presented by using the matrix inversions and Kronecker products, which cost is very high if the system dimension and the number of modes are large. In [8], a parallel algorithm was constructed for solving coupled discrete Markovian jump Lyapunov equations under two strong assumptions, which are the zero initial condition and the stability of all subsystems, respectively. In [12], it is showed that the iteration algorithm [8] was also convergent without the assumption of zero initial conditions. A gradient-based iteration algorithm was presented to solve the coupled Lyapunov matrix equations in [14]. A finite iteration algorithm [7] was also developed to solve the coupled Lyapunov

2020 *Mathematics Subject Classification*. Primary 15A24; Secondary 65F30, 65F35
Keywords. Coupled Lyapunov matrix equations, Sylvester matrix equation, Inner-outer iteration, Parameter, Convergence
Received: 20 August 2020; Revised: 23 June 2021; Accepted: 28 June 2021
Communicated by Dijana Mosić
Research supported by the Natural Science Foundation of Shanxi Province (20210302123480) and Research Project Supported by Shanxi Scholarship Council of China (2020-098)
Email address: tianzhaolu2004@126.com (Zhaolu Tian)

equations for Markovian jump systems. In [9], a new implicit iteration method was established by using the updated variables in the current step for estimation of other variables. However, one needs to solve many normal discrete Lyapunov matrix equations in each iteration step in this algorithm, which is not a simple task for large discrete-time Markovian jump linear systems. There are some iteration algorithms, which are established to solve other matrix equations and tensor matrix equations, can also been used to solve the coupled Lyapunov matrix equations, such as [1,5,15,23,24,26-32].

In [17], an inner-outer (IO) iteration algorithm was proposed for solving the Sylvester matrix equation and coupled continuous Markovian jump Lyapunov matrix equations, respectively. In this paper, by using the IO iteration and introducing some tunable parameters, an SIO iteration algorithm is presented to solve the Sylvester matrix equation and CLMEs, respectively. To improve its performance, a current-estimation-based SIO iteration algorithm is constructed in the sequel. Moreover, the choices of the parameters in these algorithms are also discussed, and some heuristical strategies are given for choosing appropriate parameters. Several numerical examples are used to illustrate the effectiveness of the proposed algorithms.

Throughout this paper, for a matrix $A \in R^{n\times n}$, A^T and $\rho(A)$ denote its transpose and spectral radius, respectively. For two integers m and n with $m \le n$, $\Pi[m, n]$ denotes the set $\{m, m + 1, \cdots, n\}$. For a matrix $A = [a_1\ a_2\ \cdots\ a_n]$, $\mathrm{vec}(A) = [a_1^T\ a_2^T\ \cdots\ a_n^T]^T$. The notation $A \otimes B$ represents the Kronecker products of the matrices A and B. The matrix $E \ge 0$ means that E is real symmetric and positive semidefinite. The matrix tuple $\mathcal{F} = \{F_1, F_2, \cdots F_n\} \ge 0$ implies all the matrices $F_i \ge 0,\ i \in \Pi[1, n]$. In what follows, it should be stated that the sum is zero if the upper limit of the sum notation is less than the lower limit.

2. Previous results

Consider the following discrete-time Markovian jump linear system:

$$x(k + 1) = A_{\theta(k)}x_k,\ x(0) = x_0,\ \ \theta(0) = \theta_0, \tag{2.1}$$

where $x(k) \in R^n$ is the state vector, and the system parameter $A_{\theta(k)}$ is changing in accordance with a discrete-time Markovian random process $\theta(k)$, which takes values in a discrete finite set $\Omega = \{1, 2, \cdots, N\}$. The dynamics of the probability distribution of the Markov chain is described by the differential equation

$$\dot{\pi}(t) = \pi(t)P, \tag{2.2}$$

where π is an N-dimensional row vector of unconditional probabilities, P is the transition rate matrix denoted by $[p_{ij}]_{n\times n}$ and satisfies the following relation:

$$p_{ij} = \Pr\{\theta(k + 1) = j|\theta(k) = i\} \tag{2.3}$$

with the properties that $p_{ij} \ge 0(i, j = 1, 2, \cdots, N)$ and $\sum\limits_{j=1}^{n} p_{ij} = 1$. The coupled Lyapunov matrix equations associated with the system (2.1)-(2.3) are given by

$$K_i = A_i^T\left(\sum_{j=1}^{N} p_{ij}K_j\right)A_i + Q_i,\ i \in \Omega, \tag{2.4}$$

where $Q_i \ge 0$ $(i \in \Omega)$ and the subscript i indicates that the system is in mode $\theta_j = i$ which implies $A_i = A(\theta_j = i)$.

Now, some main explicit numerical algorithms proposed for solving Eqs.(2.4) are listed as follows.

Lemma 2.1 [8]. Assume that Eqs. (2.4) have unique positive semidefinite or positive definite solutions with given $Q_i \ge 0$ $(i \in \Omega)$, and the matrices A_i $(i \in \Omega)$ are Schur stable, then the solutions of Eqs. (2.4) can be generated by the following iteration algorithm:

$$K_i(m + 1) = A_i^T\left(\sum_{j=1}^{N} p_{ij}K_j(m)\right)A_i + Q_i,\ K_i(0) = 0,\ i \in \Omega, \tag{2.5}$$

which satisfies

$$\lim_{m\to\infty} K_i(m) = K_i,\ i \in \Omega.$$

Lemma 2.2 [12]. Assume that Eqs. (2.4) has solutions $K_i > 0$ ($i \in \Omega$) for any given $Q_i > 0$ ($i \in \Omega$), then the solutions of Eqs. (2.4) can be obtained by the following iteration algorithm:

$$K_i(m+1) = A_i^T \left(\sum_{j=1}^{N} p_{ij} K_j(m) \right) A_i + Q_i,\ i \in \Omega \tag{2.6}$$

for any initial conditions $K_i(0)$ ($i \in \Omega$).

Lemma 2.3 [9]. If Eqs. (2.4) have unique positive semidefinite or positive definite solutions with $Q_i \geq 0$ ($i \in \Omega$), and the matrices A_i ($i \in \Omega$) are Schur stable, then the solutions of Eqs. (2.4) can be derived from the following iteration algorithm:

$$K_i(m+1) = A_i^T \left(\sum_{j=1}^{i-1} p_{ij} K_j(m+1) + \sum_{j=i}^{N} p_{ij} K_j(m) \right) A_i + Q_i,\ K_i(0) = 0,\ i \in \Omega. \tag{2.7}$$

Next, several implicit iteration algorithms are presented. In these algorithms, one should solve N normal discrete Lyapunov matrix equations in each iteration step.

Lemma 2.4 [12]. Assume that Eqs. (2.4) have unique solutions and $\rho(\Phi) < 1$, where Φ is the matrix defined as in Theorem 1 [12], then the iteration sequences $K_i(m)$ ($i \in \Omega$) generated by the following iteration algorithm:

$$p_{ii} A_i^T K_i(m+1) A_i - K_i(m+1) = -A_i^T \left(\sum_{j=1, j\neq i}^{N} p_{ij} K_j(m) \right) A_i - Q_i,\ i \in \Omega, \tag{2.8}$$

converge to the solutions of Eqs. (2.4) for any initial conditions $K_i(0)$ ($i \in \Omega$).

Lemma 2.5 [9]. If Eqs. (2.4) have unique positive semidefinite or positive definite solutions with $Q_i \geq 0$ ($i \in \Omega$), and the matrices A_i ($i \in \Omega$) are Schur stable, then the solutions of Eqs. (2.4) can be obtained from the following iteration algorithm:

$$\begin{aligned} &p_{ii} A_i^T K_i(m+1) A_i - K_i(m+1) \\ &= -A_i^T \left(\sum_{j=1}^{i-1} p_{ij} K_j(m+1) + \sum_{j=i+1}^{N} p_{ij} K_j(m) \right) A_i - Q_i,\ K_i(0) = 0,\ i \in \Omega \end{aligned} \tag{2.9}$$

converge to the unique solutions of Eqs. (2.4).

3. The SIO iteration algorithm for solving the discrete Sylvester matrix equation

In this section, we firstly review the IO iteration algorithm [17] for solving the discrete Sylvester matrix equation. Next, we propose the SIO iteration algorithm by introducing a tunable parameter, analyze its convergence properties, and discuss the choices of the parameters in this algorithm.

3.1. *The IO iteration algorithm*

Consider the following discrete Sylvester matrix equation [18]

$$X - AXB = C, \tag{3.1}$$

where $A \in R^{m\times m}$, $B \in R^{n\times n}$, $C \in R^{m\times n}$ are known matrices, and $X \in R^{m\times n}$ is the unknown matrix to be determined. If $B = A^T$, Eq. (3.1) is the well-known discrete Lyapunov matrix equation.

The IO iteration algorithm for solving Eq. (3.1) is described as follows:

$$X_{k+1} - \beta A X_{k+1} B = (1-\beta) A X_k B + C,\ k = 0, 1, 2, \cdots \tag{3.2}$$

with $0 < \beta < 1$. (3.2) is the so-called outer iteration of the IO iteration algorithm.

Let $W_k = (1-\beta)AX_kB + C$ and $Y = X_{k+1}$. Then (3.2) can be written equivalently in the following form:

$$Y - \beta AYB = W_k,$$

and X_{k+1} can be obtained by the following inner iteration:

$$Y_{j+1} = \beta AY_jB + W_k,\ j = 0, 1, 2, \cdots, s_k - 1 \tag{3.3}$$

with $Y_0 = X_k$ as the initial guess and Y_{s_k} as the approximate solution to X_{k+1} in (3.2).

Theorem 3.1 [17]. Let $0 < \beta < 1$, and s_k be the number of the inner iteration steps at the k−th outer iteration. If $\rho(A)\rho(B) < 1$, then the iteration sequence $\{X_k\}_{k=0}^{\infty}$ generated by Algorithm 1 converges to the exact solution X^* to Eq. (3.1). Furthermore, the IO iteration algorithm converges faster than the Smith method for any initial vector.

3.2. *The SIO iteration algorithm*

By introducing a tunable parameter ω, then we have

$$X = \omega X + (1-\omega)X \tag{3.4}$$

From (3.1) and (3.2), it follows that

$$X - \beta AXB = (1-\beta)AXB + C \tag{3.5}$$

By using the Kronecker products [11], (3.4) and (3.5) can be equivalently rewritten as

$$\begin{cases} x = \omega x + (1-\omega)x, \\ x = (I - \beta B^T \otimes A)^{-1}((1-\beta)B^T \otimes Ax + c) \end{cases} \tag{3.6}$$

with $x = \text{vec}(X)$ and $c = \text{vec}(C)$. Then, from (3.6) it is clear that

$$x = \omega(I - \beta B^T \otimes A)^{-1}((1-\beta)B^T \otimes Ax + c) + (1-\omega)x,$$

which leads to

$$(I - \beta B^T \otimes A)x = \omega((1-\beta)B^T \otimes Ax + c) + (1-\omega)(I - \beta B^T \otimes A)x,$$

or equivalently,

$$\begin{aligned} X - \beta AXB &= \omega((1-\beta)AXB + C) + (1-\omega)(X - \beta AXB) \\ &= \omega(1-\beta)AXB - (1-\omega)\beta AXB + (1-\omega)X + \omega C \\ &= (\omega-\beta)AXB + (1-\omega)X + \omega C. \end{aligned} \tag{3.7}$$

Based on (3.7), an outer iteration sequence can be given as follows:

$$X_{k+1} - \beta AX_{k+1}B = (\omega-\beta)AX_kB + (1-\omega)X_k + \omega C. \tag{3.8}$$

Let $E_k = (\omega-\beta)AX_kB + (1-\omega)X_k + \omega C$, then an inner iteration can be defined by

$$Z_{j+1} = \beta AZ_jB + E_k,\ j = 0, 1, 2, \cdots, l_k - 1, \tag{3.9}$$

where $Z_0 = X_k$ is the initial condition, and Z_{l_k} is treated as the approximate solution to X_{k+1} in (3.8). If $\omega = 1$, then (3.8) and (3.9) reduce to the IO iteration algorithm.

Algorithm 1: The SIO iteration algorithm for solving Eq. (3.1)

Input: $A, B, C, \omega, \beta, \varepsilon$

Output: X

1: $X \leftarrow C$

2: $Z \leftarrow AXB$

3: **while** $\|C + Z - X\| \geq \varepsilon$

4: $E \leftarrow (\omega-\beta)Z + (1-\omega)X + \omega C$
5: **for** i=1:l_k
6: $X \leftarrow \beta Z + E$
7: $Z \leftarrow AXB$
8: **end**
9: **end while**

Lemma 3.1 [11]. For all operator norms $\rho(W) \le \|W\|$. For all W and for all $\varepsilon > 0$, there is an operator norm $\|W\|_\star \le \rho(W) + \varepsilon$. The norm $\|\cdot\|_\star$ depends on both W and ε.

Lemma 3.2 [11]. Let $\|AB\| \le \|A\| \cdot \|B\|$. Then $\|X\| < 1$ implies that $I - X$ is invertible, $(I-X)^{-1} = \sum_{i=0}^{\infty} X^i$, and $\|(I-X)^{-1}\| \le \frac{1}{1-\|X\|}$.

Now, we analyze the convergence property of the SIO iteration algorithm. First, the SIO iteration algorithm can be rewritten as the following equivalent iteration framework:

$$\begin{cases} X_{k,0} = X_k, X_0 = C, \ X_{k+1} = X_{k,l_k}, \\ X_{k,j+1} = \beta A X_{k,j} B + (\omega-\beta) A X_k B + (1-\omega) X_k + \omega C, \\ k = 0,1,2,\cdots, j = 0,1,2,\cdots, l_k - 1. \end{cases} \tag{3.10}$$

Theorem 3.2. Let l_k be the number of the inner iteration steps at the k−th outer iteration in (3.10). If $\rho(A)\rho(B) < 1$, $0 < \beta < 1$ and $\beta < \omega < \frac{2}{1+\rho(A)\rho(B)}$, then the iteration sequence $\{x_k\}_{k=0}^{\infty}$ generated by (3.10) converges to the exact solution X^* to Eq. (3.1).

Proof. By making use of the Kronecker products [11], from (3.10) it follows that

$$x_{k,j+1} = \beta G x_{k,j} + (\omega-\beta) G x_k + (1-\omega) x_k + \omega c \tag{3.11}$$

with $G = B^T \otimes A$, $x_{k,j+1} = \mathrm{vec}(X_{k,j+1})$ and $x_k = \mathrm{vec}(X_k)$, respectively.

According to (3.10) and (3.11), then we have

$$\begin{aligned} x_{k+1} &= x_{k,l_k} = \beta G x_{k,l_k-1} + (\omega-\beta) G x_k + (1-\omega) x_k + \omega c \\ &= \beta G \left(\beta G x_{k,l_k-2} + (\omega-\beta) G x_k + (1-\omega) x_k + \omega c\right) \\ &\quad + (\omega-\beta) G x_k + (1-\omega) x_k + \omega c \\ &= (\beta G)^2 x_{k,l_k-2} + \sum_{s=0}^{1} (\beta G)^s ((\omega-\beta) G + (1-\omega) I) x_k + \omega \sum_{s=0}^{1} (\beta G)^s c \\ &= (\beta G)^{l_k} x_k + \sum_{s=0}^{l_k-1} (\beta G)^s ((\omega-\beta) G + (1-\omega) I) x_k + \omega \sum_{s=0}^{l_k-1} (\beta G)^s c. \end{aligned} \tag{3.12}$$

From (3.12), then

$$x_{k+1} = F_k x_k + \omega H_k c, \ \ k = 0,1,2,\cdots \tag{3.13}$$

with

$$\begin{cases} F_k = (\beta G)^{l_k} + \sum_{s=0}^{l_k-1} (\beta G)^s ((\omega-\beta) G + (1-\omega) I), \\ H_k = \sum_{s=0}^{l_k-1} (\beta G)^s. \end{cases}$$

Since X^* is the exact solution to Eq. (3.1), then from (3.10) and (3.11) it follows that

$$x^* = F_k x^* + \omega H_k c, \ \ k = 0,1,2,\cdots. \tag{3.14}$$

Subtracting (3.14) from (3.13), then we obtain

$$x_{k+1} - x^* = F_k (x_k - x^*) = \cdots = F_k F_{k-1} \cdots F_0 (x_0 - x^*), \ \ k = 0,1,2,\cdots. \tag{3.15}$$

Let λ_i be an eigenvalue of the matrix G. Since

$$\begin{aligned} F_k &= (\beta G)^{l_k} + \sum_{s=0}^{l_k-1} (\beta G)^s((\omega-\beta)G + (1-\omega)I) \\ &= (\beta G)^{l_k} + \sum_{s=0}^{l_k-1} (\beta G)^s(I-\beta G) - \omega \sum_{s=0}^{l_k-1} (\beta G)^s(I-G) \\ &= (\beta G)^{l_k} + I - (\beta G)^{l_k} - \omega \sum_{s=0}^{l_k-1} (\beta G)^s(I-G) \\ &= I - \omega \sum_{s=0}^{l_k-1} (\beta G)^s(I-G), \end{aligned}$$

then

$$\theta_i^{(k)} = 1 - \frac{\omega(1-\lambda_i)(1-(\beta\lambda_i)^{l_k})}{1-\beta\lambda_i} \tag{3.16}$$

is an eigenvalue of F_k.

From (3.16), then

$$\begin{aligned} \left|\theta_i^{(k)}\right| &= \left|1 - \frac{\omega(1-\lambda_i)(1-(\beta\lambda_i)^{l_k})}{1-\beta\lambda_i}\right| \\ &= \left|\frac{1-\omega+(\omega-\beta)\lambda_i+\omega(1-\lambda_i)(\beta\lambda_i)^{l_k}}{1-\beta\lambda_i}\right| \\ &\le \frac{|1-\omega|+(\omega-\beta)\rho(G)+\omega(1+\rho(G))(\beta\rho(G))^{l_k}}{|1-\beta\lambda_i|} \\ &\le \frac{|1-\omega|+(\omega-\beta)\rho(G)}{1-\beta\rho(G)}. \end{aligned} \tag{3.17}$$

with an appropriate l_k and $\rho(G) = \rho(A)\rho(B) < 1$.

For the case $\beta < \omega < 1$, from (3.17) we have

$$\begin{aligned} \left|\theta_i^{(k)}\right| &\le \frac{|1-\omega|+(\omega-\beta)\rho(G)}{1-\beta\rho(G)} \\ &= \frac{1-\omega+(\omega-\beta)\rho(G)}{1-\beta\rho(G)} \\ &< \frac{1-\beta}{1-\beta\rho(G)} < 1 \end{aligned} \tag{3.18}$$

with $\rho(G) < 1$.

For the case $1 < \omega < \frac{2}{1+\rho(G)}$, from (3.17) we obtain

$$\begin{aligned} \left|\theta_i^{(k)}\right| &\le \frac{|1-\omega|+(\omega-\beta)\rho(G)}{1-\beta\rho(G)} \\ &= \frac{\omega-1+(\omega-\beta)\rho(G)}{1-\beta\rho(G)} \\ &< \frac{\frac{2}{1+\rho(G)}-1+\left(\frac{2}{1+\rho(G)}-\beta\right)\rho(G)}{1-\beta\rho(G)} \\ &= \frac{1+\rho(G)-\beta\rho(G)-\beta\rho^2(G)}{1+\rho(G)-\beta\rho(G)-\beta\rho^2(G)} = 1. \end{aligned} \tag{3.19}$$

Let $\delta := \sup_{k\in\mathbb{N}} (\rho(F_k)) < 1$ $(k = 0, 1, 2, \cdots)$ and ϱ_i be an eigenvalue of $F_kF_{k-1}\cdots F_0$, so

$$\varrho_i = \prod_{s=0}^{k}\left(1 - \frac{\omega(1-\lambda_i)(1-(\beta\lambda_i)^{l_s})}{1-\beta\lambda_i}\right).$$

Thus,

$$\rho(F_kF_{k-1}\cdots F_0) \le \rho(F_k)\rho(F_{k-1})\cdots\rho(F_0) \le \delta^{k+1} < 1.$$

By Lemmas 3.1 and 3.2, there exists an operator norm $\|\cdot\|_\chi$ such that

$$\|F_kF_{k-1}\cdots F_0\|_\chi < \hat{\delta}^{k+1}$$

with $\delta < \hat{\delta} < 1$. Thus,

$$\|x_{k+1} - x^*\|_\chi \le \|F_kF_{k-1}\cdots F_0\|_\chi \|x_0 - x^*\|_\chi < \hat{\delta}^{k+1}\|x_0 - x^*\|_\chi. \tag{3.20}$$

Therefore, the iteration sequence $\{x_k\}_{k=0}^{\infty}$ converges to the exact solution x^* as $k \to \infty$ according to (3.20) , and the proof is completed. □

Remark 1. From the analysis of the computational complexity of the SIO iteration algorithm, it increase the computational cost slightly compared with the IO iteration algorithm. In each iteration of (3.8), the SIO iteration algorithm only needs an extra scalar-matrix multiplication and matrix-matrix addition to calculate the matrix $(1-\omega)X_k$ with $o(mn)$ flops.

3.3. *The analyses of the parameters in the SIO iteration algorithm*

For the choices of the parameters β and l_k, the similar conclusions can be drawn as the corresponding parameters in the IO iteration algorithm [17], respectively.

Now, we mainly discuss the choice of the parameter ω. From (3.17), it follows that

$$\rho(F_k) = \max_{1\le i\le nm} \left|\theta_i^{(k)}\right| \le \frac{|1-\omega| + (\omega-\beta)\rho(G)}{1-\beta\rho(G)}, \tag{3.21}$$

equivalently,

$$\rho(F_k) \le \begin{cases} \dfrac{1-\omega+(\omega-\beta)\rho(G)}{1-\beta\rho(G)} \ (\beta<\omega<1), \\ \dfrac{\omega-1+(\omega-\beta)\rho(G)}{1-\beta\rho(G)} \ \left(1<\omega<\dfrac{2}{1+\rho(G)}\right). \end{cases} \tag{3.22}$$

Let $f(\omega) = \frac{|1-\omega|+(\omega-\beta)\rho(G)}{1-\beta\rho(G)}$, then from (3.21) and (3.22) we have

$$f'(\omega) = \begin{cases} \dfrac{\rho(G)-1}{(1-\beta\rho(G))^2} < 0 \ (\beta<\omega<1), \\ \dfrac{1+\rho(G)}{(1-\beta\rho(G))^2} > 0 \ \left(1<\omega<\dfrac{2}{1+\rho(G)}\right) \end{cases} \tag{3.23}$$

with $\rho(G) < 1$, which turns out that $f(\omega)$ obtains its minimum value with $\omega = 1$. Here, we mention that the optimal parameter $\omega = 1$ only minimizes the upper bound of the spectral radius $\rho(F_k)$, which may not minimize the spectral radius $\rho(F_k)$ itself. However, the SIO iteration algorithm often achieve better numerical results with the parameter $\omega > 1$ and close to 1, which are verified by the numerical experiments in Section 5.

4. The SIO iteration algorithm for solving Eqs. (2.4)

First, we reformulate Eqs.(2.4) as follows:

$$K_i - p_{ii}A_i^T K_i A_i = A_i^T \left(\sum_{j=1,j\ne i}^{N} p_{ij}K_j\right) A_i + Q_i, i \in \Omega. \tag{4.1}$$

Let

$$\tilde{Q}_i = A_i^T \left(\sum_{j=1,j\ne i}^{N} p_{ij}K_j\right) A_i + Q_i.$$

Then from (4.1) it follows that

$$K_i - p_{ii}A_i^T K_i A_i = \tilde{Q}_i, i \in \Omega. \tag{4.2}$$

Now, we apply the SIO iteration algorithm presented in Section 3 to solve each equation in (4.2). The outer iteration sequence has the following form:

$$K_i(m+1) - \beta_i p_{ii} A_i^T K_i(m+1) A_i = (\omega_i - \beta_i) p_{ii} A_i^T K_i(m) A_i + (1-\omega_i)K_i(m) + \omega_i \tilde{Q}_i(m), i \in \Omega, \tag{4.3}$$

where

$$\tilde{Q}_i(m) = A_i^T\left(\sum_{j=1, j\neq i}^{N} p_{ij}K_j(m)\right)A_i + Q_i.$$

Let $W_i(m) = (\omega_i - \beta_i)p_{ii}A_i^T K_i(m)A_i + (1-\omega_i)K_i(m) + \omega_i\tilde{Q}_i(m)$, then from (4.3) we have

$$K_i(m+1) - \beta_i p_{ii}A_i^T K_i(m+1)A_i = W_i(m),\ i \in \Omega. \tag{4.4}$$

Let $Y_i = K_i(m+1)$, then we need to solve the following matrix equations

$$Y_i - \beta_i p_{ii}A_i^T Y_i A_i = W_i(m), i \in \Omega \tag{4.5}$$

to get the approximations to $K_i(m+1)(i \in \Omega)$.

The inner iteration sequences for solving (4.5) are defined by

$$Y_i(j+1) = \beta_i p_{ii}A_i^T Y_i(j)A_i + W_i(m),\ j = 0, 1, \cdots, m_k - 1,\ i \in \Omega, \tag{4.6}$$

where $Y_i(0)$ $(i \in \Omega)$ are given by $K_i(m)$ $(i \in \Omega)$ as the initial conditions, and $Y_i(m_k)$ $(i \in \Omega)$ are treated as the approximations to $K_i(m+1)$ $(i \in \Omega)$ in (4.4).

Let the relative residual

$$\zeta = \sqrt{\sum_{i=1}^{N}\left\|K_i(m) - A_i^T\left(\sum_{j=1}^{N} p_{ij}K_j(m)\right)A_i - Q_i\right\|_F^2}, \tag{4.7}$$

where m denotes the iteration number of the SIO iteration algorithm for solving Eqs. (2.4).

Algorithm 2: The SIO iteration algorithm for solving Eqs. (2.4)

Input: $A_i, \tilde{Q}_i, \alpha_i, \beta_i, p_{ij}, \varepsilon_i, \epsilon, i, j \in \Omega$
Output: K_i

while $\zeta \geq \epsilon$
  for i=1:N
    $K_i \leftarrow \tilde{Q}_i$
    $Z_i \leftarrow p_{ii}A_i^T K_i A_i$
    while $\|\tilde{Q}_i + Z_i - K_i\|_F \geq \varepsilon_i$
      $W_i \leftarrow (1-\omega_i)K_i + (\omega_i - \beta_i)Z_i + \omega_i\tilde{Q}_i$
      for s=1: m_k
        $K_i \leftarrow \beta_i Z_i + W_i$
        $Z_i \leftarrow p_{ii}A_i^T K_i A_i$
      end
    end
  end
  Update $\zeta, \tilde{Q}_i$
end

4.1. A current-estimation-based SIO iteration algorithm for solving Eqs. (2.4)

From Algorithm 2, it is clear that the estimates $K_j(m+1)$ $(j \in \Pi[1, i-1])$ have been updated before $K_i(m+1)$ is calculated. According to the information renovation idea [17,19,20,21,22], we can use the estimates $K_1(m+1), \cdots, K_{i-1}(m+1)$ and $K_{i+1}(m), \cdots, K_N(m)$ to obtain $K_i(m+1)$, then develop the following current-estimation-based SIO iteration algorithm for solving Eqs. (2.4):

$$\begin{cases} K_i(m,0) = K_i(m),\ K_i(m+1) = K_i(m, m_k),\ i \in \Omega, \\ K_i(m, j+1) = \beta_i p_{ii}A_i^T K_i(m,j)A_i + (\omega_i - \beta_i)p_{ii}A_i^T K_i(m)A_i + (1-\omega_i)K_i(m) \\ + \omega_i\hat{Q}_i(m+1), m = 0, 1, 2, \cdots, j = 0, 1, 2, \cdots, m_k - 1, \end{cases} \tag{4.8}$$

where

$$\hat{Q}_i(m+1) = A_i^T \left(\sum_{\tau=1}^{i-1} p_{i\tau} K_\tau(m+1) + \sum_{\tau=i+1}^{N} p_{i\tau} K_\tau(m) \right) A_i + Q_i.$$

Lemma 4.1. Assume that Eqs. (2.4) have unique positive semidefinite or positive definite solutions for $Q_i \geq 0$ $(i \in \Omega)$, and the matrices A_i $(i \in \Omega)$ are Schur stable. If $0 < \beta_i \leq \omega_i < 1$ for $i \in \Omega$, then the matrix tuple $\mathcal{K}(m) = \{K_1(m), K_2(m), \cdots, K_N(m)\}$ generated by Algorithm (4.8) with zero initial conditions is upper bounded by the solution $\mathcal{K} = \{K_1, K_2, \cdots, K_N\}$ to Eqs. (2.4). Namely, for any integer $m \geq 0$, we have

$$K_i(m) \leq K_i, \ i \in \Omega. \tag{4.9}$$

Proof. Due to zero initial conditions, it is clear that $K_i(0) \leq K_i$ $(i \in \Omega)$. Now, we assume that

$$K_i(l) \leq K_i, i \in \Omega \tag{4.10}$$

by the principle of the mathematical induction for $m = l$ and $l \geq 0$.

From (4.8), it follows that

$$\begin{aligned} K_i(l+1) &= (\beta_i p_{ii})^{m_k} (A_i^{m_k})^T K_i(l) A_i^{m_k} + (\omega_i - \beta_i) p_{ii} \sum_{s=0}^{m_k-1} (\beta_i p_{ii})^s (A_i^{s+1})^T K_i(l) A_i^{s+1} \\ &+ \omega_i \sum_{s=0}^{m_k-1} (\beta_i p_{ii})^s (A_i^{s+1})^T \left(\sum_{\tau=1}^{i-1} p_{i\tau} K_\tau(l+1) + \sum_{\tau=i+1}^{N} p_{i\tau} K_\tau(l) \right) A_i^{s+1} \\ &+ (1-\omega_i) \sum_{s=0}^{m_k-1} (\beta_i p_{ii} A_i^T)^s K_i(l) A_i^s + \omega_i \sum_{s=0}^{m_k-1} (\beta_i p_{ii} A_i^T)^s Q_i A_i^s, i \in \Omega. \end{aligned} \tag{4.11}$$

and

$$\begin{aligned} K_i &= (\beta_i p_{ii})^{m_k} (A_i^{m_k})^T K_i A_i^{m_k} + (\omega_i - \beta_i) p_{ii} \sum_{s=0}^{m_k-1} (\beta_i p_{ii})^s (A_i^{s+1})^T K_i A_i^{s+1} \\ &+ \omega_i \sum_{s=0}^{m_k-1} (\beta_i p_{ii})^s (A_i^{s+1})^T \left(\sum_{\tau=1, \tau \neq i}^{N} p_{i\tau} K_\tau \right) A_i^{s+1} + \omega_i \sum_{s=0}^{m_k-1} (\beta_i p_{ii} A_i^T)^s Q_i A_i^s \\ &+ (1-\omega_i) \sum_{s=0}^{m_k-1} (\beta_i p_{ii} A_i^T)^s K_i A_i^s, i \in \Omega. \end{aligned} \tag{4.12}$$

Subtracting (4.11) from (4.12), then

$$\begin{aligned} & K_i - K_i(l+1) \\ &= (\beta_i p_{ii})^{m_k} (A_i^{m_k})^T (K_i - K_i(l)) A_i^{m_k} + (1-\omega_i) \sum_{s=0}^{m_k-1} (\beta_i p_{ii} A_i^T)^s (K_i - K_i(l)) A_i^s \\ &+ \omega_i \sum_{s=0}^{m_k-1} (\beta_i p_{ii})^s (A_i^{s+1})^T \left(\sum_{\tau=1}^{i-1} p_{i\tau} (K_\tau - K_\tau(l+1)) + \sum_{\tau=i+1}^{N} p_{i\tau} (K_\tau - K_\tau(l)) \right) A_i^{s+1} \\ &+ (\omega_i - \beta_i) p_{ii} \sum_{s=0}^{m_k-1} (\beta_i p_{ii})^s (A_i^{s+1})^T (K_i - K_i(l)) A_i^{s+1}, i \in \Omega. \end{aligned} \tag{4.13}$$

For $i = 1$, from (4.10) and (4.13), we have

$$\begin{aligned} & K_1 - K_1(l+1) \\ &= (\beta_1 p_{11})^{m_k} (A_1^{m_k})^T (K_1 - K_1(l)) A_1^{m_k} + (\omega_1 - \beta_1) p_{11} \sum_{s=0}^{m_k-1} (\beta_1 p_{11})^s (A_1^{s+1})^T (K_1 - K_1(l)) A_1^{s+1} \\ &+ \omega_1 \sum_{s=0}^{m_k-1} (\beta_1 p_{11})^s (A_1^{s+1})^T \left(\sum_{\tau=2}^{N} p_{1\tau} (K_\tau - K_\tau(l)) \right) A_1^{s+1} + (1-\omega_1) \sum_{s=0}^{m_k-1} (\beta_1 p_{11} A_1^T)^s (A_1^s)^T (K_1 - K_1(l)) A_1^s. \end{aligned}$$

Hence, $K_1(l+1) \leq K_1$ with $0 < \beta_1 \leq \omega_1 < 1$.

Now, it is assumed that

$$K_\gamma(l+1) \leq K_\gamma, \gamma \in \Pi[1, t-1] \tag{4.14}$$

with $t \geq 2$. From (4.10), (4.13) and (4.14), it is clear that

$$\begin{aligned}
&K_t - K_t(l+1) \\
&= (\beta_t p_{tt})^{m_k}(A_t^{m_k})^T(K_t - K_t(l))A_t^{m_k} + (\omega_t - \beta_t)p_{tt}\sum_{s=0}^{m_k-1}(\beta_t p_{tt})^s(A_t^{s+1})^T(K_t - K_t(l))A_t^{s+1} \\
&+\omega_t\sum_{s=0}^{m_k-1}(\beta_t p_{tt})^s(A_t^{s+1})^T\left(\sum_{\tau=1}^{t-1}p_{t\tau}(K_\tau - K_\tau(l+1)) + \sum_{\tau=t+1}^{N}p_{t\tau}(K_\tau - K_\tau(l))\right)A_t^{s+1} \\
&+(1-\omega_t)\sum_{s=0}^{m_k-1}(\beta_t p_{tt}A_t^T)^s(K_t - K_t(l))A_t^s, t \in \Omega.
\end{aligned} \tag{4.15}$$

with $0 < \beta_t \leq \omega_t < 1$. Therefore, $K_t(l+1) \leq K_t$. By the induction principle, we have

$$K_i(l+1) \leq K_i, i \in \Omega. \tag{4.16}$$

By (4.10), (4.14) and (4.16), we obtain $K_i(m) \leq K_i(i \in \Omega)$ for any integer $m \geq 0$, then (4.9) holds by the mathematical induction, and the proof is completed. □

Lemma 4.2. Assume that Eqs. (2.4) have unique positive semidefinite or positive definite solutions for $Q_i \geq 0(i \in \Omega)$, and the matrices $A_i(i \in \Omega)$ are Schur stable. If $0 < \beta_i \leq \omega_i < 1$ for $i \in \Omega$, then the matrix tuple $\mathcal{K}(m) = \{K_1(m), K_2(m), \cdots, K_N(m)\}$ derived from Algorithm (4.8) with zero initial conditions is strictly monotonically increasing. Namely, for any integer $m \geq 0$, we have

$$K_i(m) \leq K_i(m+1), i \in \Omega. \tag{4.17}$$

Proof. From (4.11), it is clear that

$$\begin{aligned}
K_i(1) &= (\beta_i p_{ii})^{m_k}(A_i^{m_k})^T K_i(0)A_i^{m_k} + (\omega_i - \beta_i)p_{ii}\sum_{s=0}^{m_k-1}(\beta_i p_{ii})^s(A_i^{s+1})^T K_i(0)A_i^{s+1} \\
&+\omega_i\sum_{s=0}^{m_k-1}(\beta_i p_{ii})^s(A_i^{s+1})^T\left(\sum_{\tau=1}^{i-1}p_{i\tau}K_\tau(1) + \sum_{\tau=i+1}^{N}p_{i\tau}K_\tau(0)\right)A_i^{s+1} \\
&+(1-\omega_i)\sum_{s=0}^{m_k-1}(\beta_i p_{ii}A_i^T)^s K_i(0)A_i^s + \omega_i\sum_{s=0}^{m_k-1}(\beta_i p_{ii}A_i^T)^s Q_i A_i^s, i \in \Omega.
\end{aligned} \tag{4.18}$$

Since $K_i(0) = 0$ $(i \in \Omega)$, then from (4.18) we have

$$K_i(1) = \omega_i\sum_{s=0}^{m_k-1}(\beta_i p_{ii})^s(A_i^{s+1})^T\left(\sum_{\tau=1}^{i-1}p_{i\tau}K_\tau(1)\right)A_i^{s+1} + \omega_i\sum_{s=0}^{m_k-1}(\beta_i p_{ii}A_i^T)^s Q_i A_i^s. \tag{4.19}$$

For $i = 1$, from (4.19), it follows that

$$K_1(1) = \omega_1\sum_{s=0}^{m_k-1}(\beta_1 p_{11}A_1^T)^s Q_1 A_1^s \geq 0 \tag{4.20}$$

with $Q_1 \geq 0$. Thus, $0 = K_1(0) \leq K_1(1)$.

For $i = 2$, from (4.19) and (4.20), we obtain

$$\begin{aligned}
K_2(1) &= \omega_2 p_{21}\sum_{s=0}^{m_k-1}(\beta_2 p_{22})^s(A_2^{s+1})^T K_1(1)A_2^{s+1} + \omega_2\sum_{s=0}^{m_k-1}(\beta_2 p_{22}A_2^T)^s Q_2 A_2^s \\
&\geq \omega_2 p_{21}\sum_{s=0}^{m_k-1}(\beta_2 p_{22})^s(A_2^{s+1})^T K_1(0)A_2^{s+1} + \omega_2\sum_{s=0}^{m_k-1}(\beta_2 p_{22}A_2^T)^s Q_2 A_2^s \\
&= \omega_2\sum_{s=0}^{m_k-1}(\beta_2 p_{22}A_2^T)^s Q_2 A_2^s \geq 0
\end{aligned} \tag{4.21}$$

with $Q_2 \geq 0$, then $0 = K_2(0) \leq K_2(1)$.

According to (4.19), (4.20) and (4.21), then it turns out that

$$K_\theta(1) \geq \omega_\theta \sum_{s=0}^{m_k-1} (\beta_\theta p_{\theta\theta} A_\theta^T)^s Q_\theta A_\theta^s \geq 0, \theta \in \Pi[3, N]$$

with $Q_\theta \geq 0$. Then $K_i(0) \leq K_i(1)$ ($i \in \Omega$) hold.

Now, we assume that

$$K_i(l) \leq K_i(l+1),\ i \in \Omega. \tag{4.22}$$

From (4.11), then

$$\begin{aligned}
&K_i(l+2) - K_i(l+1) \\
&= (\beta_i p_{ii})^{m_k} (A_i^{m_k})^T (K_i(l+1) - K_i(l)) A_i^{m_k} + (1-\omega_i) \sum_{s=0}^{m_k-1} (\beta_i p_{ii} A_i^T)^s (K_i(l+1) - K_i(l)) A_i^s \\
&+\omega_i \sum_{s=0}^{m_k-1} (\beta_i p_{ii})^s (A_i^{s+1})^T \left(\sum_{\tau=1}^{i-1} p_{i\tau} (K_\tau(l+2) - K_\tau(l+1)) + \sum_{\tau=i+1}^{N} p_{i\tau} (K_\tau(l+1) - K_\tau(l)) \right) A_i^{s+1} \\
&+(\omega_i - \beta_i) p_{ii} \sum_{s=0}^{m_k-1} (\beta_i p_{ii})^s (A_i^{s+1})^T (K_i(l+1) - K_i(l)) A_i^{s+1}, i \in \Omega. \qquad (4.23)
\end{aligned}$$

For $i = 1$, from (4.22) and (4.23), it follows that

$$\begin{aligned}
&K_1(l+2) - K_1(l+1) \\
&= (\beta_1 p_{11})^{m_k} (A_1^{m_k})^T (K_1(l+1) - K_1(l)) A_1^{m_k} + (1-\omega_1) \sum_{s=0}^{m_k-1} (\beta_1 p_{11} A_1^T)^s (K_1(l+1) - K_1(l)) A_1^s \\
&+\omega_1 \sum_{s=0}^{m_k-1} (\beta_1 p_{11})^s (A_1^{s+1})^T \left(\sum_{\tau=2}^{N} p_{1\tau} (K_\tau(l+1) - K_\tau(l)) \right) A_1^{s+1} \\
&+(\omega_1 - \beta_1) p_{11} \sum_{s=0}^{m_k-1} (\beta_1 p_{11})^s (A_1^{s+1})^T (K_1(l+1) - K_1(l)) A_1^{s+1}, \qquad (4.24)
\end{aligned}$$

Therefore, $K_1(l+1) \leq K_1(l+2)$ with $0 < \beta_1 \leq \omega_1 < 1$.

Now, it is assumed that

$$K_\gamma(l+1) \leq K_\gamma(l+2),\ \gamma \in \Pi[1, t-1] \tag{4.25}$$

with $t \geq 2$.

For $i = t$, from (4.24) and (4.25), then we have

$$\begin{aligned}
&K_t(l+2) - K_t(l+1) \\
&= (\beta_t p_{tt})^{m_k} (A_t^{m_k})^T (K_t(l+1) - K_t(l)) A_t^{m_k} + (1-\omega_t) \sum_{s=0}^{m_k-1} (\beta_t p_{tt} A_t^T)^s (K_t(l+1) - K_t(l)) A_t^s \\
&+\omega_t \sum_{s=0}^{m_k-1} (\beta_t p_{tt})^s (A_t^{s+1})^T \left(\sum_{\tau=1}^{t-1} p_{t\tau} (K_\tau(l+2) - K_\tau(l+1)) + \sum_{\tau=t+1}^{N} p_{t\tau} (K_\tau(l+1) - K_\tau(l)) \right) A_t^{s+1} \\
&+(\omega_t - \beta_t) p_{tt} \sum_{s=0}^{m_k-1} (\beta_t p_{tt})^s (A_t^{s+1})^T (K_t(l+1) - K_t(l)) A_t^{s+1}, t \in \Omega.
\end{aligned}$$

with $0 < \beta_t \leq \omega_t < 1$. By the principle of mathematical induction, then $K_i(l+1) \leq K_i(l+2) (i \in \Omega)$. Thus, the relation (4.18) hold for any integer $m \geq 0$, and the proof is completed. □

Theorem 4.1. If Eqs. (2.4) have unique positive semidefinite or positive definite solutions for $Q_i \geq 0$ ($i \in \Omega$), and the matrices A_i ($i \in \Omega$) are Schur stable, then the matrix tuple $\mathcal{K}(m) = \{K_1(m), K_2(m), \cdots, K_N(m)\}$ obtained from Algorithm (4.8) with zero initial conditions converges to the unique solution $\mathcal{K} = \{K_1, K_2, \cdots, K_N\}$ to Eqs. (2.4) for $0 < \beta_i \leq \omega_i < 1$ ($i \in \Omega$). Namely, $\lim_{m\to\infty} K_i(m) = K_i,\ i \in \Omega$.

Proof. From Lemmas 4.1 and 4.2, the iteration sequence $\mathcal{K}(m) = \{K_1(m), K_2(m), \cdots, K_N(m)\}$ generated by Algorithm (4.8) is monotonically nondecreasing and upper bounded by the solutions to Eqs. (2.4), then

$$0 = K_i(0) \leq K_i(1) \leq K_i(2) \leq \cdots \leq K_i(m) \leq K_i(m+1) \leq \cdots \leq K_i,\ i \in \Omega. \tag{4.26}$$

From [3], it is obvious that the matrix tuple $\mathcal{K}(m) = \{K_1(m), K_2(m), \cdots, K_N(m)\}$ is convergent.

Let $\lim_{m\to\infty} K_i(m) = K_i(\infty)$ $(i \in \Omega)$ and substitute $K_i(\infty)$ into (4.11), then

$$\begin{aligned} K_i(\infty) &= (\beta_i p_{ii})^{m_k}(A_i^{m_k})^T K_i(\infty)A_i^{m_k} + (\omega_i - \beta_i)p_{ii}\sum_{s=0}^{m_k-1}(\beta_i p_{ii})^s(A_i^{s+1})^T K_i(\infty)A_i^{s+1} \\ &+\omega_i \sum_{s=0}^{m_k-1}(\beta_i p_{ii})^s(A_i^{s+1})^T\left(\sum_{\tau=1}^{i-1} p_{i\tau}K_\tau(\infty) + \sum_{\tau=i+1}^{N} p_{i\tau}K_\tau(\infty)\right)A_i^{s+1} \\ &+(1-\omega_i)\sum_{s=0}^{m_k-1}(\beta_i p_{ii}A_i^T)^s K_i(\infty)A_i^s + \omega_i\sum_{s=0}^{m_k-1}(\beta_i p_{ii}A_i^T)^s Q_i A_i^s, i \in \Omega. \end{aligned} \tag{4.27}$$

From Algorithm (4.8), it is clear that (4.27) is equivalent to

$$K_i(\infty) = A_i^T\left(\sum_{j=1}^{N} p_{ij}K_j(\infty)\right)A_i + Q_i,\ i \in \Omega. \tag{4.28}$$

Therefore, $\mathcal{K}(\infty) = \{K_1(\infty), K_2(\infty), \cdots, K_N(\infty)\}$ is the solution to Eqs. (2.4). Since Eqs. (2.4) have unique solutions by the assumptions, so $\mathcal{K}(\infty) = \{K_1(\infty), K_2(\infty), \cdots, K_N(\infty)\}$ is the unique solutions to Eqs.(2.4) and $K_i(\infty) = K_i$ $(i \in \Omega)$. Thus, the proof is completed. □

Remark 2. Theorem 4.1 shows that Algorithm (4.8) is convergent under zero initial conditions with $\omega_i < 1(i \in \Omega)$. In fact, from the proof of Lemmas 1 and 2, it is clear that these results may holds for the case $1 < \omega_i < \varpi$, where ϖ is a positive real scalar close to 1, which is illustrated by Example 2 in Section 5.

From Theorem 4.1, it follows that Algorithm (4.8) is only efficient for zero initial conditions though it has the monotonically nondecreasing property. Next, we give a convergence result of Algorithm (4.8) for any initial conditions.

Theorem 4.2 Assume that Eqs. (2.4) have unique solutions $\mathcal{K} = \{K_1, K_2, \cdots, K_N\}$. The matrix tuple $\mathcal{K}(m) = \{K_1(m), K_2(m), \cdots, K_N(m)\}$ derived from Algorithm (4.8) converges to $\mathcal{K} = \{K_1, K_2, \cdots, K_N\}$ with any initial conditions if and only if $\mathcal{H}$ is invertible and $\rho(\mathcal{H}^{-1}\mathcal{F}) < 1$, where $\mathcal{H}$ is a lower triangular matrix with

$$\mathcal{H}_{ii} = I, \mathcal{H}_{i\tau} = -\omega_i p_{i\tau}\sum_{s=0}^{m_k-1}(\beta_i p_{ii})^s(A_i^{s+1}\otimes A_i^{s+1})^T, \tau < i, i \in \Omega$$

and $\mathcal{F}$ is an upper triangular matrix with

$$\begin{cases} \mathcal{F}_{ii} = (\beta_i p_{ii})^{m_k}(A_i^{m_k}\otimes A_i^{m_k})^T + (\omega_i - \beta_i)p_{ii}\sum_{s=0}^{m_k-1}(\beta_i p_{ii})^s(A_i^{s+1}\otimes A_i^{s+1})^T \\ + (1-\omega_i)\sum_{s=0}^{m_k-1}(\beta_i p_{ii})^s(A_i^s\otimes A_i^s)^T, \\ \mathcal{F}_{i\tau} = \omega_i p_{i\tau}\sum_{s=0}^{m_k-1}(\beta_i p_{ii})^s(A_i^{s+1}\otimes A_i^{s+1})^T, \tau > i, i \in \Omega. \end{cases}$$

Proof. By (4.11) and the properties of Kronecker products [11], we have

$$\begin{aligned} &\mathrm{vec}(K_i(m+1)) \\ &= (\beta_i p_{ii})^{m_k}(A_i^{m_k}\otimes A_i^{m_k})^T\mathrm{vec}(K_i(l)) + (\omega_i - \beta_i)p_{ii}\sum_{s=0}^{m_k-1}(\beta_i p_{ii})^s(A_i^{s+1}\otimes A_i^{s+1})^T\mathrm{vec}(K_i(l)) \\ &+\omega_i\sum_{s=0}^{m_k-1}(\beta_i p_{ii})^s(A_i^{s+1}\otimes A_i^{s+1})^T\left(\sum_{\tau=1}^{i-1}p_{i\tau}\mathrm{vec}(K_\tau(l+1)) + \sum_{\tau=i+1}^{N}p_{i\tau}\mathrm{vec}(K_\tau(l))\right) \\ &+(1-\omega_i)\sum_{s=0}^{m_k-1}(\beta_i p_{ii})^s(A_i^s\otimes A_i^s)^T\mathrm{vec}(K_i(l)) + \omega_i\sum_{s=0}^{m_k-1}(\beta_i p_{ii})^s(A_i^s\otimes A_i^s)^T\mathrm{vec}(Q_i), \\ &i \in \Omega. \end{aligned} \tag{4.29}$$

Let

$$\phi(m) = \left(\mathrm{vec}(K_1(m))^T\ \mathrm{vec}(K_2(m))^T \cdots \mathrm{vec}(K_N(m))^T\right)^T,$$

$$\delta = \left(\mathrm{vec}(Q_1)^T\ \mathrm{vec}(Q_2)^T \cdots \mathrm{vec}(Q_N)^T\right)^T,$$

and Ψ be a diagonal matrix with $\Psi(ii) = \omega_i \sum_{s=0}^{m_k-1} (\beta_i p_{ii})^s (A_i^s \otimes A_i^s)^T$ $(i \in \Omega)$.

Then from (4.29) we have

$$\mathcal{H}\phi(m+1) = \mathcal{F}\phi(m) + \Psi\delta. \tag{4.30}$$

Since $\mathcal{H}$ is an invertible matrix, then from (4.30) it follows that

$$\phi(m+1) = \mathcal{H}^{-1}\mathcal{F}\phi(m) + \mathcal{H}^{-1}\Psi\delta. \tag{4.31}$$

From (4.31), we obtain the following recursive relation

$$\phi(m+1) = (\mathcal{H}^{-1}\mathcal{F})^{m+1}\phi(0) + \sum_{i=0}^{m} (\mathcal{H}^{-1}\mathcal{F})^i \mathcal{H}^{-1}\Psi\delta. \tag{4.32}$$

Since $\rho(\mathcal{H}^{-1}\mathcal{F}) < 1$, then

$$\begin{aligned}\lim_{m\to\infty} \phi(m+1) &= \lim_{m\to\infty} \left((\mathcal{H}^{-1}\mathcal{F})^{m+1}\phi(0) + \sum_{i=0}^{m} (\mathcal{H}^{-1}\mathcal{F})^i \mathcal{H}^{-1}\Psi\delta\right) \\ &= (I - \mathcal{H}^{-1}\mathcal{F})^{-1}\mathcal{H}^{-1}\Psi\delta = (\mathcal{H} - \mathcal{F})^{-1}\Psi\delta.\end{aligned}$$

Let $\phi = \left(\mathrm{vec}(K_1)^T\ \mathrm{vec}(K_2)^T \cdots \mathrm{vec}(K_N)^T\right)^T$, then from (4.12) we have

$$\mathcal{H}\phi = \mathcal{F}\phi + \Psi\delta, \tag{4.33}$$

and the exact solutions to Eqs. (2.4) are

$$\phi = (\mathcal{H} - \mathcal{F})^{-1}\Psi\delta.$$

Then

$$\lim_{m\to\infty} \phi(m+1) = \phi$$

and the proof is completed. □

Remark 3. If $\omega_i = 1$ $(i \in \Omega)$, then Algorithms 2 and (4.8) are just the IO and current-estimation-based IO (CIO) iteration algorithms in [17], respectively.

Remark 4. When Algorithms 2 and (4.8) are used to solve the CLMEs (2.4), N discrete Lyapunov matrix equations need to be solved in each iteration step, so for the choices of the parameters ω_i, β_i and m_k in each Lyapunov matrix equation, the similar conclusions can be obtained as those in Section 3.3.

5. Numerical results

In this section, we present two numerical examples to illustrate the convergence performances of the SIO iteration algorithms for solving Eqs. (2.4) and (3.1), respectively. The numerical experiments are performed in Matlab R2010 on an Intel dual core processor (2.30 GHz, 8 GB RAM). Three iteration parameters are used to test the proposed algorithms, which are iteration step (denoted as IT), computing time in seconds (denoted as CPU), and residual (denoted as RES), where RES is defined by

$$\sqrt{\sum_{i=1}^{N} \left\| K_i(m) - A_i^T \left(\sum_{j=1}^{N} p_{ij} K_j(m) \right) A_i - Q_i \right\|_F^2}.$$

Example 1 [16,17,27]. Consider the following discrete Lyapunov matrix equation $X - AXA^T = C$ with

$$A = \begin{bmatrix} 0 & \nu & & & & \\ -\nu & 0 & \nu & & & \\ & -\nu & 0 & \cdot & & \\ & & \cdot & \cdot & \cdot & \\ & & & \cdot & \cdot & \nu \\ & & & & -\nu & 0 \end{bmatrix}$$

and $C = I$. The eigenvalues of A are $\lambda_j = 2i|\nu|\cos\frac{\pi j}{n+1},\ j = 1, \cdots, n$. It is obvious that $\rho(A)$ approaches to 1 with large n if τ is close to 0.5. For this case, the Smith method performs very poorly.

In this example, we compare the SIO iteration algorithm with the IO iteration algorithm and Smith method for $n = 800$ in Tables 1, where we choose $\beta = 0.8, s_k = l_k = 2$ in the IO and SIO iteration algorithms, and $\omega = 1.25$ in the SIO iteration algorithm, respectively. From these numerical experiments, it follows that these iteration algorithms need more iteration numbers and CPU time with the values of ν increasing. Moreover, the SIO iteration algorithm is more effective than the IO iteration algorithm and Smith method in both the number of iteration steps and CPU time, especially for larger τ, such as the case $\nu = 0.499$. Figure 1 depicts the curves of the iteration numbers of the SIO iteration algorithm (Algorithm 1) for different n, ν, ω with $\beta = 0.65, l_k = 2$, respectively. Figure 2 illustrates the iteration numbers of the SIO iteration algorithm (Algorithm 1) for different n, β, ω with $\nu = 0.46, l_k = 2$, respectively. Figs. 1, 2 show that the SIO iteration algorithm has better convergence performances with $\omega > 1$, which is consistent with the analysis in Section 3.3. In addition, the SIO iteration algorithm is more efficient with larger ω for larger ν, and the optimal parameter ω is not sensitive to the changes of n.

Table 1: Numerical results of the different iteration algorithms with $n = 800$

	The Smith method			The IO iteration algorithm			The SIO iteration algorithm		
ν	IT	CPU	RES	IT	CPU	RES	IT	CPU	RES
0.45	35	12.279	1.07×10^{-9}	18	5.1590	1.16×10^{-9}	14	4.7579	3.72×10^{-9}
0.47	54	15.972	1.12×10^{-9}	28	7.1237	1.02×10^{-9}	21	5.8637	1.23×10^{-9}
0.495	222	60.315	1.23×10^{-9}	110	29.182	1.19×10^{-9}	87	24.065	1.22×10^{-9}
0.499	688	177.35	1.24×10^{-9}	322	88.526	1.24×10^{-9}	257	72.669	1.24×10^{-9}

Example 2 [8,9]. Consider the couple Lyapunov matrix equations in the form of Eqs. (2.4) with system matrices

$$A_1 = \begin{bmatrix} 0.0667 & 0.0665 & 0.0844 & -0.2257 \\ 0.1383 & -0.1309 & 0.0797 & 0.1162 \\ 0.0658 & 0.0298 & 0.0645 & -0.1018 \\ -0.2283 & 0.2438 & -0.1990 & 0.2997 \end{bmatrix},$$

$$A_2 = \begin{bmatrix} 0.1885 & -0.3930 & -0.0894 & -0.1919 \\ -0.4230 & 0.3598 & -0.1224 & -0.1548 \\ 0.0350 & -0.1950 & -0.1967 & -0.1017 \\ -0.2648 & -0.0240 & -0.0542 & 0.0484 \end{bmatrix},$$

$$A_3 = \begin{bmatrix} 0.2746 & 0.0634 & 0.3414 & -0.0692 \\ 0.0769 & 0.4167 & 0.0283 & -0.1207 \\ -0.1607 & 0.0344 & -0.2227 & 0.1617 \\ 0.1175 & -0.2969 & 0.4149 & 0.3314 \end{bmatrix}$$

and transition rate matrix

$$P = \begin{bmatrix} 0.1 & 0.3 & 0.6 \\ 0.5 & 0.25 & 0.25 \\ 0 & 0.3 & 0.7 \end{bmatrix}.$$

In this example, the positive definite matrices $Q_i = I_4$ $(i = 1, 2, 3)$ are chosen as identity matrices. The number of the subsystems is $N = 3$, and the dimension of the system is $n = 4$.

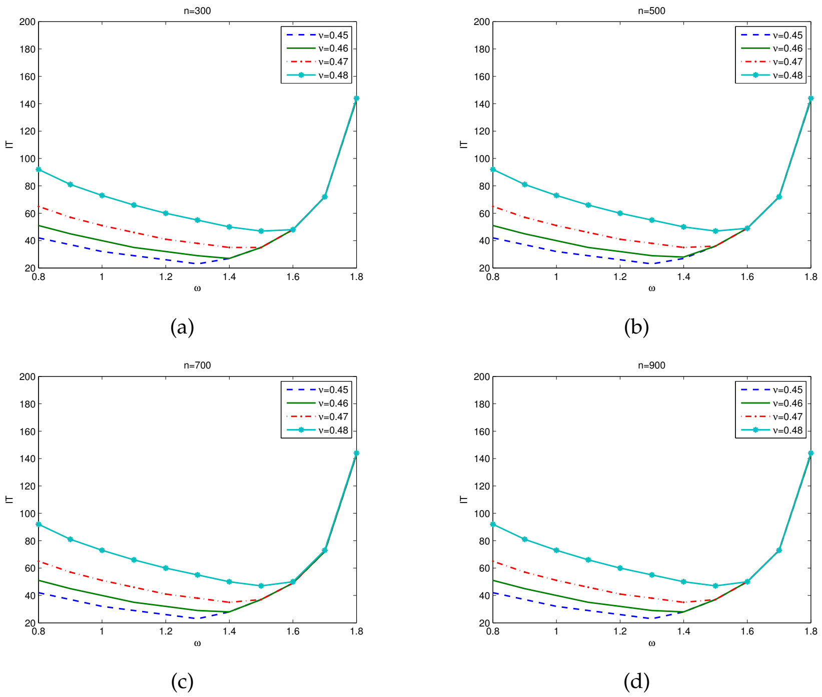


Figure 1: Convergence curves of Algorithm 1 for different n, ν and ω.

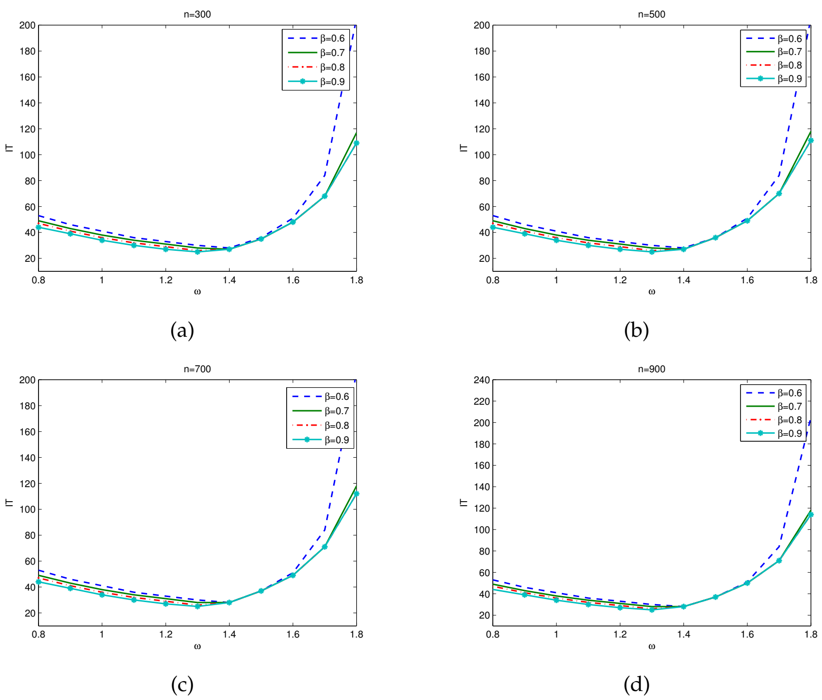


Figure 2: Convergence curves of Algorithm 1 for different n, β and ω.

First, we compare Algorithm 2 with the IO iteration algorithm [17], and Algorithm (4.8) with the CIO iteration algorithm [17], respectively. Let $m_k = 2, \beta_1 = \beta_2 = \beta_3 = 0.6$ in these algorithms , and $\omega_1 = \omega_2 = \omega_3 = 1.05$ in Algorithms 2 and (4.8), respectively. These algorithms all start with the zero initial conditions. The numerical results are reported in Figs. 3 and 4, which show that Algorithms 2 and (4.8) converges faster than the IO and CIO iteration algorithms, respectively.

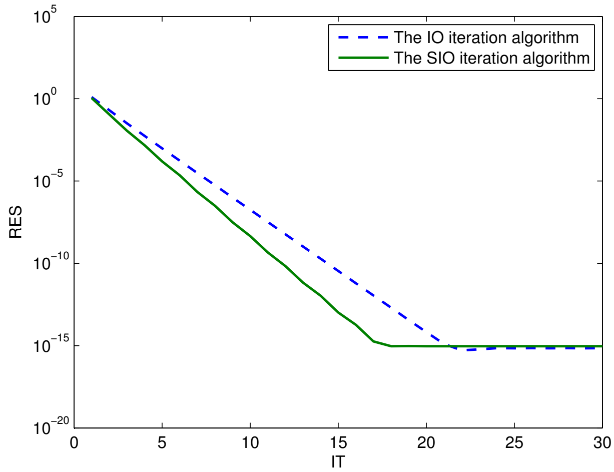


Figure 3: Convergence curves of the IO iteration algorithm and Algorithm 2.

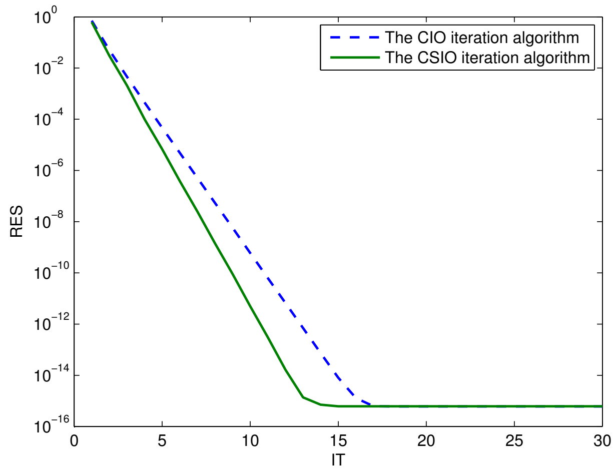
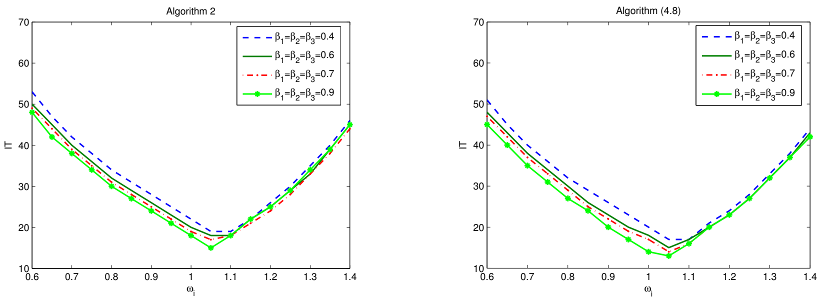


Figure 4: Convergence curves of the CIO iteration algorithm and Algorithms (4.8).

Next, we will investigate the convergence performances of the proposed algorithms for different parameters. Figs. 5 depict the convergence curves of Algorithms 2, (4.8) with different ω_i, β_i $(i \in \Omega)$, where $\omega_1 = \omega_2 = \omega_3 \in [0.6, 1.4]$, $m_k = 2$ and $\beta_1 = \beta_2 = \beta_3 = 0.4, 0.6, 0.7, 0.9$. From the numerical results in Fig. 5, it is obvious that Algorithms 2 and (4.8) converges to the exact solutions of Eqs. (2.4) for a given stopping criterion with different parameters ω_i, β_i $(i \in \Omega)$, respectively. Furthermore, Algorithms 2 and (4.8) perform better with $\omega_i \in (1, 1.1)$ $(i \in \Omega)$, and the optimal parameters ω_i can be found close to 1, just as the analyses in Section 3.3 and Remark 4.



(a) (b)

Figure 5: Convergence curves of Algorithms 2 and (4.8) with different parameters.

Finally, we compare the convergence performance of Algorithms (4.8) with the existing iteration algorithms (2.6)-(2.9) under the following initial conditions:

$$K_1(0) = \begin{bmatrix} 1 & 2 & 0.5 & 0.3 \\ 0 & 0 & 1.2 & 2.5 \\ 3 & 0.2 & 0.8 & -0.6 \\ 2 & -3 & 0 & 0.8 \end{bmatrix},$$

$$K_2(0) = \begin{bmatrix} -1 & 0.5 & 0.7 & 0.3 \\ 1 & 0 & 0.9 & -0.6 \\ 2 & 3.2 & -0.8 & -1 \\ 0 & 2.1 & -1 & 0.75 \end{bmatrix},$$

$$K_3(0) = \begin{bmatrix} 0.8 & -0.5 & 1.6 & -3.1 \\ 0.15 & 2.3 & -0.7 & 0.8 \\ -2.2 & 0.2 & 2.8 & 1.5 \\ 0.3 & -2.1 & 1.5 & -0.5 \end{bmatrix}.$$

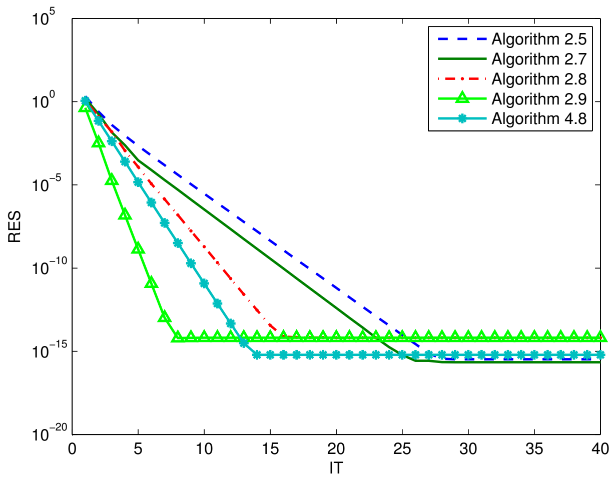


Figure 6: Convergence curves of different algorithms.

The numerical results are reported in Figure 6, where we choose $\omega_1 = \omega_2 = \omega_3 = 1.05, \beta_1 = \beta_2 = \beta_2 = 0.85$ and $m_k = 2$ in Algorithm (4.8). Here, we notice that Algorithm (4.8) needs less iteration number than Algorithms (2.6)-(2.8). Although Algorithm (2.9) performs better than Algorithm (4.8) with respect to the iteration number, it is an implicit iteration algorithm, which needs to solve N standard discrete Lyapunov matrix equations in each iteration step by the usage of the function "dlyap" , and has an enormous cost for large Eqs. (2.4). Since Algorithms 2, (4.8) are explicit iteration algorithms, so they are more efficient than the implicit iteration algorithm (2.9) for solving large Eqs. (2.4).

6. Conclusions

In this paper, an SIO iteration algorithm is presented for solving the Sylvester matrix equation and CLMEs (2.4), respectively. Numerical experiments show the stability and robustness of the proposed algorithms. Since these algorithms are parameter-dependent, then how to find the optimal parameters will be further investigated in our future work.

7. Acknowledgements

The authors are grateful to thank the anonymous referee for their recommendations and valuable suggestions and Professor Dijana Mosic for the communication.

References

[1] F. Ding and T. Chen, Iterative least-square solutions of coupled Sylvester matrix equations, Syst. Control Lett 54 (2005) 95-107.
[2] O. L. V. Costa and M. D. Fragoso, Stability results for discrete-time linear systems with Markovian jumping parameters, J. Math. Anal. Appl 179 (1993) 154-178.
[3] J. Bibby, Axiomatisations of the average and a further generalisation of monotonic sequences, Glasg. Math. J 15 (1) (1974) 65.
[4] Y. Ji, H. J. Chizeck, X. Feng, and K. A. Loparo, Stability and control of discrete-time jump linear systems, Control Theory Adv. Technol 7 (1991) 247-270.
[5] B. Zhou, G. R. Duan, and Z. Y. Li, Gradient based iterative algorithm for solving coupled matrix equations, Syst. Control Lett 58 (2009) 327-333.
[6] L. Zhang, B. Huang, and J. Lam, H_∞ model reduction of Markovian jump linear systems, Syst. Control Lett 50 (2003) 103-118.
[7] L. Tong, A. G.Wu, and G. R. Duan, Finite iterative algorithm for solving coupled Lyapunov equations appearing in discrete-time Markov jump linear systems, IET Control Theory Appl 4 (2010) 2223-2231.
[8] I. Borno, Z. Gajic, Parallel algorithm for solving coupled algebraic Lyapunov equations of discrete-time jump linear systems, Comput. Math. Appl 30 (7) (1995) 1-4.
[9] A.G.Wu, G.R.Duan, New Iterative Algorithms for Solving Coupled Markovian Jump Lyapunov Equations, IEEE Trans. Autom. Control 60 (1) (2015) 289-294.
[10] C.S. Kubrusly, O.L.V. Costa, Mean square stability conditions for discrete stochastic bilinear systems, IEEE Trans. Autom. Control 30 (11) (1985) 1082-1087.
[11] J.W. Demmel, Applied numerical linear algebra, in: Society for Industrial and Applied Mathematics, Philadelphia, (1997).
[12] Q. Wang, J. Lam, Y. Wei, T. Chen, Iterative solutions of coupled discrete Markovian jump Lyapunov equations, Comp. Math. Appl 55 (2008) 843-850.
[13] E. K. Boukas and H. Yang, Stability of discrete-time linear systems with Markovian jumping parameters, Math. Control, Signals, Syst 8 (1995) 390-402.
[14] B. Zhou, J. Lam, and G. R. Duan, Convergence of gradient-based iterative solution of coupled Markovian jump Lyapunov equation, Comp. Math. Appl 56 (2008) 3070-3078.
[15] H.M. Zhang , A finite iterative algorithm for solving the complex generalized coupled sylvester matrix equations by using the linear operators, J. Frankl. Inst 354 (4) (2017) 1856-1874.
[16] Z.L. Tian, J.X. Wang, Y.H. Dong, Z.Y. Liu, A multi-step Smith-inner-outer iteration algorithm for solving coupled continuous Markovian jump Lyapunov matrix equations, J. Frankl. Inst 357 (2020) 3656-3680.
[17] Z.L. Tian, C.M. Fan, Y.J. Deng, P.H. Wen, New explicit iteration algorithms for solving coupled continuous Markovian jump Lyapunov matrix equation, J. Frankl. Inst 355 (2018) 8346-8372.
[18] R.A. Smith, Matrix equation $XA + BX = C$, SIAM J. Appl. Math 16(1) (1968) 198-201.
[19] A.G. Wu, X. Wang, V. Sreeram, Iterative algorithms for solving continuous stochastic Lyapunov equations, IET Control Theor. Appl 11 (1) (2016) 73-80.
[20] H.J. Sun, Y. Zhang, Y.M. Fu, Accelerated smith iterative algorithms for coupled Lyapunov matrix equations, J. Frankl. Inst 354 (2017) 6877-6893.
[21] A.G.Wu, G.R. Duan, W. Liu, Implicit iterative algorithms for continuous markovian jump Lyapunov equations, IEEE Trans. Autom. Control 61 (10) (2015) 3183-3189.
[22] Y.Y. Qian , W.J. Pang , An implicit sequential algorithm for solving coupled Lyapunov equations of continuous-time Markovian jump systems, Automatica 60 (2015) 245-250.
[23] Z.L. Tian, C.Q. Gu, A numerical algorithm for Lyapunov equations, Appl. Math. Comput 202 (1) (2008) 44-53.

[24] H.M. Zhang , Reduced-rank gradient-based algorithms for generalized coupled sylvester matrix equations and its applications, Comp. Math. Appl 70 (8) (2015) 2049-2062.
[25] M.Sadkane, A low-rank Krylov squared Smith method for large-scale discrete-time Lyapunov equations, Linear Alg. Appl 436 (2012) 2807-2827.
[26] Z.L. Tian, M.Y. Tian, C.Q. Gu, X.N. Hao, An accelerated Jacobi-gradient based iterative algorithm for solving Sylvester matrix equations, Filomat 31 (2017) 2381-2390.
[27] Q.W. Wang, X.J. Xu, X.F. Duan, Least squares solution of the quaternion Sylvester tensor equation, Linear. Multilinear. A 69 (2021) 104-130.
[28] T. Li, Q.W. Wang, X.F. Duan, Numerical algorithms for solving discrete Lyapunov tensor equation, J. Comput. Appl. Math 370 (2020) 112676.
[29] Q.W. Wang, Z.H. He, Y. Zhang, Constrained two-sided coupled Sylvester-type quaternion matrix equations, Automatica 101 (2019) 207-213.
[30] X.J. Xu, Q.W. Wang, Extending BiCG and BiCR methods to solve the Stein tensor equation, Comput. Math. Appl 77 (2019) 3117-3127.
[31] Q.W. Wang, X.J. Xu, Iterative algorithms for solving some tensor equations, Linear. Multilinear. A 67 (2019) 1325-1349.
[32] Z.L. Tian, X.K. Li, T.Y. Xu, Z.Y. Liu, A relaxed MSIO iteration algorithm for solving coupled discrete Markovian jump Lyapunov equations, J. Frankl. Inst 358 (2021) 3051-3076.