An Iterative Method for Control Gain Design of Multi-Agent Systems with Process Noise

Qiang Ling, Wei Zheng and Hai Lin

Abstract—This paper aims to optimize the consensus performance of multiple homogeneous agents, each of which is governed by a general discrete-time linear system with white process noise, exchange state information with its neighboring agents according to an undirected communication topology and generates its local control in a linear way. The common control gain of agents determines the consensus performance, which is measured by the ultimate mean square deviation of the states of agents. The consensus performance optimization with respect to the control gain takes a nonlinear matrix inequality form and is difficult to solve. To handle this nonlinearity issue, this paper proposes an iterative method. At each iteration, a descent direction of the control gain is computed by solving two linear matrix inequality optimizations based on a given feasible control gain. Then a line search algorithm is implemented to move the control gain along the obtained descent direction to improve the consensus performance. That updated control gain will work as the starting feasible control gain of the next iteration. This method can well handle the nonlinearity of the original consensus performance optimization and efficiently improve the consensus performance, which is confirmed by simulations.

Index Terms—Consensus, Multi-Agent Systems, Least Mean Square Deviation, Perturbation Method

I. INTRODUCTION

The consensus problem of multi-agent systems (MASs) has attracted more and more attention in recent years due to its wide applications in many areas, such as unmanned air vehicles (UAVs), wireless sensor networks, load balancing of parallel computers. The key point of solving that consensus problem is to design a consensus protocol, under which each agent performs a local control law with the received state information of neighboring agents and the states of all agents can asymptotically converge to a same value.

The theoretical framework of the consensus problem is introduced in [1][2]. The consensus problem of first-order MASs has been well studied in [3]-[5]. Those studies show that if the communication topology of the agents is a connected undirected graph or a digraph with a spanning tree, a consensus protocol can be designed to guarantee the consensus of multiple agents. Compared with first order MASs, high-order MASs can find more applicability in reality because agents may need to reach consensus as to more than one state variable. For example, UAVs may need to reach consensus as to position, velocity and acceleration. [6]-[9] provide consensus protocols for high-order MASs. In particular, a necessary and sufficient condition is provided in [9] to guarantee the concerned consensus can be reached, whose details will be given in Section II-B.

In the current literatures, MASs are often assumed to be noise-free. In reality, MASs are, however, often perturbed by various noises, such as process noise which results from electromagnetic interference, voltage drift and other disturbance. Noises prevent MASs from reaching consensus accurately by making the states of agents to vibrate around the average state. The robust consensus problem for MASs with noise is addressed in [10], which shows that the system can still reach consensus and the consensus deviation is upper bounded by a class $\mathcal{H}_\infty$ function. In [11], a neighbor-based protocol is adopted to guarantee high-order MASs with noise to reach consensus with a desired $H_\infty$ performance. [12] focuses on the performance analysis of broadcast-based consensus algorithms in the presence of non-zero-mean stochastic noise and provides some asymptotic upper and lower bounds of the state deviation in the mean square sense. [13] studies a kind of practical consensus, which means that all agents reach an agreement with certain error as to certain variables of interest, and proposes some sufficient conditions for swarm systems with noise to achieve the practical consensus. In the aforesaid works regarding MASs with noise, the main focus has been on analysis, i.e., to determine whether the concerned system can reach various types of consensus, such as weak consensus, practical consensus or bounded consensus. No much attention has been paid to design, i.e., how to improve the consensus performance by optimizing some parameters. As a rare example, [14] considers the average consensus problem for first-order integrator MASs with additive noise and provides a method to design the optimal edge weights, which results in the least mean square deviation of the steady state. As shown in [14], the weight optimization problem takes a convex form and can be efficiently solved. That convexity in [14] cannot be maintained in the present paper because it considers more general agent dynamics and different optimization variables, which are briefly introduced below.

This paper studies the consensus performance of general discrete-time homogeneous MASs with process noise. The connection of MASs is modelled as an undirected graph. As [14], We provide a performance index based on the ultimate mean square deviation of the states of agents to measure the disagreement of all agents under the perturbation of process
noise. We attempt to improve the performance index by optimizing the control gain of agents. The original control gain optimization problem takes a nonlinear matrix inequality form, which is difficult to handle. We propose an iterative method to solve that optimization, which is our main contribution. Each iteration starts from a given feasible control gain, which can guarantee the consensus of the system. By fixing the control gain, the original nonlinear problem becomes a linear matrix inequality (LMI) and can be efficiently solved to yield some intermediate matrix variables. With the obtained matrix variables and the given feasible control gain, we introduce a perturbation method to approximate the original nonlinear problem with another LMI. By solving that LMI problem, we generate a descent direction of the control gain. Moving the control gain along this descent direction, we can improve the system’s performance. By implementing a line search algorithm, we get a local optimal control gain. That achieves the system’s performance. By implementing a line search algorithm, we get a local optimal control gain. That achieved gain is able to guarantee the consensus of agents and therefore can be used as the starting feasible control gain of the next iteration. Our iterative method explicitly takes the system’s performance into account and can efficiently attenuate the effects of process noise.

The rest of this paper is organized as follows. Section II presents some preliminary knowledge about graph theory and basic models. In Section III, the concerned performance measure is introduced and the control gain optimization problem is formulated. An iterative method is proposed in Section IV to solve the control gain optimization problem. Section V is devoted to simulation results, which confirm the effectiveness of our iterative control gain design method. Finally, some concluding remarks are placed in Section VI.

The following notation will be followed throughout this paper. \( \theta_{m \times n} \) stands for a m \times n-dimensional matrix with the entries of \( \theta_{ij} \) (i = 1, ..., m; j = 1, ..., n). \( R^{m \times n} \) denotes the set of m \times n-dimensional real matrices. \( \otimes \) denotes the Kronecker product. For a given vector or matrix \( \xi, \xi^T \) denotes its transpose. \( E(\cdot) \) represents the expectation of a stochastic variable. \( tr(\cdot) \) is the trace operator of square matrices.

II. PRELIMINARY KNOWLEDGE

A. Some Definitions related to Graph Theory

Consider a communication network modelled as a connected undirected graph \( G = (V, E) \), where \( V = \{1, ..., N\} \) is a set of vertices and \( E \subseteq V \times V \) is a set of edges. We refer to the vertices as agents, and the edges as links. The edge \((i, j) \in E \) means that agent \( j \) can obtain information from agent \( i \). For an undirected graph, \((i, j) \in E \) implies \((j, i) \in E \). The set of neighbors of agent \( i \) is denoted by \( \mathcal{N}_i = \{j \in V : (j, i) \in E\} \).

The weighted adjacency matrix \( \mathcal{A} = [a_{ij}] \in R^{N \times N} \) of a graph is defined such that \( a_{ij} > 0 \) if \((j, i) \in E \), while \( a_{ij} = 0 \) otherwise. Define the Laplacian matrix \( L = [l_{ij}] \) of a graph as \( l_{ii} = \sum_{j \neq i} a_{ij} \) and \( l_{ij} = -a_{ij} \) for all \( i \neq j \). Let \( \lambda_i \) be the \( i^{th} \) smallest eigenvalue of \( L \). For a connected undirected graph, \( L \) is symmetric and \( 0 = \lambda_1 < \lambda_2 \leq ... \leq \lambda_N \). There must exist a unitary matrix \( \Phi = [\phi_1, ..., \phi_N] \in R^{N \times N} \) to transform \( L \) into a diagonal matrix, where \( \phi_1 = \frac{1}{\sqrt{\sum_{i=1}^{N} a_{ii}}} \) (\( 1_{N\times1} \) stands for an N-dimensional vector with all entries of 1) and \( \phi_i (i = 2, ..., N) \) are n-dimensional column vectors.

B. Model of Multi-Agent Systems

In this paper, we focus on the MAS whose communication topology is a connected undirected graph \( G \). The dynamics of agent \( i \) is governed by

\[
x_i(k+1) = Ax_i(k) + Bu_i(k) + \omega_i(k), \quad i = 1, \ldots, N, \tag{1}
\]

where \( x_i(k) \in R^{n \times 1} \) is the state of agent \( i \), \( u_i(k) \in R \) is the local control of agent \( i \) and \( \omega_i(k) \in R^{n \times 1} \) is the white zero-mean process noise with the constant covariance \( W = E(\omega_i(k)\omega_i(k)^T) \). \( A \in R^{n \times n} \) and \( B \in R^{n \times 1} \) are system matrices.

In [9], a weighted-average protocol generates the local control of agents by

\[
u_i(k) = K \sum_{j \in \mathcal{N}_i} a_{ij} (x_j(k) - x_i(k)), \tag{2}
\]

where \( K \in R^{1 \times n} \) is the common control gain for all agents. The main purpose of this paper is to design \( K \).

Under the protocol (2), the dynamics of a MAS with \( N \) agents can be described as the following discrete-time linear system,

\[
x(k+1) = (I_N \otimes A - L \otimes BK)x(k) + \omega(k), \tag{3}
\]

where \( x(k) = [x_1^T(k), ..., x_N^T(k)]^T \) and \( \omega(k) = [\omega_1^T(k), ..., \omega_N^T(k)]^T \). \( L \) is the Laplacian matrix of the undirected graph \( G \) and \( I_N \otimes K \) is an \( N \times N \) identity matrix.

Lemma 1: [9] If the communication topology \( G \) is a connected undirected graph, the discrete-time MAS (3) without noise (\( \omega(k) = 0 \) for all \( k \)) is consensusable under the protocol (2) if and only if the following conditions hold,

- \((A, B)\) is a controllable pair;
- Each agent cannot change too fast. More specifically, the product of the unstable eigenvalues of matrix \( A \) is upper bounded.

\[
\prod_j |\lambda_j^*(A)| < 1 + \frac{\lambda_2}{\lambda_N}, \quad 1 - \frac{\lambda_2}{\lambda_N}. \tag{4}
\]

where \( \lambda_j^*(A) \) represents the unstable eigenvalue of matrix \( A \).

In Lemma 1, the eigenratio \( \lambda_2/\lambda_N \) plays a critical role in the consensus of the system. Large \( \lambda_2/\lambda_N \) indicates a better network synchronizability, which allows a more unstable \( A \) to achieve consensus. This paper is based on the satisfaction of the conditions in Lemma 1. Under these conditions we can find a common control gain to guarantee that the agents reach consensus.

III. PROBLEM FORMULATION

For high-order MASs, it is difficult to evaluate the consensus performance directly from the states of agents. To resolve this issue, we introduce a deviation vector as [14].

Definition 1: The deviation vector of the state \( x(k) \) is defined as

\[
\delta(k) = x(k) - 1_{N\times1} \otimes \bar{x}(k), \tag{5}
\]
where \( \bar{x}(k) \) is the average state of all agents, i.e., \( \bar{x}(k) = \frac{1}{N} \sum_{i=1}^{N} x_i(k) \). \( \delta(x) \) can also be expressed in a compact form, 

\[
\delta(k) = (l \otimes I_{n \times n})x(k),
\]

(6) 

where \( l = I_{N \times N} = \left( \frac{1}{N} \right)_{N \times N}^{1 N \times 1 N \times 1} \), 

\( \delta(k) \) explicitly shows the deviation of the state of every agent from the average state. Using \( \delta(k) \), we can define a performance index to measure how well the MAS reach consensus, which is given below.

A. Performance Measure

Similar to the definition in [14], we define the following mean square deviation, 

\[
J(k) = tr \left( E(\delta(k)\delta(k)^T) \right).
\]

(7) 

\( J(k) \) quantitatively measures how far the agents are from consensus at time \( k \). Due to its time-varying property, \( J(k) \) is not suitable for optimization. We, therefore, focus on the ultimate limit of \( J(k) \), which is defined as 

\[
J = \lim_{k_0 \rightarrow \infty} \sup_{k \geq k_0} tr \left( E(\delta(k)\delta(k)^T) \right).
\]

(8) 

\( J \) can measure how well the system reaches consensus. For the noise-free consensus problem, \( J \) is zero when the system reaches consensus. For the consensus problem with process noise, the smaller is \( J \), the better consensus the system will reach.

B. Coordinate Transformation

Define \( z(k) = (\Phi \otimes I_{n \times n})^T \delta(k) \), where \( \Phi \) is the unitary matrix defined in Section II-A that transforms \( L \) into a diagonal matrix. Partition \( z(k) \) as \( z(k) = [z_1(k)^T, z_2(k)^T]^T \), where \( z_1(k) \) is the first \( n \) elements of \( z(k) \). According to the definition of \( \delta(k) \), we get 

\[
z(k) = (\Phi \otimes I_{n \times n})^T (l \otimes I_{n \times n}) x(k).
\]

(9) 

With \( z(k) \), the system (3) can be expressed as 

\[
z(k+1) = (\bar{A}_0 - \Lambda_0 \otimes B K) z(k) + (\Phi^T l \otimes I_{n \times n}) \omega(k),
\]

(10) 

where \( \bar{A}_0 = I_{N \times N} \otimes A \) and \( \Lambda_0 = diag(0, \lambda_2, \ldots, \lambda_N) \).

Because \( \phi_1^T l = 0_{N \times 1}^T (0_{N \times 1} \otimes I_{n \times n}) \) stands for an \( N \)-dimensional column vector with all entries of \( 0 \), we have 

\[
z_1(k) = (\phi_1^T l \otimes I_{n \times n}) x(k) = 0_{n \times 1}
\]

and 

\[
z_2(k) = (\Phi^T l \otimes I_{n \times n}) x(k) = 0_{n \times 1}.
\]

The system equation in (10) can be reduced into 

\[
z(k+1) = (\bar{A} - \Lambda \otimes B K) z(k) + (\Phi_2^T l \otimes I_{n \times n}) \omega(k),
\]

(11) 

where \( \bar{A} = I_{(N-1) \times (N-1)} \otimes A \) and \( \Lambda = diag(2, \ldots, \lambda_N) \) and \( \Phi_2 = [\phi_2, \ldots, \phi_N] \).

Considering the unitary nature of \( \Phi \), we can verify that 

\[
z(k)^T z(k) = \delta(k)^T (\Phi \otimes I_{n \times n}) (\Phi \otimes I_{n \times n})^T \delta(k) = \delta(k)^T \delta(k).
\]

Due to the fact \( z(k)^T z(k) = z_2(k)^T z_2(k) \), we know that the performance measure \( J \) is equivalent to 

\[
J = \lim_{k_0 \rightarrow \infty} \sup_{k \geq k_0} tr \left( E(z_2(k)z_2(k)^T) \right).
\]

(12) 

The coordinate transformation reduces the dimension of the original system. In the sequel, we will focus on the new system in (11).

C. Optimal Control Gain Design

Here we consider the problem of finding the optimal control gain \( K \) that yields the best performance measure. It can be formulated into the following optimization problem, 

\[
\min_K \lim_{k_0 \rightarrow \infty} \sup_{k \geq k_0} tr(E(z_2(k)z_2(k)^T))
\]

s.t. 

\[
z_2(k+1) = (\bar{A} - \Lambda \otimes B K) z_2(k) + \left( \Phi_2^T l \otimes I_{n \times n} \right) \omega(k).
\]

(13) 

It is difficult to solve the above optimization problem directly. So we will give an equivalent and tractable form of this optimization problem in the next Lemma. Before that, we define the feasible set of \( K \).

Definition 2: The feasible set of the control gain \( K \) is 

\[
\Omega_K = \{ K : \rho(A - \lambda_i + BK) < 1, i \in Z_{N-1} \},
\]

(14) 

where \( Z_{N-1} = \{ 1, \ldots, N - 1 \} \), \( \rho(\cdot) \) represents the spectral radius of a square matrix and \( \lambda_2, \ldots, \lambda_N \) are the eigenvalues of \( L \), which are defined in Section II-A.

Note that \( (\bar{A} - \Lambda \otimes B K) \) is a block diagonal matrix with the diagonal blocks, \( (A - \lambda_2 BK), \ldots, (A - \lambda_N BK) \). So, the eigenvalues of \( (\bar{A} - \Lambda \otimes B K) \) are the union of eigenvalues of \( (A - \lambda_i BK) \), \( i \in Z_{N-1} \). If \( K \in \Omega_K \), the eigenvalues of \( (\bar{A} - \Lambda \otimes B K) \) are inside the unite circle. With \( \Omega_K \), we provide an equivalent form of the optimization problem (13) in the following Lemma. The proof of Lemma 2 closely follows the technical procedures of [9][14] and is still provided in the Appendix for self-containedness.

Lemma 2: When \( K \in \Omega_K \), the optimization problem (13) is equivalent to the following one,

\[
\min_{K,P} tr(P)
\]

s.t. 

\[
P = (\bar{A} - \Lambda \otimes B K)P(\bar{A} - \Lambda \otimes B K)^T + R,
\]

(15) 

where \( P \in \mathbb{R}^{(N-1) \times (N-1)} \) and \( R = I_{N-1 \times (N-1)} \otimes W \).

By Lemma 2, the design of the optimal \( K \) is equivalent to minimizing the trace of matrix \( P \) which is subjected to a matrix equality. Note that \( P \) is a block diagonal matrix because 

\[
P = \sum_{i=1}^{N} (\bar{A} - \Lambda \otimes B K)^T R(\bar{A} - \Lambda \otimes B K)^T
\]

(15) 

and \( (\bar{A} - \Lambda \otimes B K) \) and \( R \) have the same block diagonal structure.

So \( P \) has the same block diagonal structure as \( (\bar{A} - \Lambda \otimes B K) \). Denote the diagonal blocks of \( P \) as \( P_1, P_2, \ldots, P_{N-1} \), where \( P_i \) has the same dimension as \( (A - \lambda_i BK) \) \( (i \in Z_{N-1}) \). Then we know that 

\[
P_i = (A - \lambda_i BK)P_i(A - \lambda_i BK)^T + R_i,
\]

(16) 

where \( R_i = W \) for \( i \in Z_{N-1} \). The optimization problem (15) can be expressed in a lower dimensional form,

\[
\min_{K,P_1,\ldots,P_{N-1}} tr(P_1 + \ldots + P_{N-1})
\]

s.t. 

\[
P_i = (A - \lambda_i BK)P_i(A - \lambda_i BK)^T + R_i, \quad \forall i \in Z_{N-1}.
\]

(17) 

The optimization problem (17) has two groups of decision variables, the control gain \( K \) and the matrix variables \( \{P_1, \ldots, P_{N-1}\} \). Due to the product term of \( (A - \lambda_i BK)P(A - \lambda_i BK)^T \), the problem is hard to solve [15]. But when \( K \) is fixed, the matrix variables \( P_i \) can be solved. Based on this observation, we introduce an iterative
method to efficiently solve the optimization problem (17) in Section IV.

IV. AN ITERATIVE METHOD FOR THE CONTROL GAIN OPTIMIZATION

In this section, we propose an iterative method to solve the nonlinear optimization problem (17). Our method consists of 4 steps, which are shown in Fig. 1.

A. Solve Matrix Variables under given $K$

Suppose a control gain $K(\in \Omega_K)$ is given. For example, such $K$ can be generated by the method in [9]. We attempt to find the optimal matrix variables, $P_1, \cdots, P_{N-1}$, under the constraints of the optimization problem (17).

When $K$ is fixed, the optimization problem (17) can be replaced by the following LMI

$$\min_{P_1, \cdots, P_{N-1}} \quad tr(P_1 + \cdots + P_{N-1})$$
$$s.t. \quad P_i > (A - \lambda_{i+1} BK)P_i(A - \lambda_{i+1} BK)^T + R_i, \quad \forall i \in Z_{N-1}. \quad (18)$$

By implementing Schur complement to the above inequalities, we obtain the following equivalent LMI optimization,

$$\min_{P_1, \cdots, P_{N-1}} \quad tr(P_1 + \cdots + P_{N-1})$$
$$s.t. \quad \begin{bmatrix} P_i - B_i A_i P_i A_i^T & P_i^T \\ P_i & P_i \end{bmatrix} > 0, \quad \forall i \in Z_{N-1}, \quad (19)$$

where $A_i = A - \lambda_{i+1} BK$.

Optimization (19) can be efficiently solved. For any $K \in \Omega_K$, we can surely find matrix variables, $P_1, \cdots, P_{N-1}$, which will be used to compute a descent direction of $K$.

B. A Descent Direction of $K$

In Section IV-A, we compute the performance $J$ under the given $K$. $J$ is actually a function of $K$. In order to further improve (reduce) $J$, we want to find a descent direction of $K$, which is obtained through the following perturbation method.

We introduce perturbation into $K$ and $P_i$, i.e., $K \rightarrow K + \Delta K$ and $P_i \rightarrow P_i + \Delta P_i$. Under that perturbation, $J \rightarrow J + \Delta J$ with $\Delta J = tr(\Delta P_1 + \cdots + \Delta P_{N-1})$. We want to find a direction of $\Delta K$ to minimize $\Delta J$. Such minimization must be performed without violating the constraints in (16), which are changed into

$$P_i + \Delta P_i = (A - \lambda_{i+1} B(K + \Delta K)) \times (P_i + \Delta P_i) (A - \lambda_{i+1} B(K + \Delta K))^T + R_i. \quad (20)$$

Due to (16), the above equation yields

$$\Delta P_i = A_i \Delta P_i A_i^T - A_i \Delta P_i (\lambda_{i+1} B K)^T - \lambda_{i+1} B \Delta K P_i A_i^T$$
$$+ \lambda_{i+1} B \Delta K (P_i + \Delta P_i) (\lambda_{i+1} B K)^T$$
$$- A_i \Delta P_i (\lambda_{i+1} B K)^T - \lambda_{i+1} B \Delta K \Delta P_i A_i^T. \quad (21)$$

Of course, we can find the descent direction, $\Delta K$, by solving the following optimization

$$\min_{\Delta K, \Delta P_1, \cdots, \Delta P_{N-1}} \quad tr(\Delta P_1 + \cdots + \Delta P_{N-1}), \quad (22)$$
$$s.t. \quad \begin{bmatrix} \Delta P_1 > A_i \Delta P_i A_i^T \\ - A_i \Delta P_i (\lambda_{i+1} B K)^T - \lambda_{i+1} B \Delta K P_i A_i^T + \lambda_{i+1} B \Delta K (P_i + \Delta P_i) (\lambda_{i+1} B K)^T \\ - A_i \Delta P_i (\lambda_{i+1} B K)^T - \lambda_{i+1} B \Delta K \Delta P_i A_i^T \end{bmatrix}. \quad (23)$$

Unfortunately the optimization (22) cannot be easily solved due to the high-order terms of the constraints in (23), such as $\lambda_{i+1} B \Delta K \Delta P_i A_i^T$. A similar problem was discussed in [16]. Although we can neglect the high-order terms in (23) to get approximate linear constraints, that approximation error may be too large and the obtained $\Delta K$ cannot effectively reduce $\Delta J$. We cope with the high-order terms from another way.

Suppose $-I_{n \times n} < \Delta P_i < I_{n \times n}$ for $i \in Z_{N-1}$. We add the non-negative term, $A_i \Delta P_i (A_i \Delta P_i)^T + \lambda_{i+1} B \Delta K (I_{n \times n} - \Delta P_i) (\lambda_{i+1} B K)^T$, to the right hand of (23) to yield

$$\Delta P_i > A_i \Delta P_i A_i^T - A_i \Delta P_i (\lambda_{i+1} B K)^T - \lambda_{i+1} B \Delta K P_i A_i^T + \lambda_{i+1} B \Delta K (P_i + \Delta P_i) (\lambda_{i+1} B K)^T + (A_i \Delta P_i - \lambda_{i+1} B \Delta K) (\lambda_{i+1} B K)^T. \quad (24)$$

Comparing the constraints in (23) and the ones in (24), we see that the feasible set of $\Delta P_i$ defined by (24) is a subset of that one defined by (23) and the difference between the two feasible sets become negligible when $\Delta K$ is small enough.

Implementing Schur complement to (24) gives us

$$\begin{bmatrix} M_{i1} & M_{i2} & M_{i3} \\ * & I_{n \times n} & 0 \\ * & * & P_i \end{bmatrix} > 0, \quad (25)$$

where $M_{i1} = \Delta P_i - A_i \Delta P_i A_i^T + A_i \Delta P_i (\lambda_{i+1} B K)^T + \lambda_{i+1} B \Delta K P_i A_i^T$, $M_{i2} = A_i \Delta P_i - \lambda_{i+1} B \Delta K$, and $M_{i3} = \lambda_{i+1} B \Delta K P_i$.

In summary, we obtain a descent direction of $K$, $\Delta K$, by solving the following LMI problem,

$$\min_{\Delta K, \Delta P_1, \cdots, \Delta P_{N-1}} \quad tr(\Delta P_1 + \cdots + \Delta P_{N-1})$$
$$s.t. \quad \begin{bmatrix} M_{i1} & M_{i2} & M_{i3} \\ * & I_{n \times n} & 0 \\ * & * & P_i \end{bmatrix} > 0, \quad (26)$$

where $M_{i1} = \Delta P_i - A_i \Delta P_i A_i^T + A_i \Delta P_i (\lambda_{i+1} B K)^T + \lambda_{i+1} B \Delta K P_i A_i^T$, $M_{i2} = A_i \Delta P_i - \lambda_{i+1} B \Delta K$, and $M_{i3} = \lambda_{i+1} B \Delta K P_i$. $\forall i \in Z_{N-1}$.
In the above optimization, the constraints $-I_{n \times n} < \Delta P_i < I_{n \times n}$ place bounds on $\Delta P_i$ and guarantee the feasible set of $\Delta P_i$ is NOT infinite. Because what we expect here is just a descent direction of $K$, the scaling effect due to $-I_{n \times n} < \Delta P_i < I_{n \times n}$ does not matter, which will be explained more clear in Section IV-C.

C. Line Search along the Descent Direction

In Section IV-B, we obtain a descent direction of $K$, $\Delta K$. Here we want to implement a line search algorithm to improve the consensus performance. More specifically, we choose the following new control gain,

$$K^+ = K + \epsilon \Delta K,$$

where $\epsilon$ is the step size. Note that $\epsilon$ can affect the consensus performance $J$. To emphasize such effects, $J$ is represented as $J(\epsilon)$, which can be computed through solving the LMI optimization (19), i.e.,

$$J(\epsilon) = \min_{P_1, \ldots, P_{N-1}} \text{tr}(P_1 + \ldots + P_{N-1})$$

s.t. $\left[ \begin{array}{cc} P_i - R_i & A(\epsilon_i)P_i \\ P_i^T A(\epsilon_i)^T & P_i \end{array} \right] > 0, \forall i \in Z_{N-1},$ (28)

where $A(\epsilon_i) = A - \lambda_{i+1} B(K + \epsilon \Delta K)$. We want to find an appropriate $\epsilon$ to improve the consensus performance, i.e.,

$$\min \epsilon \ J(\epsilon).$$

(29)

Because $J(\epsilon)$ may not be convex with respect to $\epsilon$, it is not easy to find the optimal $\epsilon$ by solving the above optimization. Actually the optimal solution of the optimization (29) could not be so important because $\Delta K$ is just an approximate descent direction of $K$. Our focus is to try our best to improve/reduce $J(\epsilon)$ by finding appropriate $\epsilon$. The line search in Algorithm 1 is implemented to improve $J(\epsilon)$. Note that $\delta_k$ in Algorithm 1 is a small positive threshold.

Algorithm 1 Framework of the line search algorithm

Initialize $\epsilon_0 = 0.01, \epsilon = 0$
repeat
    if $J(\epsilon + \epsilon_0) \leq J(\epsilon)$ then
        $\epsilon = \epsilon + \epsilon_0$
        $\epsilon_0 = 2 \times \epsilon_0$
    else
        $\epsilon_0 = -\frac{1}{2} \times \epsilon_0$
    end if
until $|\epsilon_0| < \delta_k$
return $\epsilon$

D. Stopping criterion

In Section IV-C, an updated control gain $K^+$ is generated from the last control gain $K$ by the linear search algorithm. Such $K^+$ is provided to Section IV-A and starts another iteration. When the difference between $K^+$ and $K$ is too small, it is reasonable to stop the iteration of our control gain design method. So we take the following stopping criterion,

$$||K^+ - K||_2 = ||\epsilon \Delta K||_2 < \delta_K,$$

(30)

where $|| \cdot ||_2$ stands for the 2-norm of a vector and $\delta_K$ is a small positive threshold.

By performing the operations of Section IV-A – Section IV-C, our method is expected to resolve the nonlinearity issue of the original optimization problem (18) and improve the consensus performance in each iteration. Besides the stopping criterion in (30), other criteria, such as setting an upper bound on the number of iterations, can also be implemented. Although the method cannot guarantee to achieve the globally optimal control gain, it can generate a better control gain from an initial one (e.g., the control gain provided by [9]) in the sense of improving the consensus performance, which is illustrated in the following section.

V. ILLUSTRATIVE EXAMPLES

In this section, the effectiveness and efficiency of our iterative method is verified through some example systems, which are a third-order MAS with the dynamics in (1) and the following system matrices,

$$A = \begin{bmatrix} 3.6 & -4.31 & 1.716 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}.$$  \label{eq:sys}

The process noise is white zero-mean Gaussian with the variance of $W = 0.25I_{3 \times 3}$. We choose $\delta_k = 0.001$ and $\delta_K = 0.0001$. We consider two different communication topologies, including a large eigenratio one and a small eigenratio one, which are shown in Fig. 2.

Our method is compared with another LMI method adopted from [17]. According to [17], we can approximate the optimization problem (19) into

$$\min_{G,P_1, \ldots, P_{N-1}} \text{tr}(P_1 + \ldots + P_{N-1})$$

s.t. $\left[ \begin{array}{cc} P_i - R_i & A_i G \\ G^T A_i^T & G + G^T - P_i \end{array} \right] > 0, \forall i \in Z_{N-1},$ (31)

where $G$ is a full matrix with the same dimension as $P_i$. It can be verified that the constraints in (31) imply those in (19).

In order to solve (31), we follow [17] to define $X = G$ and $L = KG$ and equivalently rewrite the constraints in (31) into

$$\left[ \begin{array}{cc} P_i - R_i & A_i X - \lambda_{i+1} BL \\ (AX - \lambda_{i+1} BL)^T & X + X^T - P_i \end{array} \right] > 0,$$

(32)

where $i \in Z_{N-1}$. When the above constraints are feasible, $K = LX^{-1}$. Due to the LMI form, the optimization problem (31) can be efficiently solved to yield some sub-optimal performance (because (31) implies (19); but not vice versa).
LMI method in (32), respectively. For easy comparison, the
be seen that the \( \hat{J}(k) \) curves under the control gains by our method and the LMI method in (32), respectively. Note that in Fig. 3, "Our" and "LMI" represent the results by our method and the LMI method in (32), respectively. After 5 iterations, our method obtains the final control gain \( K = [0.3639, -0.4702, 0.1800] \), which corresponds to \( J = 18.7203 \). Compared with the initial control gain from [9], our method improves \( J \) by 62.1\%, which is not surprising because [9] aims to optimize the convergence rate of noise-free multi-agent systems, instead of \( J \). Compared with the LMI method, our method improves \( J \) by 1.5\%, which is not significant.

For further comparison, we blindly compute \( J \) for \( K = [k_1, k_2, k_3] \) with \( k_1 = 0.2 : 0.001 : 0.4, k_2 = -0.6 : 0.001 : -0.4 \) and \( k_3 = 0.1 : 0.001 : 0.3 \), i.e., we compute \( J \) of 201 \( \times \) 201 grids. The best performance is achieved by the grid \( K = [0.364, -0.470, 0.180] \) with \( J = 18.7205 \). The performance of our method is very close to the one of the best grid.

\( J(k) \), an estimate of \( J(k) \) in (7), was computed from \( M(=10,000) \) samples according to the following rule,

\[
J(k) = \frac{1}{M} \sum_{j=1}^{M} \text{tr} \left( \delta^{(j)}(k) \delta^{(j)}(k)^T \right),
\]

where \( \delta^{(j)}(k) \) stands for the state deviation vector of the \( j \)-th sample at time \( k \). Three \( J(k) \) curves are shown in Fig. 4, where "initial" represents the \( J(k) \) curve under the initial control gain from [9], and "Our" and "LMI" represent the \( J(k) \) curves under the control gains by our method and the LMI method in (32), respectively. For easy comparison, the best \( J \) computed by our method is also plotted in Fig. 4. It can be seen that the \( \hat{J}(k) \) curve of our method is below the ones of the LMI method and the control gain from [9], which again confirms the performance advantage of our method. Moreover, we see that \( \hat{J}(k) \) oscillates around \( J \) for \( k \geq 20 \), which indicates that the MAS well reaches consensus when \( k \geq 20 \).

### A. The Large Eigenratio Case

The large eigenratio topology is illustrated in Fig. 2(a). The second smallest eigenvalue of its Laplacian matrix is \( \lambda_2 = 8 \) and the largest eigenvalue is \( \lambda_{10} = 10 \). So its eigenratio \( \lambda_2/\lambda_{10} \) is equal to 0.8. The product of the unstable eigenvalues of the system matrix \( A \) is 1.716. Therefore, the conditions in Lemma 1 are satisfied and an initial feasible control gain \( K = [0.2663, -0.4289, 0.1907] \) can be obtained by the method in [9], which yields \( J = 49.4011 \). By solving the LMI problem in (32), we obtain \( K = [0.3544, -0.4639, 0.1822] \), which yields \( J = 19.0038 \). Our iterative method is implemented to this case, whose iterations are shown in Fig. 3. Note that in Fig. 3, "Our" and "LMI" represent the results by our method and the LMI method in (32), respectively. After 5 iterations, our method obtains the final control gain \( K = [0.3639, -0.4702, 0.1800] \), which corresponds to \( J = 18.7203 \). Compared with the initial control gain from [9], our method improves \( J \) by 62.1\%, which is not surprising because [9] aims to optimize the convergence rate of noise-free multi-agent systems, instead of \( J \). Compared with the LMI method, our method improves \( J \) by 1.5\%, which is not significant.

### B. The Small Eigenratio Case

The topology of the small eigenratio case is illustrated in Fig. 2(b). The second smallest eigenvalue of its Laplacian matrix is \( \lambda_2 = 3.7466 \) and the largest eigenvalue is \( \lambda_{10} = 10 \). Its eigenratio \( \lambda_2/\lambda_{10} \) is equal to 0.3747. The conditions in Lemma 1 are also satisfied and the method in [9] provides us an initial feasible control gain \( K = [0.2683, -0.5100, 0.2497] \), which yields \( J = 1770.1564 \). By solving the LMI problem in (32), we obtain \( K = [0.2747, -0.5009, 0.2328] \), which yields \( J = 1160.9131 \). Our iterative method is implemented to this case, whose iterations are shown in Fig. 5. After 174 iterations, our method obtains the final control gain \( K = [0.3128, -0.5251, 0.2216] \), which corresponds to \( J = 513.6678 \). Compared with the initial control gain from [9], our method improves \( J \) by 71.0\%. Compared with the LMI method, our method improves \( J \) by 55.8\%. Note that we pay much high computational cost for such 55.8\% improvement.

As Section V-A, we blindly compute \( J \) for \( K = [k_1, k_2, k_3] \) with \( k_1 = 0.2 : 0.001 : 0.4, k_2 = -0.6 : 0.001 : -0.4 \) and \( k_3 = 0.2 : 0.001 : 0.3 \), i.e., we compute \( J \) of 201 \( \times \) 201 grids. The best performance is achieved by the grid \( K = [0.313, -0.525, -0.179] \) with \( J = 514.7531 \). Due to the limited resolutions of \( k_1, k_2 \) and \( k_3 \), the performance of our method is even slightly better than the one of the best grid. All these comparison results confirm the performance advantage of our method.
existence and uniqueness of the optimal solution of the original optimization is nonlinear and hard to solve. We introduce an appropriate control gain of agents to improve the consensus performance of multi-agent systems. The original control gain curves in Fig. 6. In this small eigenratio case, the performance advantage of our method becomes more prominent than the large eigenratio case in Fig. 4.

VI. CONCLUSION

This paper investigates the problem of designing an appropriate control gain of agents to improve the consensus performance of multi–agent systems. The original control gain optimization is nonlinear and hard to solve. We introduce an iterative method to solve that optimization. Simulation results demonstrate the control gain from our method can achieve better consensus performance than the previous methods. The optimality of our method, however, is still an open question. Because the original control gain optimization problem is not convex and even the feasible set \( \Omega_K \) is not convex, the existence and uniqueness of the optimal solution of the original optimization is not guaranteed. So it is hard to determine whether our method can achieve the global optimal solution although the performance of our method is quite pleasing.

APPENDIX: PROOF OF LEMMA 2

From (11), we get

\[
\begin{align*}
z_2(k+1)z_2(k+1)^T & = (\bar{A} - \Lambda \otimes BK)z_2(k)z_2(k)^T (\bar{A} - \Lambda \otimes BK)^T + (\bar{A} - \Lambda \otimes BK)\omega(k)T (\Phi_2^T1 \otimes I_{n \times n})^T \\
& + (\Phi_2^T1 \otimes I_{n \times n})\omega(k)^T (\Phi_2^T1 \otimes I_{n \times n})^T + (\Phi_2^T1 \otimes I_{n \times n})^T (\Phi_2^T1 \otimes I_{n \times n}) \\
& + (\Phi_2^T1 \otimes I_{n \times n})\omega(k)^T (\Phi_2^T1 \otimes I_{n \times n})^T \\
& + (\Phi_2^T1 \otimes I_{n \times n})^T (\Phi_2^T1 \otimes I_{n \times n})^T \\
& + (\Phi_2^T1 \otimes I_{n \times n})^T (\Phi_2^T1 \otimes I_{n \times n})^T.
\end{align*}
\]

By the system’s dynamics in (11), we know that \( z_2(k) \) is determined by the process noise before time \( k \). Because the process noise \( \{w(k)\} \) is white, \( z_2(k) \) is independent of \( \omega(k) \) and \( E(z_2(k)\omega(k)^T) = E(\omega(k))E(\omega(k)^T) = 0_{(N-1) \times N} \). Considering such independence and \( P(k) = E(z_2(k)z_2(k)^T) \), we can obtain, from (34),

\[
P(k+1) = (\bar{A} - \Lambda \otimes BK)P(k)(\bar{A} - \Lambda \otimes BK)^T + R, \tag{35}
\]

where \( R = (\Phi_2^T1 \otimes I_{n \times n})E(\omega(k)\omega(k)^T)(\Phi_2^T1 \otimes I_{n \times n})^T \). Since the process noises of different agents, \( \omega_i(k)(i = 1, \ldots , N) \), are independent and the covariance matrices are identical for all agents, i.e., \( E(\omega_i(k)\omega_i(k)^T) = W(i = 1, \ldots , N) \), the matrix \( R = (\Phi_2^T1 \otimes I_{n \times n})(I_{N \times N} \otimes W) (\Phi_2^T1 \otimes I_{n \times n})^T = I_{N-1 \times (N-1)} \otimes W \).

Because \( K \in \Omega_K \), all eigenvalues of \( (\bar{A} - \Lambda \otimes BK) \) are inside the unit circle. Thus the iteration of \( P(k) \) in (35) will converge, i.e., \( \lim_{k \to \infty} P(k) \) exists. Define \( P = \lim_{k \to \infty} \sup \{P(k)\} \). Then considering the limiting situation of (35), i.e., \( k \to \infty \), we can replace both \( P(k+1) \) and \( P(k) \) with \( P \) and get

\[
P = (\bar{A} - \Lambda \otimes BK)P(\bar{A} - \Lambda \otimes BK)^T + R. \tag{36}
\]

Because the performance index of the optimization problem (13) is a kind of ultimate limit, only the ultimate constraints, i.e., \( k \to \infty \), are active. Such ultimate constraints are equivalent to (36) according to the above derivation. Due to the definition of \( P \), we know \( tr(P) \) is actually the performance index of the optimization problem (13). So the optimization problems (13) and (15) have the same performance index and the same active constraints, and should be equivalent to each other.

REFERENCES


