Finding the LQR Weights to Ensure the Associated Riccati Equations Admit a Common Solution

Jianglin Lan, and Dezong Zhao, *Senior Member, IEEE*

***Abstract*—This paper addresses the problem of finding the linear quadratic regulator (LQR) weights such that the associated discrete-time algebraic Riccati equations admit a common optimal stabilising solution. Solving such a problem is key to designing LQR controllers to stabilise discrete-time switched linear systems under arbitrary switching, or stabilise polytopic systems (e.g., Takagi-Sugeno fuzzy systems and linear parameter varying systems) in the entire operating region. To ensure problem tractability and reduce the searching space, this paper proposes an efficient framework of finding only the state weights based on the given input weights. Under this framework, linear matrix inequality conditions are derived to conveniently check feasibility of the problem. An iterative algorithm with quadratic convergence and low computational complexity is developed to solve the problem. Efficacy of the proposed method is illustrated through numerical simulations of systems with various sizes.**

***Index Terms*—Inexact Kleinman-Newton method, linear matrix inequality, linear quadratic regulator, Riccati equation.**

I. Introduction

Many dynamical systems can be described by a family of linear time-invariant (LTI) subsystems with a logical rule that governs the switching between the subsystems. The critical examples are switched linear systems [1] and polytopic systems [2] such as Takagi-Sugeno (T-S) fuzzy systems and linear parameter varying (LPV) systems. Switched linear systems are useful for modelling physical systems that have discrete parameter changes [3], e.g., networked systems, systems integrating logic and power systems, or systems that have sudden failures in sensors or actuators [4]. Polytopic systems use a convex combination of the subsystems to cover the entire operating region of nonlinear systems [5], [6], e.g., wind turbines, spacecrafts, vibroacoustic systems and automotive systems. Stabilisation of switched linear systems under arbitrary switching, or polytopic systems in the entire operating region, has attracted great practical and theoretical interests.

Stabilisation of switched linear systems or polytopic systems can be realised respectively via switching control or gain-scheduled control, both of which are based on the common Lyapunov function (CLF) method. It is thus of fundamental importance to check the existence of the CLF and provide an efficient way to find it. Many works have studied the existence of CLF for special linear systems [7], [8] or linear copositive CLF for switched positive linear systems [9], but it remains an open question for more general systems. Existing methods for finding the CLF include randomised gradient iteration [10] and particle swarm optimisation [11], but they are only for Schur stable LTI systems. There is also an increasing interest in learning a CLF from system data [12]. Based on the CLF method, linear matrix inequality (LMI) technique has been widely adopted to synthesise switched or gain-scheduled controllers to ensure system stability [2], [5], [6], [13], [14], and robustness/fault-tolerance [4]. The most popular CLFs are quadratic functions which are known to be conservative. The conservativeness can be reduced by using parameter-dependent Lyapunov function [13] or switched Lyapunov function [15]–[17]. Some works have also studied the SOS-convex CLF [18] and neural network CLF [19]. However, control performance optimality is not considered in these CLF-based designs. Some works have leveraged the Riccati inequality (RI) method to find a CLF and obtain a linear quadratic regulator (LQR) controller with suboptimal performance [20]–[23].

This paper considers LQR design for discrete-time switched linear systems or polytopic systems such that the overall system stabilisation and optimal cost for each subsystem are both met. Each subsystem optimises its own cost using the LQR gain solved from a discrete-time algebraic Riccati equation (DARE) [24]. Different from the RI method which solves the inequality variants of DAREs [20]–[22], the present work seeks to solve the DAREs directly. If a common solution to the DAREs of all subsystems exists, the control performance of each subsystem is optimised and the overall system stabilisation is guaranteed under arbitrary switching or in the entire operating region. This motivates the present research of finding the common optimal stabilising (COS) solution to a set of DAREs. To the best of the authors' knowledge, such a problem has not been studied in the literature. This work contains the problem of finding the CLF as a special case and the results will be useful for designing model predictive control for piecewise linear systems [25] and observers for switched linear systems [26].

As for the CLF method, it is of fundamental importance to check the existence of the COS solution to a set of DAREs. It is known that a DARE has a unique optimal solution upon satisfaction of the stabilisability and detectability conditions [24]. Several efficient methods have been developed to solve DARE, including numerical methods such as the Schur-type method [27] and Newton-type methods [28]–[30], and optimisation methods such as LMI [31] and distributed optimisation [32]. If the common solution to a set of DAREs is known to exist, then it can be determined by simply solving any

Jianglin Lan was supported by a Leverhulme Trust Early Career Fellowship under Award ECF-2021-517. Dezong Zhao was supported by the Engineering and Physical Sciences Research Council of UK under the EPSRC-UKRI Innovation Fellowship scheme (EP/S001956/1).

J. Lan and D. Zhao are with the James Watt School of Engineering, University of Glasgow, Glasgow G12 8QQ, UK (emails: {Jianglin.Lan, Dezong.Zhao}@glasgow.ac.uk).

of the DAREs. However, it remains as an open question of choosing the LQR weights such that the set of DAREs admit a COS solution. This paper aims to develop an efficient strategy to address it and the main contributions are summarised as follows:

1) An efficient framework is proposed for determining the LQR weights to ensure the DAREs admit a COS solution, where only the state weights are to be found while the input weights are prescribed.
2) A quadratically convergent iterative algorithm is developed for finding the COS solution to the DAREs and the associated LQR weights.
3) LMI conditions are provided for examining feasibility of the iterative algorithm and generating an initial feasible solution to start the iteration.
4) The proposed algorithm is useful for synthesising LQR controllers to stabilise switched linear systems under arbitrary switching, and stabilise T-S fuzzy or LPV systems in the entire operating region.

In the rest of this paper, Section II describes the problem, Section III presents the proposed strategy, Section IV shows the numerical results, and Section V draws the conclusion.

II. Problem description

Consider a class of discrete-time systems described by

$$x(t+1) = A_{\delta(t)}x(t) + B_{\delta(t)}u(t) \tag{1}$$

where $x \in \mathbb{R}^n$ and $u \in \mathbb{R}^m$ are the vectors of state and control inputs, respectively. $\delta(t)$ is the signal that governs the switches between the N subsystems with constant system matrices $A_i \in \mathbb{R}^{n\times n}$ and $B_i \in \mathbb{R}^{n\times m}$, $i \in [1, N]$. It is assumed that all subsystems are stabilisable and $x(t)$ is measurable. This class of systems can represent switched linear systems or polytopic systems such as T-S fuzzy or LPV systems.

Switched linear systems. In this case, $\delta(t) = \{\delta_i(t)\}_{i=1}^N$ is a time-dependent switching sequence indicating which subsystem is active at time step t. This paper focuses on designing a switched LQR controller $u(t) = -K_{\delta(t)}x(t)$ to ensure stability of system (1) under arbitrary switching. The gain $K_i \in \mathbb{R}^{m\times n}$ for each subsystem i, $i \in [1, N]$, is designed to minimise the infinite-horizon cost function $J_i = \sum_{t=0}^{\infty}[x(t)^\top \bar{Q}_i x(t) + u(t)^\top R_i u(t)]$, where the weights $\bar{Q}_i \in \mathbb{R}^{n\times n}$ and $R_i \in \mathbb{R}^{m\times m}$ are symmetric positive semi-definite (s.p.s.d.) and symmetric positive definite (s.p.d.) matrices, respectively. The design problem is as follows:

Problem 2.1: For the switched linear system (1), design the switched LQR controller $u(t) = -K_{\delta(t)}x(t)$ with the gains

$$K_i = (R_i + B_i^\top P B_i)^{-1} B_i^\top P A_i, \ i \in [1, N] \tag{2}$$

where the s.p.s.d. matrix P is the COS solution to the DAREs:

$$A_i^\top P A_i - P + \bar{Q}_i - A_i^\top P B_i(R_i + B_i^\top P B_i)^{-1} B_i^\top P A_i = 0, \\ i \in [1, N] \tag{3}$$

with the s.p.s.d. matrix $\bar{Q}_i$ and s.p.d. matrix R_i.

Polytopic systems. In this case, system (1) represents a T-S fuzzy system or a LPV system, where $\delta(t) = [\delta_1(t), \ldots, \delta_N(t)]$ is the vector of coefficients satisfying $0 \le \delta_i(t) \le 1$, $i \in [1, N]$, and $\sum_{i=1}^N \delta_i(t) = 1$. A convex combination of all the subsystems with $A_{\delta(t)} = \sum_{i=1}^N \delta_i(t)A_i$ and $B_{\delta(t)} = \sum_{i=1}^N \delta_i(t)B_i$ is used to capture the system dynamics across the operating space. This paper considers designing a gain-scheduled LQR controller $u(t) = -\sum_{i=1}^N \delta_i(t)K_i x(t)$ to stabilise the overall system (1) whilst minimising the cost functions J_i, $i \in [1, N]$. The design problem is as follows:

Problem 2.2: For the polytopic system (1), design a gain-scheduled LQR controller $u(t) = -\sum_{i=1}^N \delta_i(t)K_i x(t)$ with the gains $\{K_i\}_{i=1}^N$ in the form of (2), where P is the COS solution to (3) and also satisfies

$$(A_i - B_iK_j)^\top P(A_i - B_iK_j) < P, \ \forall i \ne j, \ i, j \in [1, N]. \tag{4}$$

Problem 2.2 includes an extra condition (4) to ensure that the gain-scheduled LQR controller stabilises the system not only at each operating point (or vertex) but also at anywhere in the entire operating region [2], [5], [6], [13], [14].

For both Problems 2.1 and 2.2, the common solution P is said to be *stabilising* if for any $i \in [1, N]$, the obtained LQR gain K_i makes $A_i - B_iK_i$ Schur stable. The solution is also *optimal* if for any $i \in [1, N]$, the controller $u(t) = -K_i x(t)$ minimises the cost function J_i. The quadratic function $V(t) = x(t)^\top P x(t)$ is a CLF for all the subsystems of (1). The existence of P guarantees asymptotic stability of the switched linear systems or the polytopic systems if P also satisfies (4). Therefore, solving the DAREs in (3) is key to the LQR controller designs in both Problems 2.1 and 2.2.

It is well-known that the solution to the i-th DARE in (3) is unique when the pair (A_i, B_i) is stabilisable and $(A_i, \sqrt{\bar{Q}_i})$ is detectable. Hence, if the COS solution to the DAREs in (3) is known to exist for the given sets of weights $\{\bar{Q}_i\}_{i=1}^N$ and $\{R_i\}_{i=1}^N$, then it can be directly solved from any of the DAREs. The main challenge is how to find the weights such that a common solution exists, especially for large-scale systems. To tackle the challenge, this paper is dedicated to developing a strategy to solve Problem 2.3.

Problem 2.3: Find the sets of LQR weights $\{\bar{Q}_i\}_{i=1}^N$ and $\{R_i\}_{i=1}^N$ such that the DAREs in (3) admit a COS solution P.

A strategy is derived in Section III for solving Problem 2.3, and subsequently Problems 2.1 and 2.2, to obtain the LQR gains $\{K_i\}_{i=1}^N$. For the sake of conciseness and clarity, the strategy will be illustrated based on switched linear systems, while its adaptation to polytopic systems (i.e., T-S fuzzy or LPV systems) will be discussed in Remarks 3.2, 3.3 and 3.4.

III. Main Results

In this section, the following operators are defined for any s.p.s.d. matrices $X, Q_i \in \mathbb{R}^{n\times n}$ and s.p.d. matrix $R_i \in \mathbb{R}^{m\times m}$:

$$\begin{aligned}
&\mathcal{K}_i(X) := (R_i + B_i^\top X B_i)^{-1} B_i^\top X A_i, \\
&\mathcal{A}_i(X) := A_i - B_i\mathcal{K}_i(X), \\
&\mathcal{G}_i(X) := B_i(R_i + B_i^\top X B_i)^{-1} B_i^\top, \\
&\mathcal{F}_i(X) := A_i^\top X A_i - X + Q_i - A_i^\top X \mathcal{G}_i(X) X A_i, \\
&\mathcal{L}_i(X) := \mathcal{A}_i(X)^\top X \mathcal{A}_i(X) - X + \mathcal{K}_i(X)^\top R_i \mathcal{K}_i(X) + Q_i.
\end{aligned} \tag{5}$$

It can be verified that $\mathcal{F}_i(X) = \mathcal{L}_i(X)$.

A. Solvability of Problem 2.3 and its Checking Condition

In Problem 2.3, two sets of weights $\{\bar{Q}_i\}_{i=1}^N$ and $\{R_i\}_{i=1}^N$ need to be found. They can be found naively via a heuristic procedure: (i) Select the candidate weights $\{\bar{Q}_i\}_{i=1}^N$ and $\{R_i\}_{i=1}^N$; (ii) Solve the N DAREs for a solution set $\{P_i\}_{i=1}^N$; (iii) If $P_1 = \cdots = P_N$, a common solution is found, otherwise starting over from (i). This heuristics may work for special systems such as small-scale systems or systems whose matrices $\{A_i\}_{i=1}^N$ and $\{B_i\}_{i=1}^N$ have special structures (e.g., diagonal, upper or lower triangular structures). However, the heuristics has no convergence guarantee, making it computationally expensive and even intractable for large-scale systems. If fixing one of the two weights sets to find the other set, then the searching space (i.e., computational complexity) can be greatly reduced. This inspires the proposal of Theorem 3.1.

Theorem 3.1: Choosing the s.p.s.d. matrix Q_i making $(A_i, \sqrt{Q_i})$ detectable and s.p.d. matrix R_i, $i \in [1, N]$. If the s.p.s.d. matrix P is the common stabilising (CS) solution to

$$\mathcal{F}_i(P) = \Upsilon_i,\ i \in [1, N] \tag{6}$$

with the symmetric matrix $\Upsilon_i \leq \alpha Q_i$ and $\alpha \in [0, 1)$, then P is the COS solution to the DAREs in (3) and the obtained weights are $\{\bar{Q}_i\}_{i=1}^N$ and $\{R_i\}_{i=1}^N$, where $\bar{Q}_i = Q_i - \Upsilon_i$. For any $i \in [1, N]$, if choosing $R_i = \varphi_i Q_i$, the obtained weights satisfy $R_i \geq (\varphi_i/(1-\alpha))\bar{Q}_i$. Hence, the control performance is tunable via choosing α and φ_i.

Proof: It follows from (5) that (6) is equivalent to

$$A_i^\top X A_i - X + \bar{Q}_i - A_i^\top X \mathcal{G}_i(X) X A_i = 0,\ i \in [1, N] \tag{7}$$

where $\bar{Q}_i = Q_i - \Upsilon_i$. Since $\Upsilon_i \leq \alpha Q_i$ and $\alpha \in [0, 1)$, it is true that $\bar{Q}_i = \alpha_i^* Q_i$ with $0 < \alpha_i^* \leq 1 - \alpha \leq 1$, $i \in [1, N]$. Since Q_i is s.p.s.d. and $(A_i, \sqrt{Q_i})$ is detectable, the obtained matrix $\bar{Q}_i$ is s.p.s.d. and $(A_i, \sqrt{\bar{Q}_i})$ is detectable. Therefore, (7) has the same form as (3) with the known sets of weights $\{\bar{Q}_i\}_{i=1}^N$ and $\{R_i\}_{i=1}^N$. By choosing $R_i = \varphi_i Q_i$, then it can be derived that $R_i = (\varphi_i/\alpha^*)\bar{Q}_i \geq (\varphi_i/(1-\alpha))\bar{Q}_i$. ■

Theorem 3.1 shows that Problem 2.3 is solved by finding the CS solution P to (6) for the given weights $\{Q_i\}_{i=1}^N$ and $\{R_i\}_{i=1}^N$. This lays the theoretic foundation for the solving strategy proposed in Section III-B. As a prerequisite, it is necessary to check the solvability of Problem 2.3, i.e., examining whether there is a CS solution P to (6) for the given $\{Q_i\}_{i=1}^N$ and $\{R_i\}_{i=1}^N$. The checking condition is given as the LMIs in Corollary 3.1. Solving the LMIs will also produce an initial feasible CS solution to start the proposed iterative algorithm.

Corollary 3.1: Choosing the s.p.s.d. matrix Q_i making $(A_i, \sqrt{Q_i})$ detectable and s.p.d. matrix R_i, $i \in [1, N]$. Problem 2.3 is solvable if there is a s.p.d. matrix Z and matrices Y_i, $i \in [1, N]$, satisfying

$$\begin{bmatrix} -Z & \star & \star & \star \\ A_i Z - B_i Y_i & -Z & \star & \star \\ \sqrt{Q_i} Z & 0 & -I & \star \\ \sqrt{R_i} Y_i & 0 & 0 & -I \end{bmatrix} \leq 0,\ i \in [1, N] \tag{8}$$

where $\star$ indicates symmetry. The CS solution is $P = Z^{-1}$.

Proof: By using Schur complement [31], the LMIs in (8) are equivalently reformulated as the Riccati inequalities $\mathcal{L}_i(P) \leq 0$, $i \in [1, N]$, by setting $P = Z^{-1}$ and $K_i = Y_i Z^{-1} = (R_i + B_i^\top P B_i)^{-1} B_i^\top P A_i$. The obtained matrices $A_i - B_i K_i$, $i \in [1, N]$, are Schur stable. As defined in (5), $\mathcal{F}_i(P) = \mathcal{L}_i(P)$. Hence, the equations in (6) always hold with the s.p.s.d. matrices Υ_i satisfying $\Upsilon_i < \alpha Q_i$, $i \in [1, N]$, where $\alpha \in [0, 1)$. Therefore, P is the CS solution to the equations in (6) and Problem 2.3 is solvable for the given sets of weights $\{Q_i\}_{i=1}^N$ and $\{R_i\}_{i=1}^N$. ■

Remark 3.1: Corollary 3.1 shows that Problem 2.3 could be solved using the Riccati inequality (RI) method [23], which aims to find a CS solution P to satisfy the LMIs in (8) and minimise the upper bound of the LQR cost $x(0)^\top P x(0)$. This is formulated as the optimisation problem:

$$\begin{aligned} &\min\ \operatorname{trace}(V) \\ &\text{subject to: (8)},\ \begin{bmatrix} Z & I \\ I & V \end{bmatrix} \geq 0,\ Z = Z^\top > 0,\ V > 0. \end{aligned} \tag{9}$$

Solving (9) gives the CS solution $P = Z^{-1}$. Then Problem 2.3 is solved with $\{\bar{Q}_i\}_{i=1}^N$ and $\{R_i\}_{i=1}^N$, where $\bar{Q}_i = Q_i - \mathcal{F}_i(P)$.

The computational complexity of solving the optimisation problem (9) can be characterised by [33]: $\mathcal{O}_{\text{RI}} = \mathcal{R}_{\text{RI}} \mathcal{S}_{\text{RI}}^3$, where $\mathcal{R}_{\text{RI}} = (3N + 6)n + Nm$ is the LMI row size and $\mathcal{S}_{\text{RI}} = 1.5n^2 + (0.5 + Nm)n$ is the total number of scalar decision variables. As a function of n^7 and N^4, $\mathcal{O}_{\text{RI}}$ grows dramatically with the subsystem dimension n and number of subsystems N. This issue is computationally critical so that solving (9) with standard interior point methods may be very difficult or even intractable. Therefore, this paper will develop a more efficient strategy for solving Problem 2.3.

Remark 3.2: In terms of T-S fuzzy or LPV systems, the matrix Y_i in the LMI (8) in Corollary 3.1 is replaced by Y_j. Then the LMI needs to be solved for all $i, j \in [1, N]$. The same treatment applies to the RI method in Remark 3.1.

B. Outline of the Proposed Iterative Strategy

It is well-known from the LQR theory [24] that the optimal LQR cost for each subsystem in (1) is $x(0)^\top P x(0)$. It is thus desirable to find a CS solution P that is as small as possible to minimise the LQR cost. This motivates the presented iterative strategy for solving Problem 2.3 based on the results in Theorem 3.1. The key idea is to iteratively solve the DAREs:

$$\mathcal{F}_i(P) = 0,\ i \in [1, N] \tag{10}$$

with the given weights $\{Q_i\}_{i=1}^N$ and $\{R_i\}_{i=1}^N$, up to a prescribed accuracy. When the iteration is terminated, the DAREs in (10) are solved with a minimum P and the residuals Υ_i, i.e., $\mathcal{F}_i(P) = \Upsilon_i$, $i \in [1, N]$. Problem 2.3 can then be solved by using the results in Theorem 3.1 directly.

The iterative strategy is developed based on the inexact Kleinman-Newton method [30]. The first Fréchet derivative of $\mathcal{F}_i(P)$ at P is defined as

$$\mathcal{F}_i'|_P(S) = \mathcal{A}_i(P)^\top S \mathcal{A}_i(P) - S,\ \forall S \in \mathbb{R}^{n \times n}. \tag{11}$$

By leveraging (11), the following Newton system is defined:

$$\mathcal{F}_i'|_{P_{i,k}}(P_{i,k+1} - P_{i,k}) + \mathcal{F}_i(P_{i,k}) = 0 \tag{12}$$

where $P_{i,k}$ and $P_{i,k+1}$ are the iterates at steps k and $k+1$.

Algorithm 1 The squared Smith method

$\bar{P}_{i,0} \leftarrow \mathcal{K}_i(P_k)^\top R_i \mathcal{K}_i(P_k) + Q_i$.
$\mathcal{R}_i^0 \leftarrow \mathcal{A}_i(P_k)^\top \bar{P}_{i,0} \mathcal{A}_i(P_k)$.
for $\mu = 1, 2, \dots$ **do**
  $\bar{P}_{i,\mu} \leftarrow (\mathcal{A}_i(P_k)^\top)^{2^{\mu-1}} \bar{P}_{i,\mu-1} \mathcal{A}_i(P_k)^{2^{\mu-1}} + \bar{P}_{i,\mu-1}$.
  $\mathcal{R}_i^\mu \leftarrow \mathcal{A}_i(P_k)^\top P_{i,\mu} \mathcal{A}_i(P_k) - P_{i,\mu} + \bar{P}_{i,0}$.
  If $\|\mathcal{R}_i^\mu - \mathcal{R}_i^{\mu-1}\| \leq \eta_{\text{in}} \|\mathcal{R}_i^{\mu-1}\|$, then stop.
end for

Define $\mathcal{S}_k := \{P_{i,k}\}_{i=1}^N$ as the set of iterates at step k for all the DAREs in (10) and P_k as the common solution satisfying $P_k \in \mathcal{S}_k$. To ensure all the DAREs generate the next iterate from the current common iterate, replacing $P_{i,k}$ in (12) by P_k. Then (12) is rewritten as the Stein equation

$$P_{i,k+1} - \mathcal{A}_i(P_k)^\top P_{i,k+1} \mathcal{A}_i(P_k) = \mathcal{K}_i(P_k)^\top R_i \mathcal{K}_i(P_k) + Q_i. \tag{13}$$

The Stein equation (13) is solved using the squared Smith method [34] described in Algorithm 1. It is shown in [34] that $\bar{P}_{i,\mu}$ approaches its exact value very rapidly as μ goes to ∞. Hence, given a tolerance η_{in}, Algorithm 1 will terminate at a finite μ^* and return the iterate $P_{i,k+1} = \bar{P}_{i,\mu^*}$ with the residual $\mathcal{R}_{i,k+1} = \mathcal{A}_i(P_k)^\top P_{i,\mu^*} \mathcal{A}_i(P_k) - P_{i,\mu^*} + \bar{P}_{i,0}$.

Based on the above analysis, the proposed iterative strategy is outlined in Algorithm 2. The common iterate P_{k+1} needs to satisfy the conditions (17a) - (17c). It will be shown in Section III-C that these conditions are necessary to establish the existence and convergence of the CS solution sequence $\{P_k\}_{k=1}^\infty$. Algorithm 2 omits the details of choosing the iterate P_{k+1} and setting the stopping criteria, which can simplify the analysis but will not affect the existence and convergence proofs in Section III-C. A practically implementable algorithm with all the necessary details will be provided in Section III-D.

Remark 3.3: In terms of T-S fuzzy or LPV systems, when constructing the common solution set $\mathcal{S}_{k+1}^{\text{c}}$ (see Line 6, Algorithm 2), the following condition is added to (17): $\mathcal{A}_{i,s}(P_{j,k+1}^{\text{c}})^\top P_{j,k+1}^{\text{c}} \mathcal{A}_{i,s}(P_{j,k+1}^{\text{c}}) < P_{j,k+1}^{\text{c}},\ \forall i \neq s,\ i, s \in [1, N]$, where $\mathcal{A}_{i,s}(P_{j,k+1}^{\text{c}}) = A_i - B_i \mathcal{K}_s(P_{j,k+1}^{\text{c}})$.

Algorithm 2 Outline of the proposed strategy

1: **Input:** System matrices $\{A_i\}_{i=1}^N$ and $\{B_i\}_{i=1}^N$. The sets of s.p.s.d. matrices $\{Q_i\}_{i=1}^N$ and s.p.d. matrices $\{R_i\}_{i=1}^N$ such that $(A_i, \sqrt{Q}_i)$, $i \in [1, N]$, are detectable and the LMIs in (8) are feasible. A scalar $\alpha \in [0, 1)$ and small positive constants η_{in} and η_{out}.
2: **Initialise:** Solve Z from (8) and set $P_0 \leftarrow Z^{-1}$.
3: **for** $k = 0, 1, \dots$ **do**
4:   Construct the sets $\mathcal{S}_{k+1} := \{P_{i,k+1}\}_{i=1}^N$ and $\mathcal{R}_{k+1} := \{\mathcal{R}_{i,k+1}\}_{i=1}^N$ by running Algorithm 1 for all $i \in [1, N]$.
5:   Construct the common solution set $\mathcal{S}_{k+1}^{\text{c}} := \{P_{j,k+1}^{\text{c}}\}_{j=1}^\ell$, where $P_{j,k+1}^{\text{c}} \in \mathcal{S}_{k+1}$ and satisfies

$$\mathcal{K}_j(P_k)^\top R_j \mathcal{K}_j(P_k) + Q_j - \mathcal{R}_{j,k+1} \geq 0 \tag{17a}$$
$$W_{i,k+1} + \alpha Q_i - \mathcal{R}_{i,k+1} \geq 0,\ \forall i \in [1, N] \tag{17b}$$
$$\mathcal{R}_{j,k+1} \leq W_{j,k+1} \tag{17c}$$

  with $W_{i,k+1} = \mathcal{A}_i(P_k)^\top \Delta P_{i,k} \mathcal{G}_i(P_{i,k+1}^{\text{c}}) \Delta P_{i,k} \mathcal{A}_i(P_k)$ and $\Delta P_{i,k} = P_{i,k+1}^{\text{c}} - P_k$.
6:   Set the common iterate P_{k+1} as any element of $\mathcal{S}_{k+1}^{\text{c}}$.
7:   If $\|P_{k+1} - P_k\| \leq \eta_{\text{out}} \|P_k\|$, then stop.
8: **end for**
9: **Output:** Common solution P_{k+1}.

C. Properties of the Proposed Strategy

This section analyses the properties of Algorithm 2, including the existence and convergence of the common iterate sequence $\{P_k\}_{k=1}^\infty$. To facilitate the analysis, rewriting the Stein equation (13) solved in Algorithm 1 as the Stein equation

$$\begin{aligned} &P_{i,k+1} - \mathcal{A}_i(P_k)^\top P_{i,k+1} \mathcal{A}_i(P_k) \\ &= \mathcal{K}_i(P_k)^\top R_i \mathcal{K}_i(P_k) + Q_i - \mathcal{R}_{i,k+1}. \end{aligned} \tag{14}$$

This Stein equation is used in [30] for a single DARE. Based on Lemma 2.3 in [30], it can be proved that the following two equations hold under the Stein equation (14):

$$\begin{aligned} &P_{k+1} - \mathcal{A}_i(P_{k+1})^\top P_{k+1} \mathcal{A}_i(P_{k+1}) \\ =&W_{i,k+1} + \mathcal{K}_i(P_{k+1})^\top R_i \mathcal{K}_i(P_{k+1}) + Q_i - \mathcal{R}_{i,k+1}, \end{aligned} \tag{15}$$

$$\begin{aligned} &(P_{k+1} - P_{i,k+2}) - \mathcal{A}_i(P_{k+1})^\top (P_{k+1} - P_{i,k+2}) \mathcal{A}_i(P_{k+1}) \\ =&W_{i,k+1} - \mathcal{R}_{i,k+1} + \mathcal{R}_{i,k+2}, \end{aligned} \tag{16}$$

with $W_{i,k+1} = \mathcal{A}_i(P_k)^\top \Delta P_k \mathcal{G}_i(P_{k+1}) \Delta P_k \mathcal{A}_i(P_k) = \Delta \mathcal{K}_{i,k}^\top (R_i + B_i^\top P_{k+1} B_i) \Delta \mathcal{K}_{i,k}$, where $\Delta P_k = P_{k+1} - P_k$ and $\Delta \mathcal{K}_{i,k} = \mathcal{K}_i(P_{k+1}) - \mathcal{K}_i(P_k)$.

Existence of the CS solution P_{k+1} in Algorithm 2 is shown in Lemma 3.1.

Lemma 3.1: Given the sets of s.p.s.d. matrices $\{Q_i\}_{i=1}^N$ and s.p.d. matrices $\{R_i\}_{i=1}^N$. Let P_k be a s.p.s.d. matrix such that the matrices $\mathcal{A}_i(P_k)$, $i \in [1, N]$, are Schur stable. Then Algorithm 2 has the properties:

(i) The iterate P_{k+1} is well-defined and s.p.s.d..
(ii) The matrices $\mathcal{A}_i(P_{k+1})$, $i \in [1, N]$, are Schur stable.

Proof: Note that $P_{k+1} \in \mathcal{S}_{k+1}^{\text{c}} \subseteq \mathcal{S}_{k+1}$. Without loss of generality, assuming that $P_{k+1} = P_{j,k+1}$, where $1 \leq j \leq \ell \leq N$, which is a solution to the Stein equation (14). Since $\mathcal{A}_j(P_k)$ is Schur stable, $P_{j,k+1}$ can be represented as [35]: $P_{j,k+1} = \sum_{r=0}^\infty (\mathcal{A}_j(P_k)^\top)^r Z_{j,k} \mathcal{A}_j(P_k)^r$ with $Z_{j,k} = \mathcal{K}_j(P_k)^\top R_j \mathcal{K}_j(P_k) + Q_j - \mathcal{R}_{j,k+1}$. Under (17a), $Z_{j,k}$ is s.p.s.d. and so is $P_{j,k+1}$. Hence, P_{k+1} is well-defined and s.p.s.d..

The property (ii) is proved by contradiction. Assuming that $\mathcal{A}_i(P_{k+1})$, $\forall i \in [1, N]$, is not Schur stable, then the relation $\mathcal{A}_i(P_{k+1}) x = \lambda x$ holds with $|\lambda| \geq 1$ and $x \neq 0$. Hence, the left-hand side of (15) satisfies

$$\begin{aligned} &x^\top [P_{k+1} - \mathcal{A}_i(P_{k+1})^\top P_{k+1} \mathcal{A}_i(P_{k+1})] x \\ =&(1 - |\lambda|^2) x^\top P_{k+1} x \leq 0. \end{aligned} \tag{18}$$

Denote the right-hand side of (15) as $\hat{W}_{i,k} = W_{i,k+1} + \mathcal{K}_i(P_{k+1})^\top R_i \mathcal{K}_i(P_{k+1}) + Q_i - \mathcal{R}_{i,k+1}$. Since $P_{k+1} \in \mathcal{S}_{k+1}^{\text{c}}$, (17b) is satisfied and the term $W_{i,k+1} + \alpha Q_i - \mathcal{R}_{i,k+1}$ is s.p.s.d.. Hence, $\hat{W}_{i,k} \geq 0$ because both Q_i and $\mathcal{K}_i(P_{k+1})^\top R_i \mathcal{K}_i(P_{k+1})$ are s.p.s.d.. Subsequently, it is true that for $x \neq 0$, $x^\top \hat{W}_{i,k} x \geq 0$. It follows from (15) and (18) that $\hat{W}_{i,k} = 0$. This implies that the terms $W_{i,k+1} +$

$\alpha Q_i - \mathcal{R}_{i,k+1}$ and $(1-\alpha)Q_i + \mathcal{K}_i(P_{k+1})^\top R_i \mathcal{K}_i(P_{k+1})$ are both zero. However, since Q_i is s.p.s.d. and R_i is s.p.d., $(1-\alpha)Q_i + \mathcal{K}_i(P_{k+1})^\top R_i \mathcal{K}_i(P_{k+1}) \neq 0$. This leads to a contradiction and thus the matrices $\mathcal{A}_i(P_{k+1})$, $\forall i \in [1, N]$, are Schur stable. ■

By using Lemma 3.1, Theorem 3.2 summarises the properties of Algorithm 2.

Theorem 3.2: Given the sets of s.p.s.d. matrices $\{Q_i\}_{i=1}^N$ and s.p.d. matrices $\{R_i\}_{i=1}^N$. If P_0 is a s.p.s.d. matrix stabilising $\mathcal{A}_i(P_0)$, $i \in [1, N]$, then Algorithm 2 has the properties:

(i) The common solution sequence $\{P_k\}_{k=1}^\infty$ is well-defined and s.p.s.d.. The matrices $\mathcal{A}_i(P_k)$, $i \in [1, N]$, are Schur stable for any $k \geq 1$.

(ii) The sequence $\{P_k\}_{k=1}^\infty$ is nonincreasing and bounded below by the zero matrix. Moreover, it has quadratic convergence, i.e., there is a positive scalar δ such that

$$\|P_{k+2} - P_\infty\| \leq \delta \|P_{k+1} - P_\infty\|^2, \ k \in \mathbb{Z}. \quad (19)$$

Proof: Since P_0 is solved from (8), it is a s.p.s.d. matrix stabilising $\mathcal{A}_i(P_0)$, $i \in [1, N]$. Therefore, following Lemma 3.1, the properties in (i) can be proved simply by induction.

To prove the first part in (ii), without loss of generality, let the common iterate obtained at iteration $k+1$ be $P_{k+2} = P_{j,k+2}$, where $1 \leq j \leq \ell \leq N$. Since $\mathcal{A}_j(P_{k+1})$ is Schur stable, the equation (16) has the exact solution [35]:

$$P_{k+1} - P_{k+2} = \sum_{r=0}^{\infty} (\mathcal{A}_j(P_{k+1})^\top)^r \hat{Z}_{j,k+1} (\mathcal{A}_j(P_{k+1}))^r \quad (20)$$

where $\hat{Z}_{j,k+1} = W_{j,k+1} - \mathcal{R}_{j,k+1} + \mathcal{R}_{j,k+2}$. Under the condition (17c), one has $\mathcal{R}_{j,k+1} \leq W_{j,k+1}$ and thus $\hat{Z}_{j,k+1} \geq 0$. Applying this to (20) gives $0 \leq P_{k+2} \leq P_{k+1}$. By induction, it is true that $0 \leq P_{k+2} \leq P_{k+1}, \forall k \geq 1$, and the limit $\lim_{k\to\infty} P_k = P_\infty$ exists. Hence, the sequence $\{P_k\}_{k=1}^\infty$ is nonincreasing and bounded below by the zero matrix.

To prove the second part in (ii), let the common iterate obtained at iteration k be $P_{k+1} = P_{l,k+1}$, where $1 \leq l \leq \ell \leq N$. The following equation is derived from (20):

$$P_{k+2} - P_{k+1} = \sum_{r=0}^{\infty} (\mathcal{A}_l(P_{k+1})^\top)^r \tilde{Z}_{l,k+1} (\mathcal{A}_l(P_{k+1}))^r \quad (21)$$

where $\tilde{Z}_{l,k+1} = \mathcal{R}_{l,k+1} - \mathcal{R}_{l,k+2} - W_{l,k+1}$.

It follows from (15) and (16) that $W_{l,k+1}$ satisfies

$$\begin{aligned} W_{l,k+1} &= \Delta\mathcal{K}_{l,k}^\top (R_l + B_l^\top P_{k+1} B_l) \Delta\mathcal{K}_{l,k} \\ &= \mathcal{A}_l(P_k)^\top \Delta P_k \mathcal{G}_l(P_{k+1}) \Delta P_k \mathcal{A}_l(P_k), \end{aligned}$$

where $\Delta\mathcal{K}_{l,k} = \mathcal{K}_l(P_{k+1}) - \mathcal{K}_l(P_k)$ and $\Delta P_k = P_{k+1} - P_k$. Since $R_l + B_l^\top P_{k+1} B_l > 0$, it is true that $W_{l,k+1} \geq 0$. Also, it will be shown in Lemma 3.2 that $\mathcal{R}_{l,k+2} \geq 0$. Hence, $\tilde{Z}_{l,k+1} \leq \mathcal{R}_{l,k+1}$. Therefore, under the condition (17c) with $P_{j,k} = P_k$ and $P_{j,k+1} = P_{k+1}$, the following relations hold:

$$\|\tilde{Z}_{l,k}\| \leq \|\mathcal{R}_{l,k+1}\| \leq \|W_{l,k+1}\| \leq \epsilon_k \|P_{k+1} - P_k\|^2 \quad (22)$$

with a positive finite scalar $\epsilon_k = \|\mathcal{A}_l(P_k)\|^2 \|\mathcal{G}_l(P_{k+1})\|$.

Applying (22) to (21) yields

$$\|P_{k+2} - P_{k+1}\| \leq \beta_{k+1} \|P_{k+1} - P_k\|^2, \ k \geq 1 \quad (23)$$

with $\beta_{k+1} = \sum_{r=0}^\infty \epsilon_k \|\mathcal{A}_l(P_{k+1})^r\|^2$.

Define a positive scalar $\beta := \max_{k\geq 1}\{\beta_{k+1}\}$. By using (23), the following inequality holds for all $s \geq 1$:

$$\|P_{k+1+s} - P_{k+s}\| \leq \beta^{s-1} \|P_{k+2} - P_{k+1}\|^{2s-2}. \quad (24)$$

If $\|P_{k+2} - P_{k+1}\| = 0$, then by using (24), the equation $\|P_{k+1+s} - P_{k+s}\| = 0$ holds for all $s \geq 1$.

Since $\{P_k\}_{k=1}^\infty$ is nonincreasing, one has $P_{k+s} - P_{k+s+1} \geq 0$, $\forall k, s \geq 1$. Based on (23) and (24), for all $k^* \geq k+2$,

$$\begin{aligned} \|P_{k+1} - P_{k^*}\|^2 &\geq \sum_{s=1}^{k^*-k-1} \|P_{k+1+s} - P_{k+s}\|^2 \\ &= \delta_1 \|P_{k+2} - P_{k+1}\|^2 \end{aligned} \quad (25)$$

with a scalar $\delta_1 = \sum_{s=1}^{k^*-k-1} \frac{\|P_{k+1+s} - P_{k+s}\|^2}{\|P_{k+2} - P_{k+1}\|^2}$ satisfying $1 \leq \delta_1 \leq \sum_{s=1}^{k^*-k-1} \beta^{2s-2} \|P_{k+2} - P_{k+1}\|^{4s-6} < \infty$.

Similarly, by applying (23) and (24), for all $k^* \geq k+3$,

$$\begin{aligned} \|P_{k+2} - P_{k^*}\| &\leq \sum_{s=1}^{k^*-k-2} \|P_{k+2+s} - P_{k+1+s}\| \\ &\leq \delta_2 \|P_{k+2} - P_{k+1}\|^2 \end{aligned} \quad (26)$$

with a scalar $\delta_2 = \sum_{s=1}^{k^*-k-2} \frac{\|P_{k+2+s} - P_{k+1+s}\|}{\|P_{k+2} - P_{k+1}\|^2}$ satisfying $\beta \leq \delta_2 \leq \sum_{s=1}^{k^*-k-2} \beta^s \|P_{k+2} - P_{k+1}\|^{2s-2} < \infty$.

The above proof for establishing (25) and (26) requires $\|P_{k+2} - P_{k+1}\| \neq 0$. When $\|P_{k+2} - P_{k+1}\| = 0$, $P_{k^*} = P_{k+2} = P_{k+1}$, $\forall k^* \geq k+2$. Hence, (25) and (26) automatically hold. This means that (25) and (26) always hold.

Combining (25) and (26) gives

$$\|P_{k+2} - P_{k^*}\| \leq \delta \|P_{k+1} - P_{k^*}\|^2, \ \forall k^* \geq k+3 \quad (27)$$

with $\delta = \delta_2/\delta_1 > 0$. Passing (27) to the limit yields (19). ■

Based on the results in Theorem 3.2, it is shown below that the proposed iterative strategy solves Problem 2.3: By using (15) and the operators $\mathcal{F}_i(X)$ and $\mathcal{L}_i(X)$ defined in (5), it can be derived that $\mathcal{F}_i(P_{k+1}) = \Upsilon_i$, $i \in [1, N]$, where $\Upsilon_i = \mathcal{R}_{i,k+1} - W_{i,k+1}$. Under the condition (17c), $\Upsilon_i \leq 0$. It is indicated by (14) that $\mathcal{R}_{i,k+1}$ is a symmetric matrix. Also, $W_{i,k+1}$ is a s.p.s.d. matrix as defined in Algorithm 2. Hence, Υ_i is symmetric. Moreover, the iterate P_{k+1} obtained from Algorithm 2 satisfies (17b), indicating that $\Upsilon_i \leq \alpha Q_i$, $i \in [1, N]$, where $\alpha \in [0, 1)$. Therefore, Problem 2.3 is solved and the sequence $\{P_k\}_{k=1}^\infty$ generated by Algorithm 2 converges quadratically to the COS solution P_∞.

D. Implementation Details of the Proposed Strategy

In Algorithm 2, the conditions (17a) - (17c) are necessary for establishing the properties claimed in Section III-C. The conditions (17b) and (17c) need to be verified at each iteration. The condition (17a) automatically holds by using Algorithm 1 to solve the Stein equation (13), as shown in Lemma 3.2.

Lemma 3.2: Using Algorithm 1 to solve the Stein equation (13) ensures $0 \leq \mathcal{R}_{i,k+1} \leq \mathcal{K}_i(P_k)^\top R_i \mathcal{K}_i(P_k) + Q_i$.

Proof: Denote $S_{i,k} = \mathcal{K}_i(P_k)^\top R_i \mathcal{K}_i(P_k) + Q_i$. Since $\mathcal{A}_i(P_k)$ is Schur stable, the exact solution to (13) is given as

$$P_{i,k+1}^* = \sum_{r=0}^{\infty} (\mathcal{A}_i(P_k)^\top)^r S_{i,k} (\mathcal{A}_i(P_k))^r. \quad (28)$$

Let $\{\bar{P}_{i,\mu}\}_{\mu=0}^{\infty}$ be the sequence generated iteratively by

$$\bar{P}_{i,\mu} = \bar{P}_{i,\mu-1} + (\mathcal{A}_i(P_k)^\top)^{2^\mu} \bar{P}_{i,\mu-1}(\mathcal{A}_i(P_k))^{2^\mu} \tag{29}$$

with $\bar{P}_{i,0} = S_{i,k}$. At step μ, the residual is given by

$$\mathcal{R}_{i,\mu} = \mathcal{A}_i(P_k)^\top \bar{P}_{i,\mu}\mathcal{A}_i(P_k) - \bar{P}_{i,\mu} + S_{i,k}. \tag{30}$$

Applying $S_{i,k} = P^*_{i,k+1} - \mathcal{A}_i(P_k)^\top P^*_{i,k+1}\mathcal{A}_i(P_k)$ to (30) gives

$$\mathcal{R}_{i,\mu} = (P^*_{i,k+1} - \bar{P}_{i,\mu}) - \mathcal{A}_i(P_k)^\top (P^*_{i,k+1} - \bar{P}_{i,\mu})\mathcal{A}_i(P_k). \tag{31}$$

By using (29), $\bar{P}_{i,\mu} = \sum_{r=0}^{2^\mu-1}(\mathcal{A}_i(P_k)^\top)^r S_{i,k}(\mathcal{A}_i(P_k))^r$. Subtracting it from (28) gives

$$P^*_{i,k+1} - \bar{P}_{i,\mu} = \sum_{r=2^\mu}^{\infty}(\mathcal{A}_i(P_k)^\top)^r S_{i,k}(\mathcal{A}_i(P_k))^r. \tag{32}$$

Substituting (32) into (31) yields

$$\mathcal{R}_{i,\mu} = \sum_{r=2^\mu}^{\infty}(\mathcal{A}_i(P_k)^\top)^r \bar{S}_{i,k}(\mathcal{A}_i(P_k))^r \tag{33}$$

where $\bar{S}_{i,k} = S_{i,k} - \mathcal{A}_i(P_k)^\top S_{i,k}\mathcal{A}_i(P_k)$. Since $S_{i,k} \geq 0$ and $\mathcal{A}_i(P_k)$ is Schur stable, it is true that $S_{i,k} = \sum_{r=0}^{\infty}(\mathcal{A}_i(P_k)^\top)^r \bar{S}_{i,k}(\mathcal{A}_i(P_k))^r \geq 0$ and thus $\bar{S}_{i,k} \geq 0$. Then it can be derived from (33) that $\mathcal{R}_{i,\mu} \geq 0$ and $\mathcal{R}_{i,\mu} - \mathcal{R}_{i,\mu+1} = \sum_{r=2^\mu}^{2^{\mu+1}-1}(\mathcal{A}_i(P_k)^\top)^r \bar{S}_{i,k}(\mathcal{A}_i(P_k))^r \geq 0$. Hence, by induction, the following relations hold:

$$\mathcal{R}_{i,\infty} \leq \cdots \leq \mathcal{R}_{i,\mu+1} \leq \mathcal{R}_{i,\mu} \leq \cdots \leq \mathcal{R}_{i,0}. \tag{34}$$

Since $P_{i,0} = S_{i,k}$ and $\mathcal{A}_i(P_k)$ is Schur stable, it follows from (30) that $\mathcal{R}_{i,0} = \mathcal{A}_i(P_k)^\top S_{i,k}\mathcal{A}_i(P_k) \leq S_{i,k}$. Applying this to (34) yields $\mathcal{R}_{i,\mu} \leq S_{i,k}$, $\forall \mu \geq 0$. ■

The implementation of the proposed strategy is detailed in Algorithm 4, which consists of an inner iteration loop (Lines 4 - 6) and an outer iteration loop (Lines 3 - 15). The inner iteration solves the Stein equation (13) for each subsystem and constructs the common solution set $\{P^{\text{c}}_{\ell,k+1}\}$ using the function $\text{FindP}(A_i, B_i, Q_i, R_i, P_k, \alpha, \eta_{\text{in}})$ in Algorithm 3. This function is a detailed implementation of Lines 4&5 in Algorithm 2. For each subsystem, the search of a common solution is stopped either when a feasible common solution is found (see Lines 5&6, Algorithm 3) or when the relative change of the residual is within the given tolerance η_{in} (see Lines 7&8, Algorithm 3). The outer iteration constructs the CS solution sequence $\{P_{k+1}\}$, which is stopped when the relative change of the iterate is within the given tolerance η_{out} (see Lines 12&14, Algorithm 4). At iteration k, if the set $\{P^{\text{c}}_{\ell,k+1}\}$ is not empty, the obtained common iterate P_{k+1} is set as the element of $\{P^{\text{c}}_{\ell,k}\}$ that gives the smallest sum of residuals $\mathcal{R}^{\text{c}}_{\ell}$ (see Line 8, Algorithm 4). This ensures that the obtained common iterate solves the DAREs of all subsystems more accurately. If $\{P^{\text{c}}_{\ell,k+1}\}$ is empty, then setting $P_{k+1} = P_k$ (see Line 10, Algorithm 4). Algorithm 4 can be terminated at any iteration k and it still gives a CS solution.

Remark 3.4: In terms of T-S fuzzy or LPV systems, when using Algorithm 3 to search for the common solution $P^{\text{c}}_{i,k+1}$, the checking conditions in Line 7 need to include an extra condition: $\mathcal{A}_{i,s}(\bar{P}_{i,\mu})^\top \bar{P}_{i,\mu}\mathcal{A}_{i,s}(\bar{P}_{i,\mu}) < \bar{P}_{i,\mu}$, $\forall i \neq s,\ i,s \in [1,N]$, where $\mathcal{A}_{i,s}(\bar{P}_{i,\mu}) = A_i - B_i\mathcal{K}_s(\bar{P}_{i,\mu})$.

Algorithm 3 $[P^{\text{c}}_{i,k+1}, \mathcal{R}^{\text{c}}_i] = \text{FindP}(A_i, B_i, Q_i, R_i, P_k, \alpha, \eta_{\text{in}})$

1: $\mathcal{A}_i(P_k) \leftarrow A_i - B_i\mathcal{K}_i(P_k)$, $\bar{P}_{i,0} \leftarrow \mathcal{K}_i(P_k)^\top R_i\mathcal{K}_i(P_k) + Q_i$, $\mathcal{R}^0_i \leftarrow \mathcal{A}_i(P_{i,0})^\top \bar{P}_{i,0}\mathcal{A}_i(P_{i,0})$.
2: **for** $\mu = 1, 2, \ldots$ **do**
3: $\quad\bar{P}_{i,\mu} \leftarrow (\mathcal{A}_i(P_k)^\top)^{2^{\mu-1}}\bar{P}_{i,\mu-1}\mathcal{A}_i(P_k)^{2^{\mu-1}} + \bar{P}_{i,\mu-1}$.
4: $\quad$Construct $\{\mathcal{R}^\mu_j\}_{j=1}^N$, $\{W^\mu_j\}_{j=1}^N$ and $\{\bar{W}_j\}_{j=1}^N$ using $\mathcal{R}^\mu_j \leftarrow \mathcal{A}_j(P_k)^\top \bar{P}_{i,\mu}\mathcal{A}_j(P_k) - \bar{P}_{i,\mu} + S_{j,k}$, $W^\mu_j \leftarrow \mathcal{A}_j(P_k)^\top(\bar{P}_{i,\mu} - P_k)\mathcal{G}_j(P_k)(\bar{P}_{i,\mu} - P_k)\mathcal{A}_j(P_k)$, $\bar{W}_j \leftarrow W^\mu_j + \alpha Q_j - \mathcal{R}^\mu_j$, where $S_{j,k} = \mathcal{K}_j(P_k)^\top R_j\mathcal{K}_j(P_k) + Q_j$.
5: $\quad$**if** $(\{\bar{W}_j\}_{j=1}^N \geq 0)AND(\mathcal{R}^\mu_i \leq W^\mu_i)$ **then**
6: $\quad\quad P^{\text{c}}_{i,k+1} \leftarrow \bar{P}_{i,\mu}$, $\mathcal{R}^{\text{c}}_i \leftarrow \sum_{j=1}^N \|\mathcal{R}^\mu_j\|$, stop.
7: $\quad$**else if** $\|\mathcal{R}^\mu_i - \mathcal{R}^{\mu-1}_i\| \leq \eta_{\text{in}}\|\mathcal{R}^{\mu-1}_i\|$ **then**
8: $\quad\quad P^{\text{c}}_{i,k+1} \leftarrow [\,]$, $\mathcal{R}^{\text{c}}_i \leftarrow \infty$, stop. ([ ] is an empty matrix)
9: $\quad$**end if**
10: **end for**

Algorithm 4 Implementation of the proposed strategy

1: **Input:** $\{A_i\}_{i=1}^N$, $\{B_i\}_{i=1}^N$, $\{Q_i\}_{i=1}^N$, $\{R_i\}_{i=1}^N$, α, η_{in}, η_{out}.
2: **Initialise:** Solve Z from (8) and set $P_0 \leftarrow Z^{-1}$.
3: **for** $k = 0, 1, \ldots$ **do**
4: $\quad$**for** $i = 1, 2, \cdots, N$ **do**
5: $\quad\quad[P^{\text{c}}_{i,k+1}, \mathcal{R}^{\text{c}}_i] \leftarrow \text{FindP}(A_i, B_i, Q_i, R_i, P_k, \alpha, \eta_{\text{in}})$.
6: $\quad$**end for**
7: $\quad$**if** $\{P^{\text{c}}_{i,k+1}\}_{i=1}^N$ is not empty **then**
8: $\quad\quad P_{k+1} \leftarrow P^{\text{c}}_{i^*,k+1}$, where $i^* = \arg\min_{i\in[1,N]}(\mathcal{R}^{\text{c}}_i)$.
9: $\quad$**else**
10: $\quad\quad P_{k+1} \leftarrow P_k$.
11: $\quad$**end if**
12: $\quad$**if** $\|P_{k+1} - P_k\| \leq \eta_{\text{out}}\|P_k\|$ **then**
13: $\quad\quad P^* \leftarrow P_{k+1}$, $K_i \leftarrow \mathcal{K}_i(P_{k+1})$, $i \in [1,N]$, stop.
14: $\quad$**end if**
15: **end for**
16: **Output:** Common solution P^*, controller gains $\{K_i\}_{i=1}^N$.

IV. Numerical Examples

The simulations are carried out in MATLAB R2020a on a Windows machine with Intel i5-7200U CPU and 8GB RAM. The LMIs are solved using the toolbox YALMIP [36] and the solver MOSEK that adopts the interior point method [37].

Simulation 1: Large-scale mass-spring-damper systems. Consider a mass-spring-damper system consisting of M identical masses that are connected through identical springs [38]:

$$\begin{aligned}
\dot{x}_0 &= A_{\text{self}}(\bar{k})x_0 + A_{\text{inter}}(x_1 + x_{M-1}),\\
\dot{x}_j &= A_{\text{self}}(\bar{k})x_j + A_{\text{inter}}(x_{j-1} + x_{j+1}) + B_c u_j, j \in [1, M-2],\\
\dot{x}_{M-1} &= A_{\text{self}}(\bar{k})x_{M-1} + A_{\text{inter}}(x_0 + x_{M-2}) + B_c u_{M-1},
\end{aligned}$$

where $x_i = [\alpha_i, \dot{\alpha}_i]^\top$, $i \in [0, M-1]$, $B_c = [0, 1/\bar{m}]^\top$, $A_{\text{self}}(\bar{k}) = [0, 1; -(2k_1 + \bar{k})/\bar{m}, -2k_2/\bar{m}]$ and $A_{\text{inter}} = [0, 0; k_1/\bar{m}, k_2/\bar{m}]$. α_i and $\dot{\alpha}_i$ are the displacement and velocity of mass i, respectively. k_1 and $\bar{k}$ are the fixed and variable spring constants, respectively. k_2 is the damper coefficient. $\bar{m}$ is the mass of each block. u_i, $i \in [1, M-1]$, are the external force. The values of k_1, k_2, $\bar{m}$ and initial state are from [38].

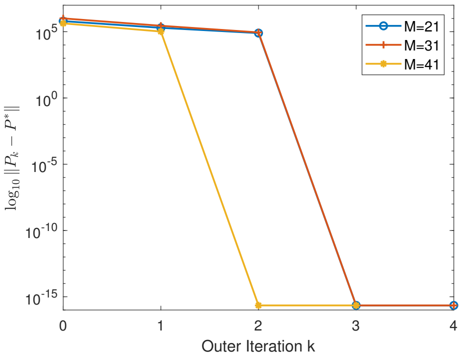

Fig. 1: Convergence of iterates for systems with M masses.

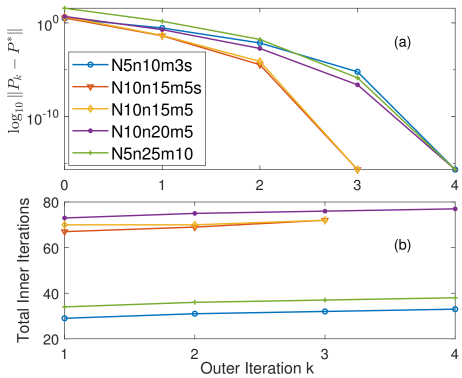

Fig. 3: Convergence of iterates and numbers of iterations.

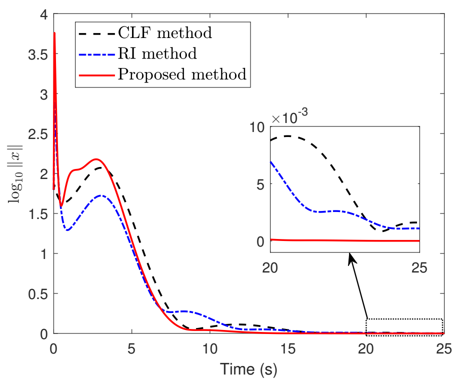

Fig. 2: Closed-loop state response of the 21 masses system using different methods.

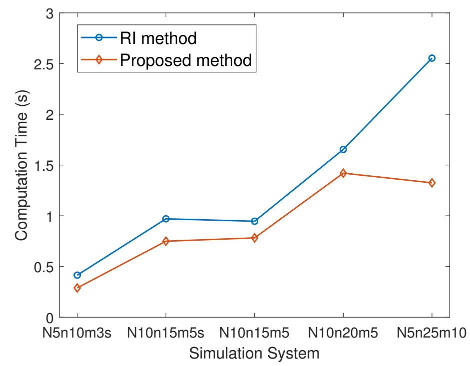

Fig. 4: Computation time of the RI and proposed methods.

The spring constant $\bar{k}$ is assumed to be capable of automatically switching between the values of $\bar{k}_1 = 1$ N/m and $\bar{k}_2 = 3.8$ N/m. This scenario could happen in certain mechatronic systems that encounter both low-frequency and high-frequency motions in one control task [15]. For simplicity, $\bar{k}$ of all subsystems are assumed to be $\bar{k}_1$ in the first t_0 seconds. After that, $\bar{k}$ of one subsystem switches from $\bar{k}_1$ to $\bar{k}_2$ at the start of each second. Hence, the mass-spring-damper system has $M+1$ subsystems. By defining $x = [x_0^\top, \cdots, x_{M-1}^\top]^\top$ and $u = [u_1, \cdots, u_{M-1}]^\top$ and considering the sampling time $t_s = 0.01$ s, the mass-spring-damper system can be reformulated as (1) with $N = M+1$, $n = 2M$ and $m = M-1$.

Simulations are performed for the mass-spring-damper system with $M = 21, 31, 41$. For each M, the proposed method designs a switched LQR controller $u(t) = K_{\delta(t)}x(t)$ whose gains $\{K_i\}_{i=1}^N$ are computed from Algorithm 4 using $\alpha = 0.95$, $\eta_{\text{in}} = \eta_{\text{out}} = 2.2204\text{e-}16$, $Q_i = I_n$ and $R_i = \sigma \times I_m$, $i \in [1, N]$, with $\sigma =$1.0e-6, 1.0e-10, 1.0e-13 for $M = 21, 31, 41$, respectively. Fig. 1 shows that in each case the sequence $\{P_k\}$ converges to the associated COS solution P^* within 4 iterations. The total computation time of the $M = 21, 31, 41$ cases are 38.4 s, 187 s and 708.1 s, respectively.

For comparison, the gains $\{K_i\}_{i=1}^N$ are also computed using the CLF method [14] and the RI method in Remark 3.1. The CLF method involves solving LMIs similar to (8) but with $Q_i = 0$ and $R_i = 0$, and "$\leq$" being replaced by "$<$". The RI method involves solving the LMI problem (9) using the same Q_i and R_i as the proposed method. The case when $M = 21$ is taken as an example to compare the performance of the three methods. The computation time of the CLF, RI and proposed methods are 15.5 s, 90.4 s and 38.4 s, respectively. To simulate the closed-loop control system, all the subsystems are assumed to have $\bar{k} = \bar{k}_1$ for $t \leq 4$ s. During $t \in (4, 25]$ s, one subsystem switches from $\bar{k}_1$ to $\bar{k}_2$ at the start of each second, by following a randomly generated subsystems order: $10, 7, 19, 13, 8, 14, 2, 12, 9, 3, 4, 17, 11, 16, 0, 20, 5, 6, 15, 18, 1$. Fig. 2 shows that the proposed method stabilises the closed-loop state x much faster than the CLF and RI methods.

Simulation 2: Randomly generated systems. Efficacy of the proposed strategy is further demonstrated by using five different discrete-time switched linear systems, whose matrix pairs (A_i, B_i) are randomly generated from the MATLAB function `drss`. The five systems are referred to as *N5n10m3s*, *N10n15m5s*, *N10n15m5*, *N10n20m5*, and *N5n25m10* respectively, where the integers next to N, n and m indicate their corresponding values, and the last character "*s*" indicates that all the subsystems are Schur stable. For example, *N5n10m3s* consists of 5 Schur stable subsystems each having 10 state variables and 3 control inputs. The simulations use the α, η_{in} and η_{out} in Case 1, and random s.p.d. matrices Q_i and R_i.

As shown in Fig. 3(a), the sequence $\{P_k\}$ converges quadratically to the associated COS solution P^* in each case. Fig. 3(b) shows the total inner iterations of all subsystems in each outer iteration. For all the five cases, the inner iterations needed to find the common solution increase with the outer iterations. This is mainly because $\lim_{k\to\infty} W_{i,k+1} \to 0$, which makes the conditions (17b) and (17c) increasingly difficult to satisfy. As shown in (34), increasing the inner iterations μ is sufficient for obtaining a smaller residual $\mathcal{R}_{i,k+1}$ to satisfy (17b) and (17c). Compared to the two cases with $N = 5$ (*N5n10m3s* and *N5n25m10*), the three cases with $N = 10$ (*N10n15m5s*, *N10n15m5* and *N10n20m5*) need much more total inner iterations. Also, when N is fixed, increases in the state dimension n and input dimension m lead to an increased number of inner iterations. Moreover, comparing the cases *N10n15m5s* and *N10n15m5* reveals that fewer inner iterations are needed when the subsystems are all Schur stable.

The computational complexity of the proposed and RI methods are compared in Fig. 4. The proposed method needs considerably less computation time than the RI method for all five randomly generated systems. Since the CLF method is used to obtain the initial common solution P_0 for the proposed method, its computation time is always less than the proposed method and is thus not reported here. Simulation results also showed that the obtained common solutions P^* (from the proposed method) and P_{RI} (from the RI method) satisfy $P^* < P_{\mathrm{RI}}$ for all five systems. These results demonstrate the promising advantages of the proposed method in obtaining a more optimal common solution with less computational cost.

V. Conclusion

An iterative strategy is proposed for finding the LQR weights such that the associated set of DAREs admit a common optimal stabilising solution. LMI conditions are derived for checking feasibility of the strategy and generating an initial feasible solution for the iteration. The iterative algorithm has quadratic rate of convergence and low computational complexity. The design efficacy has been illustrated through numerical simulations of systems with various sizes. The strategy is applied to determine switched LQR for switched linear systems under arbitrary switching and gain-scheduled LQR for T-S fuzzy or LPV systems in the entire operating region. Future research will investigate the possibility of extending the proposed strategy to include state and control input constraints.

References

[1] H. Lin and P. J. Antsaklis, "Stability and stabilizability of switched linear systems: a survey of recent results," *IEEE Trans. Autom. Control*, vol. 54, no. 2, pp. 308–322, 2009.

[2] F.-R. López-Estrada, D. Rotondo, and G. Valencia-Palomo, "A review of convex approaches for control, observation and safety of linear parameter varying and Takagi-Sugeno systems," *PRO.*, vol. 7, no. 11, p. 814, 2019.

[3] D. Liberzon, *Switching in systems and control*. Springer, 2003.

[4] J. Lan and R. J. Patton, *Robust Integration of Model-Based Fault Estimation and Fault-Tolerant Control*. Springer, 2020.

[5] J. Mohammadpour and C. W. Scherer, *Control of linear parameter varying systems with applications*. Springer, 2012.

[6] P. Li *et al.*, "Polytopic LPV approaches for intelligent automotive systems: State of the art and future challenges," *Mech. Syst. Signal Process.*, vol. 161, p. 107931, 2021.

[7] K. S. Narendra and J. Balakrishnan, "A common Lyapunov function for stable LTI systems with commuting A-matrices," *IEEE Trans. Autom. Control*, vol. 39, no. 12, pp. 2469–2471, 1994.

[8] Z. Sun and S. S. Ge, *Switched linear systems: control and design*. Springer, 2005.

[9] O. Mason and R. Shorten, "On linear copositive Lyapunov functions and the stability of switched positive linear systems," *IEEE Trans. Autom. Control*, vol. 52, no. 7, pp. 1346–1349, 2007.

[10] D. Liberzon and R. Tempo, "Common Lyapunov functions and gradient algorithms," *IEEE Trans. Autom. Control*, vol. 49, no. 6, pp. 990–994, 2004.

[11] R. H. Ordóñez-Hurtado and M. A. Duarte-Mermoud, "Finding common quadratic Lyapunov functions for switched linear systems using particle swarm optimisation," *Int. J. Control*, vol. 85, no. 1, pp. 12–25, 2012.

[12] S. Chen *et al.*, "Learning Lyapunov functions for hybrid systems," *arXiv preprint arXiv:2012.12015*, 2020.

[13] K. Tanaka and H. O. Wang, *Fuzzy control systems design and analysis: a linear matrix inequality approach*. John Wiley & Sons, 2004.

[14] V. Montagner, V. Leite, R. Oliveira, and P. Peres, "State feedback control of switched linear systems: An LMI approach," *J. Comput. Appl. Math.*, vol. 194, no. 2, pp. 192–206, 2006.

[15] L. Zhang, Y. Zhu, P. Shi, and Q. Lu, *Time-dependent switched discrete-time linear systems: Control and filtering*. Springer, 2016.

[16] W. Zhang, A. Abate, J. Hu, and M. P. Vitus, "Exponential stabilization of discrete-time switched linear systems," *Automatica*, vol. 45, no. 11, pp. 2526–2536, 2009.

[17] M. Fiacchini and S. Tarbouriech, "Control co-design for discrete-time switched linear systems," *Automatica*, vol. 82, pp. 181–186, 2017.

[18] A. A. Ahmadi and R. M. Jungers, "Switched stability of nonlinear systems via SOS-convex Lyapunov functions and semidefinite programming," in *Proc. IEEE Conf. Decis. Control*. IEEE, 2013, pp. 727–732.

[19] H. Dai, B. Landry, M. Pavone, and R. Tedrake, "Counter-example guided synthesis of neural network Lyapunov functions for piecewise linear systems," in *Proc. IEEE Conf. Decis. Control*. IEEE, 2020, pp. 1274–1281.

[20] M. V. Kothare, V. Balakrishnan, and M. Morari, "Robust constrained model predictive control using linear matrix inequalities," *Automatica*, vol. 32, no. 10, pp. 1361–1379, 1996.

[21] R. Guicherd, A. R. Mills, P. A. Trodden, and V. Kadirkamanathan, "Simultaneous and sequential control design for discrete-time switched linear systems using semi-definite programming," *IEEE Control Syst. Lett.*, vol. 5, no. 4, pp. 1393–1398, 2020.

[22] W. Zhang, J. Hu, and A. Abate, "Infinite-horizon switched LQR problems in discrete time: A suboptimal algorithm with performance analysis," *IEEE Trans. Autom. Control*, vol. 57, no. 7, pp. 1815–1821, 2011.

[23] E. Alcalá, V. Puig, and J. Quevedo, "TS-MPC for autonomous vehicles including a TS-MHE-UIO estimator," *IEEE Trans. Veh. Technol.*, vol. 68, no. 7, pp. 6403–6413, 2019.

[24] B. D. Anderson and J. B. Moore, *Optimal control: linear quadratic methods*. Courier Corporation, 2007.

[25] M. Lazar, W. Heemels, S. Weiland, and A. Bemporad, "Stabilization conditions for model predictive control of constrained PWA systems," in *Proc. IEEE Conf. Decis. Control*, vol. 5. IEEE, 2004, pp. 4595–4600.

[26] S. Chaib *et al.*, "Observer design for linear switched systems: A common Lyapunov function approach," in *Proc. IEEE Int. Conf. Control Appl.* IEEE, 2006, pp. 361–366.

[27] W. F. Arnold and A. J. Laub, "Generalized eigenproblem algorithms and software for algebraic Riccati equations," *Proc. IEEE*, vol. 72, no. 12, pp. 1746–1754, 1984.

[28] G. Hewer, "An iterative technique for the computation of the steady state gains for the discrete optimal regulator," *IEEE Trans. Autom. Control*, vol. 16, no. 4, pp. 382–384, 1971.

[29] P. Benner and H. Faßbender, "On the numerical solution of large-scale sparse discrete-time Riccati equations," *Adv. Comput. Math.*, vol. 35, pp. 119–147, 2011.

[30] A. Bouhamidi and K. Jbilou, "On the convergence of inexact Newton methods for discrete-time algebraic Riccati equations," *Linear Algebra Appl.*, vol. 439, no. 7, pp. 2057–2069, 2013.

[31] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan, *Linear matrix inequalities in system and control theory*. SIAM, 1994, vol. 15.

[32] X. Zeng, J. Chen, and Y. Hong, "Distributed optimization design for computation of algebraic Riccati inequalities," *IEEE Trans. Cybern.*, vol. 52, no. 3, pp. 1924–1935, 2020.

[33] J. Lan and R. J. Patton, "Integrated design of fault-tolerant control for nonlinear systems based on fault estimation and T–S fuzzy modeling," *IEEE Trans. Fuzzy Syst.*, vol. 25, no. 5, pp. 1141–1154, 2016.

[34] R. Smith, "Matrix equation XA+BX=C," *SIAM J Appl Math.*, vol. 16, no. 1, pp. 198–201, 1968.

[35] P. Lancaster and L. Rodman, *Algebraic Riccati equations*. Clarendon Press, 1995.

[36] J. Löfberg, "YALMIP: A toolbox for modeling and optimization in MATLAB," in *Proc. CACSD*, vol. 3, 2004.

[37] Mosek ApS, "The MOSEK optimization software." [Online]. Available: https://www.mosek.com

[38] Y. Cheng and V. Ugrinovskii, "Gain-scheduled leader-follower tracking control for interconnected parameter varying systems," *Int. J. Robust Nonlin.*, vol. 26, no. 3, pp. 461–488, 2016.