Robotics: Science and Systems 2021
Held Virtually, July 12–16, 2021

Distributed Covariance Steering with Consensus ADMM for Stochastic Multi-Agent Systems

Augustinos D. Saravanos[1], Alexandros G. Tsolovikos[2], Efstathios Bakolas[2] and Evangelos A. Theodorou[1]
[1]Georgia Institute of Technology, GA, USA
[2]The University of Texas at Austin, TX, USA
Email: asaravanos3@gatech.edu

***Abstract*—In this paper, we address the problem of steering a team of agents under stochastic linear dynamics to prescribed final state means and covariances. The agents operate in a common environment where inter-agent constraints may also be present. In order for our method to be scalable to large-scale systems and computationally efficient, we approach the problem in a distributed control framework using the Alternating Direction Method of Multipliers (ADMM). Each agent solves its own covariance steering problem in parallel, while additional copy variables for its closest neighbors are introduced to ensure that the inter-agent constraints will be satisfied. The inclusion of these additional variables creates a requirement for consensus between original and copy variables that involve the same agent. For this reason, we employ a variation of ADMM for consensus optimization. Simulation results on multi-vehicle systems under uncertainty with collision avoidance constraints illustrate the effectiveness of our algorithm. The substantially improved scalability of our distributed approach with respect to the number of agents is also demonstrated, in comparison with an equivalent centralized scheme.**

I. Introduction

Multi-agent control is a discipline with applications in several fields of robotics, such as the formation of swarms of drones [28], convoys of autonomous vehicles [30], coverage control [20] and multi-robot coordination [22], to name but a few. In such problems, the members of the multi-agent system have to work in unison to complete specific tasks while operating safely in a common environment. These are challenging problems, especially when the agents operate under uncertainty. Having probabilistic guarantees on the stochastic state trajectories of such agents is imperative to ensure their safe operation and optimal performance. Toward that goal, we leverage recent results in covariance steering to address the problem of guiding a team of agents to prescribed goal state distributions while satisfying probabilistic inter-agent constraints.

In contrast with standard LQG control where the final state covariance is indirectly controlled, covariance steering [17] aims at driving the final state mean and covariance to specific prescribed targets. While the first contributions [15, 17, 33] in the area focused on the steady-state (infinite-horizon) covariance control problem, recently, covariance steering has also been formulated in a finite-horizon control setting, under continuous [2, 10, 11, 12, 16] and discrete-time [3, 4, 7, 8] linear dynamics, as well as for systems with partial state information [5, 18, 27] and nonlinear dynamics [26, 31, 34].

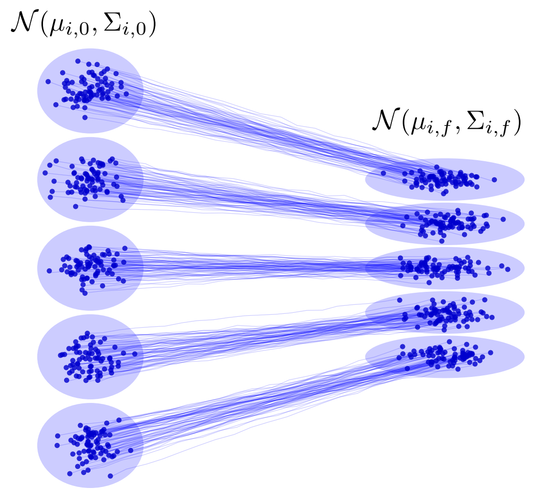

$\mathcal{N}(\mu_{i,0}, \Sigma_{i,0})$
$\mathcal{N}(\mu_{i,f}, \Sigma_{i,f})$

Fig. 1: The problem of steering a team of agents under stochastic dynamics from their initial Gaussian distributions to prescribed final ones.

Several successful applications of covariance control can be found in robotics [21], aerospace engineering [25], etc.

Covariance steering essentially provides an upper bound on the uncertainty of the final state. This attribute makes it very appealing for problems where safety guarantees have to be established. For instance, consider the scenario of driving multiple agents from an initial formation to a desired one. In this case, it is critical that the final positions of the agents will be within a maximum specific distance from their target positions with a prescribed probability while avoiding collisions.

Solving the covariance steering problem for discrete-time linear systems scales significantly with the number of states, control inputs, and time steps [4]. Therefore, attempting to deal with the multi-agent covariance steering problem in a centralized fashion can prove to be computationally demanding, especially as the number of agents increases. Consequently, it is of paramount importance to come up with an approach that will be computationally efficient and that will scale well with the increase of agents. For this reason, we address the problem in a distributed control framework using the Alternating Direction Method of Multipliers (ADMM).

ADMM is a powerful optimization method [9] suitable for distributed optimization and control. There exist some works that have employed ADMM in multi-agent control problems [13, 19, 23, 24, 29, 32]. One of the main advantages of several ADMM variations is that they allow for large-scale multi-agent optimization problems to be handled in a decentralized fashion, yielding significantly faster solutions compared to a centralized approach. Therefore, we believe that ADMM is very suitable for multi-agent stochastic optimal control where the computational effort must be distributed properly between the available processing units so that we achieve scalability to large-scale systems.

In this paper, we propose a distributed algorithm for multi-agent covariance steering based on a variation of ADMM for consensus optimization. Our main contribution can be seen from two perspectives. First, to the best of our knowledge, the multi-agent covariance steering problem has not been addressed yet. For the reasons we have stated, we believe that covariance steering fits well with problems where we wish to establish probabilistic guarantees for safety and, therefore, it is suitable for the control of multiple agents in a common environment. Second, we address the issue of increased computational complexity incurred by centralized multi-agent covariance steering. To do so, we use a distributed optimization architecture based on ADMM that is shown to be both effective and scalable to a large number of agents. Although this work specifically addresses covariance steering, it also indicates the compatibility of the distributed nature of ADMM with multi-agent stochastic optimal control in general and can potentially lead to more powerful algorithms for this class of problems.

The paper is structured as follows. Section II provides an overview of single-agent covariance steering and ADMM. In Section III, we formulate the multi-agent covariance control problem. Our proposed distributed ADMM-based covariance steering algorithm is presented in Section IV. In Section V, we demonstrate simulation results that verify the effectiveness and scalability of our approach. The conclusions of our work and a discussion for future directions are provided in Section VI.

II. Preliminaries

In this section, we introduce some prerequisites on covariance steering and ADMM. First, we define the notation that we follow throughout the paper. Next, we present the single-agent covariance steering problem for discrete-time stochastic linear systems and demonstrate how it can be reduced to a convex semi-definite program (SDP). Finally, we overview ADMM and one of its variations for addressing consensus optimization problems in a distributed fashion.

A. Notation

The mean and covariance of a random vector x are denoted by $\mathbb{E}[x]$ and $\mathrm{Cov}[x]$, respectively, where $\mathrm{Cov}[x] := \mathbb{E}[(x - \mathbb{E}[x])(x - \mathbb{E}[x])^\top]$. Let $\mathbb{S}_n$ be the space of real symmetric $n \times n$ matrices, whereas $\mathbb{S}_n^+$ and $\mathbb{S}_n^{++}$ denote the convex cone of $n \times n$ positive semi-definite and positive definite matrices, respectively. Given a matrix $A \in \mathbb{R}^{n\times n}$, its trace is denoted as $\mathrm{tr}(A)$. With $\mathrm{diag}(a_1, \ldots, a_\ell) \in \mathbb{R}^{\ell\times\ell}$ we denote the diagonal matrix made up by the scalars a_i, $i = 1, \ldots, \ell$, while $\mathrm{bdiag}(A_1, \ldots, A_\ell)$ is the block diagonal matrix made up by the matrices A_i, $i = 1, \ldots, \ell$. Given a sequence of vectors $\mathscr{X} = \{x(t) : t = 1, \ldots, m\}$, we denote the vertical concatenation of its vectors as $\mathrm{vertcat}(\mathscr{X})$. We also denote the cardinality of $\mathscr{X}$ as $|\mathscr{X}|$, where $|\mathscr{X}| = m$. Finally, we define the ℓ_2-norm of a vector $x \in \mathbb{R}^n$ as $\|x\|_2^2 = \sum_{i=1}^n x_i(t)^2$.

B. Single-Agent Covariance Steering

The goal of covariance steering is to find a feedback control policy that will steer the uncertain state from a given initial mean and covariance to prescribed terminal ones. Here, we show how to formulate the latter problem for linear dynamics subject to white Gaussian process noise as a convex semi-definite program.

1) Problem Formulation: Consider the following discrete-time stochastic linear system:

$$x(t+1) = A(t)x(t) + B(t)u(t) + w(t), \tag{1a}$$
$$x(0) = x_0, \quad x_0 \sim \mathcal{N}(\mu_0, \Sigma_0), \tag{1b}$$

for $t = 0, \ldots, T-1$, where $w(t) \sim \mathcal{N}(0, W_t)$ is the process noise, $\mu_0 \in \mathbb{R}^n$ and $\Sigma_0 \in \mathbb{S}_n^{++}$ are given and $W_t \in \mathbb{S}_n^+$. This system yields a random state process $\mathscr{X}_{0:t} = \{x(\tau) \in \mathbb{R}^n : \tau = 0, \ldots, t\}$, for $t = 0, \ldots, T$, that depends on the control input process $\mathscr{U}_{0:t-1} = \{u(\tau) : \tau = 0, \ldots, t-1\}$, the noise process $\mathscr{W}_{0:t-1} = \{w(\tau) : \tau = 0, \ldots, t-1\}$, and the initial (random) state x_0.

In addition, consider admissible control policies $\varpi := \{\pi(t, \cdot) : t = 0, \ldots, T-1\}$ that are affine functions of the elements of the uncertain state process:

$$\pi(t, \mathscr{X}_{0:t}) = v(t) + \sum_{\tau=0}^{t} K(t,\tau)x(\tau), \tag{2}$$

where $v(t) \in \mathbb{R}^m$ and $K(t,\tau) \in \mathbb{R}^{m\times n}$ for all $\tau = 0, \ldots, T-1$. The admissible control policies are completely defined by the sequence of vectors $\mathscr{V} = \{v(t) : t = 0, \ldots, T-1\}$ and the collection of matrix gains $\mathscr{K} = \{K(t,\tau) : t, \tau = 0, \ldots, T-1,\ t \geq \tau\}$.

Among all admissible control policies $\varpi = \varpi(\mathscr{V}, \mathscr{K})$, we seek the sequences $\mathscr{V}$ and $\mathscr{K}$ that minimize the following performance index:

$$J(\mathscr{V}, \mathscr{K}) = \sum_{t=0}^{T-1} \mathbb{E}\Big[u(t)^\top u(t)\Big] \tag{3}$$

subject to the dynamic constraints (1a) and the boundary conditions

$$\mathbb{E}[x(0)] = \mu_0, \qquad \mathrm{Cov}[x(0)] = \Sigma_0, \tag{4a}$$
$$\mathbb{E}[x(T)] = \mu_f, \qquad \big(\Sigma_f - \mathrm{Cov}[x(T)]\big) \in \mathbb{S}_n^+. \tag{4b}$$

The performance index (3) ensures that the goal will be reached without excessive actuation, while the positive semi-definite terminal constraint (4b) is a relaxation of the non-convex equality constraint $\mathrm{Cov}[x(T)] = \Sigma_f$ that sets an upper bound on the uncertainty with which the terminal mean, μ_f, will be reached [4, 6].

2) Reduction to a Convex Semi-Definite Program: The above covariance steering problem can be reduced to and solved as a convex SDP (for details, see [4, 6]). First, the discrete-time dynamics (1a) are compactly written as:

$$\boldsymbol{x} = \mathbf{G}_0 x_0 + \mathbf{G}_u \boldsymbol{u} + \mathbf{G}_w \boldsymbol{w}, \tag{5}$$

where $\boldsymbol{x} = \mathrm{vertcat}(\mathscr{X}_{0:T}) \in \mathbb{R}^{(T+1)n}$, $\boldsymbol{u} = \mathrm{vertcat}(\mathscr{U}_{0:T-1}) \in \mathbb{R}^{Tm}$, and $\boldsymbol{w} = \mathrm{vertcat}(\mathscr{W}_{0:T-1}) \in \mathbb{R}^{Tn}$. In addition, $\mathbf{G}_0$, $\mathbf{G}_u$, and $\mathbf{G}_w$ are defined as follows:

$$\mathbf{G}_0 := \begin{bmatrix} I & \Phi(1,0)^\top & \cdots & \Phi(T,0)^\top \end{bmatrix}^\top,$$

$$\mathbf{G}_u := \begin{bmatrix} 0 & 0 & \cdots & 0 \\ B(0) & 0 & \cdots & 0 \\ \Phi(2,1)B(0) & B(1) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \Phi(T,1)B(0) & \Phi(T,2)B(1) & \cdots & B(T-1) \end{bmatrix},$$

$$\mathbf{G}_w := \begin{bmatrix} 0 & 0 & \cdots & 0 \\ I & 0 & \cdots & 0 \\ \Phi(2,1) & I & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \Phi(T,1) & \Phi(T,2) & \cdots & I \end{bmatrix},$$

where $\Phi(k,m) = A(k-1)\cdots A(m)$ and $\Phi(k,k) = I$, for $k \geq m$. In view of (2), an admissible control sequence can be written as:

$$\boldsymbol{u} = \boldsymbol{v} + \mathbf{K}\boldsymbol{x}$$

where $\boldsymbol{v} := \mathrm{vertcat}(\mathscr{V})$ and

$$\mathbf{K} := \begin{bmatrix} K(0,0) & 0 & \dots & 0 & 0 \\ K(1,0) & K(1,1) & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ K(T-1,0) & K(T-1,1) & \dots & K(T-1,T-1) & 0 \end{bmatrix}.$$

In order to formulate a convex program, we need to define the decision variables $\mathbf{L}$ and $\boldsymbol{\nu}$:

$$\mathbf{L} := \mathbf{K}(I - \mathbf{G}_u\mathbf{K})^{-1}, \quad \boldsymbol{\nu} := (I + \mathbf{L}\mathbf{G}_u)\,\boldsymbol{v} \tag{6}$$

where $\mathbf{L}$ is a block lower-triangular matrix and $(I - \mathbf{G}_u\mathbf{K})$ is well defined, as explained in [4].

The performance index (3) can now be written with respect to the new decision variables as:

$$\begin{aligned} \mathcal{J}(\boldsymbol{\nu}, \mathbf{L}) := \mathrm{tr}\big(&\mathbf{L}\mathbf{G}_0(\Sigma_0 + \mu_0\mu_0^\top)\mathbf{G}_0^\top\mathbf{L}^\top \\ &+ 2\mathbf{L}\mathbf{G}_0\mu_0\boldsymbol{\nu}^\top + \boldsymbol{\nu}\boldsymbol{\nu}^\top + \mathbf{L}\mathbf{G}_w\mathbf{W}\mathbf{G}_w^\top\mathbf{L}^\top\big) \end{aligned} \tag{7}$$

where $\mathbf{W} := \mathrm{bdiag}(W_0, \ldots, W_{T-1})$. The mean and covariance of the random vector $\boldsymbol{x}$ are given by:

$$\mathbb{E}[\boldsymbol{x}] = \mathfrak{f}(\boldsymbol{\nu}, \mathbf{L}), \quad \mathrm{Cov}[\boldsymbol{x}] = \mathfrak{g}(\boldsymbol{\nu}, \mathbf{L}) \tag{8}$$

where

$$\mathfrak{f}(\boldsymbol{\nu}, \mathbf{L}) := \mathbf{T}_0\mu_0 + \mathbf{G}_u\boldsymbol{\nu}, \tag{9a}$$

$$\begin{aligned} \mathfrak{g}(\boldsymbol{\nu}, \mathbf{L}) := {}& \mathbf{T}_0(\Sigma_0 + \mu_0\mu_0^\top)\mathbf{T}_0^\top + \mathbf{T}_0\mu_0\boldsymbol{\nu}^\top\mathbf{G}_u^\top \\ &+ \mathbf{G}_u\boldsymbol{\nu}\mu_0^\top\mathbf{T}_0^\top + \mathbf{G}_u\boldsymbol{\nu}\boldsymbol{\nu}^\top\mathbf{G}_u^\top \\ &+ \mathbf{T}_w\mathbf{W}\mathbf{T}_w^\top, \end{aligned} \tag{9b}$$

with $\mathbf{T}_0 := (I + \mathbf{G}_u\mathbf{L})\mathbf{G}_0$, and $\mathbf{T}_w := (I + \mathbf{G}_u\mathbf{L})\mathbf{G}_w$.

In addition, $x(t)$ can be extracted from $\boldsymbol{x}$ with $x(t) = \mathbf{P}_{t+1}\boldsymbol{x}$ where $\mathbf{P}_{t+1} \in \mathbb{R}^{n\times(T+1)n}$ is a block matrix whose blocks are all equal to the zero matrix except from the $(t+1)$-th one which is equal to the $n \times n$ identity matrix.

As a result, the terminal mean and covariance constraints can be expressed as:

$$\mathcal{F}(\boldsymbol{\nu}, \mathbf{L}) := \mathbf{P}_T\mathfrak{f}(\boldsymbol{\nu}, \mathbf{L}) - \mu_f = 0 \tag{10a}$$

$$\mathcal{G}(\boldsymbol{\nu}, \mathbf{L}) := \Sigma_f + \mu_f\mu_f^\top - \mathbf{P}_T\mathfrak{g}(\boldsymbol{\nu}, \mathbf{L})\mathbf{P}_T^\top \in \mathbb{S}_n^+ \tag{10b}$$

Now, the covariance steering problem can be formulated as follows: find a pair $(\boldsymbol{\nu}, \mathbf{L})$ that minimizes the performance index (7) subject to the equality constraint (10a) and the semi-definite constraint (10b). The above is a convex semi-definite program and can be solved efficiently using any available conic solver. Note that the semi-definite constraint (10b) can be converted to a linear matrix inequality (LMI) convex constraint using the Schur complement.

The variables of interest, namely $\boldsymbol{v}$ and $\mathbf{K}$, can be computed from the optimal solution $(\boldsymbol{\nu}^*, \mathbf{L}^*)$ by inverting the transformation (6):

$$\mathbf{K} := (I + \mathbf{L}\mathbf{G}_u)^{-1}\mathbf{L}, \quad \boldsymbol{v} := (I + \mathbf{L}\mathbf{G}_u)^{-1}\boldsymbol{\nu} \tag{11}$$

where $(I + \mathbf{L}\mathbf{G}_u)$ is also well defined, i.e. invertible.

C. Alternating Direction Method of Multipliers (ADMM)

1) Classical ADMM: In the standard ADMM formulation [9], we have a separable objective function with respect to two variables $\boldsymbol{x} \in \mathbb{R}^n$ and $\boldsymbol{z} \in \mathbb{R}^m$ and a linear coupling constraint between them. The optimization problem has the following form:

$$\min_{\boldsymbol{x},\boldsymbol{z}} \; f(\boldsymbol{x}) + g(\boldsymbol{z}) \quad \text{s.t. } \mathbf{A}\boldsymbol{x} + \mathbf{B}\boldsymbol{z} = \boldsymbol{c} \tag{12}$$

where $f : \mathbb{R}^n \to \mathbb{R}$, $g : \mathbb{R}^m \to \mathbb{R}$, $\mathbf{A} \in \mathbb{R}^{p\times n}$, $\mathbf{B} \in \mathbb{R}^{p\times m}$ and $\boldsymbol{c} \in \mathbb{R}^p$. The augmented Lagrangian (AL) for problem (12) is:

$$\begin{aligned} \mathcal{L}_\rho(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{y}) = {}& f(\boldsymbol{x}) + g(\boldsymbol{z}) + \boldsymbol{y}^\top(\mathbf{A}\boldsymbol{x} + \mathbf{B}\boldsymbol{z} - \boldsymbol{c}) \\ &+ \frac{\rho}{2}\|\mathbf{A}\boldsymbol{x} + \mathbf{B}\boldsymbol{z} - \boldsymbol{c}\|_2^2 \end{aligned}$$

where $\boldsymbol{y} \in \mathbb{R}^p$ is the dual variable and $\rho > 0$ is the penalty parameter. The classical ADMM algorithm consists of the following sequential updates for the primal and dual variables:

$$\begin{aligned} \boldsymbol{x}^{k+1} &:= \arg\min_{\boldsymbol{x}} \mathcal{L}_\rho(\boldsymbol{x}, \boldsymbol{z}^k, \boldsymbol{y}^k) \\ \boldsymbol{z}^{k+1} &:= \arg\min_{\boldsymbol{z}} \mathcal{L}_\rho(\boldsymbol{x}^{k+1}, \boldsymbol{z}, \boldsymbol{y}^k) \\ \boldsymbol{y}^{k+1} &:= \boldsymbol{y}^k + \rho(\mathbf{A}\boldsymbol{x}^{k+1} + \mathbf{B}\boldsymbol{z}^{k+1} - \boldsymbol{c}). \end{aligned}$$

The problem formulation (12) can be extended to a multi-block variation with N optimization variables $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$, obtaining the following form:

$$\min_{\boldsymbol{x}_1,\ldots,\boldsymbol{x}_N} \sum_{i=1}^{N} f_i(\boldsymbol{x}_i) \quad \text{s.t.} \quad \sum_{i=1}^{N} \mathbf{A}_i \boldsymbol{x}_i = \boldsymbol{b} \tag{13}$$

with $\boldsymbol{x}_i \in \mathbb{R}^{n_i}$, $f_i : \mathbb{R}^{n_i} \rightarrow \mathbb{R}$, $\mathbf{A}_i \in \mathbb{R}^{m \times n_i}$ and $\boldsymbol{b} \in \mathbb{R}^m$. The resulting multi-block ADMM algorithm consists of the following sequential updates:

$$\boldsymbol{x}_1^{k+1} := \arg\min_{\boldsymbol{x}_1} \mathcal{L}_\rho(\boldsymbol{x}_1, \boldsymbol{x}_2^k, \ldots, \boldsymbol{x}_N^k, \boldsymbol{y}^k)$$
$$\ldots$$
$$\boldsymbol{x}_N^{k+1} := \arg\min_{\boldsymbol{x}_N} \mathcal{L}_\rho(\boldsymbol{x}_1^{k+1}, \ldots, \boldsymbol{x}_{N-1}^{k+1}, \boldsymbol{x}_N, \boldsymbol{y}^k)$$
$$\boldsymbol{y}^{k+1} := \boldsymbol{y}^k + \frac{\rho}{2}\big(\sum_{i=1}^{N} \mathbf{A}_i \boldsymbol{x}_i^{k+1} - \boldsymbol{b}\big).$$

2) Consensus ADMM: Next, we present an ADMM variation for general form consensus optimization problems [9]. Let us consider a set of local optimization variables $\boldsymbol{x}_i \in \mathbb{R}^{n_i}$, $i = 1, \ldots, N$ and the following problem where the objective function is separable with respect to $\boldsymbol{x}_i$:

$$\min_{\boldsymbol{x}_1,\ldots,\boldsymbol{x}_N} \sum_{i=1}^{N} f_i(\boldsymbol{x}_i) \tag{14}$$

where $f_i : \mathbb{R}^{n_i} \rightarrow \mathbb{R} \cup \{+\infty\}$. Constraints can be incorporated in each term by assigning the value $f_i(\boldsymbol{x}_i) = +\infty$ when a constraint is violated. Let us also define the global variable $\boldsymbol{z} \in \mathbb{R}^n$ with $n \geq n_i, \forall i = 1, \ldots, N$. Each local variable $\boldsymbol{x}_i$ corresponds to a specific selection of components of the global variable $\boldsymbol{z}$. In other words, each local variable component $(\boldsymbol{x}_i)_j$ corresponds to a component $\boldsymbol{z}_g$ of $\boldsymbol{z}$. By expressing this linking with the mapping $g = G(i, j)$, the occurring consensus constraints can be written as:

$$(\boldsymbol{x}_i)_j = \boldsymbol{z}_{G(i,j)}, \quad j = 1, \ldots, n_i, \quad i = 1, \ldots, N.$$

Let us now also define $\tilde{\boldsymbol{z}}_i \in R^{n_i}$, with $(\tilde{\boldsymbol{z}}_i)_j = \boldsymbol{z}_{G(i,j)}$. Essentially, $\tilde{\boldsymbol{z}}_i$ expresses the global variable's idea of what the local variable $\boldsymbol{x}_i$ should be. The optimization problem can now be expressed in the following general consensus form:

$$\min_{\boldsymbol{x}_1,\ldots,\boldsymbol{x}_N} \sum_{i=1}^{N} f_i(\boldsymbol{x}_i) \tag{15a}$$
$$\text{s.t. } \boldsymbol{x}_i - \tilde{\boldsymbol{z}}_i = 0, \; i = 1, \ldots, N. \tag{15b}$$

By formulating the AL for (15), the following ADMM algorithm can be derived:

$$\boldsymbol{x}_i^{k+1} := \arg\min_{\boldsymbol{x}_i} \Big(f_i(\boldsymbol{x}_i) + \boldsymbol{y}_i^{k^\top} \boldsymbol{x}_i + \frac{\rho}{2} \|\boldsymbol{x}_i - \tilde{\boldsymbol{z}}_i^k\|_2^2 \Big) \tag{16a}$$
$$\boldsymbol{z}_g^{k+1} := \frac{1}{k_g} \sum_{g=G(i,j)} \Big((\boldsymbol{x}_i^{k+1})_j + \frac{1}{\rho} (\boldsymbol{y}_i^k)_j \Big) \tag{16b}$$
$$\boldsymbol{y}_i^{k+1} := \boldsymbol{y}_i^k + \rho(\boldsymbol{x}_i^{k+1} - \tilde{\boldsymbol{z}}_i^{k+1}) \tag{16c}$$

where k_g is the number of local variable entries that correspond to the global variable entry $\boldsymbol{z}_g$. Note that the primal (16a) and dual (16c) updates can be carried out in parallel by each processing element i.

III. Multi-agent covariance steering problem formulation

In this section, we present our formulation for the multi-agent covariance control problem. Our goal is to steer a team of agents under stochastic linear dynamics to prescribed final state means and covariances while minimizing their control effort and taking into account additional inter-agent constraints.

A. Problem Setup

We consider a team of N agents. Each agent $i = 1, \ldots, N$ has a set of neighbors N_i. We denote the number of neighbors of each agent with m_i, i.e. $|N_i| = m_i$. We can allow for each agent to have a different number of neighbors. Therefore, our approach is flexible in terms of how one defines each agent's vicinity. Moreover, if agent i considers agent j as a neighbor, it is not necessary that agent j should consider i as its neighbor as well. In other words, we do not require that $j \in N_i \Leftrightarrow i \in N_j$. Here, we assume that the sets N_i, $i = 1, \ldots, N$ are fixed throughout the time horizon $t = 0, \ldots, T$. Further, we define the set of agents that contain agent i as a neighbor with $P_i = \{j : i \in N_j\}$. In terms of communication requirements, we only assume that each agent should be able to communicate with the agents belonging in $N_i \cup P_i$.

B. Agent Dynamics, Cost and Constraints

Each agent i is subject to the following discrete-time linear stochastic dynamics:

$$x_i(t+1) = A_i(t)x_i(t) + B_i(t)u_i(t) + w_i(t), \tag{17a}$$
$$x_i(0) = x_{i,0}, \quad x_{i,0} \sim \mathcal{N}(\mu_{i,0}, \Sigma_{i,0}). \tag{17b}$$

Our goal is to propagate the team of agents from a set of initial state Gaussian distributions $x_{i,0} \sim \mathcal{N}(\mu_{i,0}, \Sigma_{i,0})$, $i = 1, \ldots, N$ to a set of prescribed final ones $x_{i,f} \sim \mathcal{N}(\mu_{i,f}, \Sigma_{i,f})$, $i = 1, \ldots, N$ while minimizing the expected value of the control effort of each agent. Therefore, similar to (3), the objective function of each agent is formulated as:

$$J_i(\mathscr{V}_i, \mathscr{K}_i) = \sum_{t=0}^{T-1} \mathbb{E}\Big[u_i(t)^\top u_i(t)\Big]. \tag{18}$$

The terminal constraints on the final state mean and covariance of each agent are also expressed similar to (4b) as:

$$\mathbb{E}[x_i(T)] = \mu_{i,f}, \quad \big(\Sigma_{i,f} - \mathrm{Cov}[x_i(T)]\big) \in \mathbb{S}_n^+. \tag{19}$$

In a multi-agent control setting, it is critical to also account for inter-agent constraints. We can incorporate a variety of constraints that can include the mean and the covariance of the state of each agent. In general, we formulate the inter-agent constraints between a pair of agents (i, j) as:

$$D_{ij}(\mathbb{E}[x_i(t)], \mathbb{E}[x_j(t)], \mathrm{Cov}[x_i(t)], \mathrm{Cov}[x_j(t)]) \leq 0. \tag{20}$$

One example could be collision avoidance constraints between the mean positions of the agents. Alternatively, one could place a lower bound on the expected value of the distance (which depends on both the means and the covariances) between the agents. Another inter-agent constraint could be to enforce a team of agents to remain in a circle of a given radius with a virtual agent center, in order to maintain communication connectivity.

C. Centralized Multi-Agent Covariance Steering Problem

We will now formulate the multi-agent covariance steering problem in its centralized version. The problem can be expressed as follows: compute the control policy variables $(\mathscr{V}_i, \mathscr{K}_i)$ for all agents $i = 1, \dots, N$ that solve:

$$\min \sum_{i=1}^{N} J_i(\mathscr{V}_i, \mathscr{K}_i) = \sum_{i=1}^{N} \sum_{t=0}^{T-1} \mathbb{E}\Big[u_i(t)^\top u_i(t)\Big] \tag{21a}$$

$$\text{s.t.} \quad \mathbb{E}[x_i(T)] = \mu_{i,f}, \quad i = 1, \dots, N \tag{21b}$$

$$\big(\Sigma_{i,f} - \mathrm{Cov}[x_i(T)]\big) \in \mathbb{S}_n^+, \quad i = 1, \dots, N \tag{21c}$$

$$D_{ij}(\mathbb{E}[x_i(t)], \mathbb{E}[x_j(t)], \mathrm{Cov}[x_i(t)], \mathrm{Cov}[x_j(t)]) \leq 0$$
$$j \in N_i, \ i = 1, \dots, N. \tag{21d}$$

Note that the inter-agent constraints only involve the neighbors of each agent i.

IV. Distributed Covariance Steering with Consensus ADMM

In this section, we propose our new distributed covariance steering method based on ADMM for consensus optimization.

A. Transformation to General Consensus Problem

In order for each agent to be capable of handling inter-agent constraints, it will need to also contain information about its neighbors $j \in N_i$. Therefore, for each agent i, we define the augmented state and control vectors $\tilde{x}_i(t)$ and $\tilde{u}_i(t)$ as:

$$\tilde{x}_i(t) = \mathrm{vertcat}\big(x_i(t), \{x_j^{(i)}(t)\}_{j \in N_i}\big) \tag{22a}$$

$$\tilde{u}_i(t) = \mathrm{vertcat}\big(u_i(t), \{u_j^{(i)}(t)\}_{j \in N_i}\big) \tag{22b}$$

where $x_j^{(i)}(t)$ and $u_j^{(i)}(t)$ are copy variables computed by agent i for agent j. In the distributed control setting, the actual variables $x_j(t), u_j(t)$ of agent j will be different from the copy variables of agent i concerning agent j, since the control executed by agent j is based on its own neighbors, measurements and control computations. The inter-agent constraints (21d) can now be written from each agent's perspective as:

$$D_{ij}(\mathbb{E}[x_i(t)], \mathbb{E}[x_j^{(i)}(t)], \mathrm{Cov}[x_i(t)], \mathrm{Cov}[x_j^{(i)}(t)]) \leq 0$$
$$j \in N_i, \ i = 1, \dots, N$$

or more compactly with respect to the local variables $\tilde{x}_i(t)$ as:

$$D_i(\mathbb{E}[\tilde{x}_i(t)], \mathrm{Cov}[\tilde{x}_i(t)]) \leq 0, \quad i = 1, \dots, N. \tag{23}$$

The inclusion of the copy variables makes it necessary to enforce a consensus between the copy variables that each agent uses for its neighbors and the original variables of its neighbors. If we do not enforce this consensus, then two neighboring agents may compute control policies that are not in agreement with each other, causing significant difficulties. For instance, imagine a multi-vehicle scenario where agent i has decided for itself to move and for agent j to stop, while agent j has decided for itself to move and for its neighbor i to stop. In the absence of consensus, this would result to an undesirable collision. Such issues can be resolved by incorporating consensus constraints.

Our proposed strategy is to enforce this consensus through the control policy decision variables (6) of each agent. We also define the augmented decision variables:

$$\tilde{\boldsymbol{\nu}}_i = \mathrm{vertcat}\big(\boldsymbol{\nu}_i, \{\boldsymbol{\nu}_j^{(i)}\}_{j \in N_i}\big), \tag{24a}$$

$$\tilde{\mathbf{L}}_i = \mathrm{vertcat}\big(\mathbf{L}_i, \{\mathbf{L}_j^{(i)}\}_{j \in N_i}\big) \tag{24b}$$

where $(\boldsymbol{\nu}_i, \mathbf{L}_i)$ correspond to the actual policy that agent i will apply for itself, while $(\boldsymbol{\nu}_j^{(i)}, \mathbf{L}_j^{(i)})$ are copy decision variables calculated by agent i for its neighbors so that the inter-agent constraints will be satisfied. Note that through the expressions (8), the inter-agent constraints (23) can now be written as:

$$\mathcal{D}_i(\tilde{\boldsymbol{\nu}}_i, \tilde{\mathbf{L}}_i) \leq 0, \quad i = 1, \dots, N. \tag{25}$$

Let us now define the global variables $\boldsymbol{z}$ and $\mathbf{W}$ which contain the decision variables $\boldsymbol{\nu}_i$ and $\mathbf{L}_i$, respectively, of all N agents:

$$\boldsymbol{z} = \mathrm{vertcat}\big(\boldsymbol{\nu}_1, \dots, \boldsymbol{\nu}_N\big), \quad \mathbf{W} = \mathrm{vertcat}\big(\mathbf{L}_1, \dots, \mathbf{L}_N\big).$$

Each local variable $\tilde{\boldsymbol{\nu}}_i$ and $\tilde{\mathbf{L}}_i$ will consist of a selection of the components of the global variables $\boldsymbol{z}$ and $\mathbf{W}$, respectively. Let $g = G(i,j)$ be the mapping from local variable indices $i = 1, \dots, N$, $j \in \{i \cup N_i\}$ to a global variable index $g = 1, \dots, N$, i.e. the local variable component $\boldsymbol{\nu}_j^{(i)}$ (or $\mathbf{L}_j^{(i)}$) corresponds to the global variable component $\boldsymbol{z}_g$ (or $\mathbf{W}_g$). Therefore, the consensus constraints can be expressed as:

$$\boldsymbol{\nu}_j^{(i)} = \boldsymbol{z}_{G(i,j)}, \quad j \in \{i \cup N_i\}, \ i = 1, \dots, N \tag{26a}$$

$$\mathbf{L}_j^{(i)} = \mathbf{W}_{G(i,j)}, \quad j \in \{i \cup N_i\}, \ i = 1, \dots, N. \tag{26b}$$

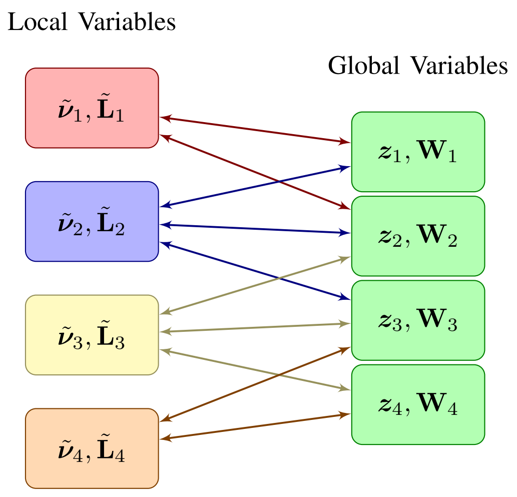
By using the notation (24), the consensus constraints (26) can be written in a more compact form as:

$$\tilde{\boldsymbol{\nu}}_i - \tilde{\boldsymbol{z}}_i = 0, \quad i = 1, \dots, N \tag{27a}$$

$$\tilde{\mathbf{L}}_i - \tilde{\mathbf{W}}_i = 0, \quad i = 1, \dots, N \tag{27b}$$

where $\tilde{\boldsymbol{z}}_i$ and $\tilde{\mathbf{W}}_i$ are defined with $(\tilde{\boldsymbol{z}}_i)_j = \boldsymbol{z}_{G(i,j)}$ and $(\tilde{\mathbf{W}}_i)_j = \mathbf{W}_{G(i,j)}$ respectively. The connection between the local and global variable components is illustrated in Fig. 2 for a particular 4-agent example.

Therefore, we can now reformulate the multi-agent covariance steering problem (21) to a general consensus optimization problem where we wish to compute the optimal local agent



Local Variables
Global Variables
$\tilde{\boldsymbol{\nu}}_1, \tilde{\mathbf{L}}_1$
$\tilde{\boldsymbol{\nu}}_2, \tilde{\mathbf{L}}_2$
$\tilde{\boldsymbol{\nu}}_3, \tilde{\mathbf{L}}_3$
$\tilde{\boldsymbol{\nu}}_4, \tilde{\mathbf{L}}_4$
$\boldsymbol{z}_1, \mathbf{W}_1$
$\boldsymbol{z}_2, \mathbf{W}_2$
$\boldsymbol{z}_3, \mathbf{W}_3$
$\boldsymbol{z}_4, \mathbf{W}_4$

Fig. 2: The connections (consensus constraints) between the local variable of each agent and the global variable components. In this figure, we demonstrate a 4-agent example with $N_1 = \{2\}$, $N_2 = \{1, 3\}$, $N_3 = \{2, 4\}$, $N_4 = \{3\}$.

variables $\tilde{\boldsymbol{\nu}}_i$ and $\tilde{\mathbf{L}}_i$ such that:

$$\{\tilde{\boldsymbol{\nu}}_i, \tilde{\mathbf{L}}_i\}_{i=1,\ldots,N} = \arg\min \sum_{i=1}^{N} \mathcal{J}_i(\boldsymbol{\nu}_i, \mathbf{L}_i) \tag{28a}$$

$$\text{s.t.} \quad \mathcal{F}_i(\boldsymbol{\nu}_i, \mathbf{L}_i) = 0, \tag{28b}$$

$$\mathcal{G}_i(\boldsymbol{\nu}_i, \mathbf{L}_i) \in \mathbb{S}_n^+, \tag{28c}$$

$$\mathcal{D}_i(\tilde{\boldsymbol{\nu}}_i, \tilde{\mathbf{L}}_i) \leq 0, \tag{28d}$$

$$\tilde{\boldsymbol{\nu}}_i - \tilde{\boldsymbol{z}}_i = 0, \ \tilde{\mathbf{L}}_i - \tilde{\mathbf{W}}_i = 0, \tag{28e}$$

$$i = 1, \ldots, N$$

where for (28a), (28b) and (28c), we are using the notation introduced through (7) and (10). By defining the indicator functions $\mathcal{I}_{\mathcal{F}_i}(\boldsymbol{\nu}_i, \mathbf{L}_i)$, $\mathcal{I}_{\mathcal{G}_i}(\boldsymbol{\nu}_i, \mathbf{L}_i)$ and $\mathcal{I}_{\mathcal{D}_i}(\tilde{\boldsymbol{\nu}}_i, \tilde{\mathbf{L}}_i)$ which take a zero value if the constraints (28b), (28c) and (28d) respectively are satisfied and an infinite value if they are violated, and by formulating the new objective function:

$$\hat{\mathcal{J}}_i(\tilde{\boldsymbol{\nu}}_i, \tilde{\mathbf{L}}_i) = \mathcal{J}_i(\boldsymbol{\nu}_i, \mathbf{L}_i) + \mathcal{I}_{\mathcal{F}_i}(\boldsymbol{\nu}_i, \mathbf{L}_i) + \mathcal{I}_{\mathcal{G}_i}(\boldsymbol{\nu}_i, \mathbf{L}_i) + \mathcal{I}_{\mathcal{D}_i}(\tilde{\boldsymbol{\nu}}_i, \tilde{\mathbf{L}}_i)$$

the problem (28) can finally be written as:

$$\{\tilde{\boldsymbol{\nu}}_i, \tilde{\mathbf{L}}_i\}_{i=1,\ldots,N} = \arg\min \sum_{i=1}^{N} \hat{\mathcal{J}}_i(\tilde{\boldsymbol{\nu}}_i, \tilde{\mathbf{L}}_i) \tag{29a}$$

$$\text{s.t.} \quad \tilde{\boldsymbol{\nu}}_i - \tilde{\boldsymbol{z}}_i = 0, \ \tilde{\mathbf{L}}_i - \tilde{\mathbf{W}}_i = 0, \quad i = 1, \ldots, N. \tag{29b}$$

Consequently, we have now transformed the multi-agent covariance steering problem to a general consensus optimization problem form (15).

B. Distributed Covariance Steering with ADMM

After transforming the initial problem to a consensus optimization one, we can proceed with deriving an ADMM distributed algorithm for solving it. The AL for (29) is formulated as follows:

$$\mathcal{L}_\rho = \sum_{i=1}^{N} \Big[\hat{\mathcal{J}}_i(\tilde{\boldsymbol{\nu}}_i, \tilde{\mathbf{L}}_i) + \boldsymbol{\lambda}_i^\top (\tilde{\boldsymbol{\nu}}_i - \tilde{\boldsymbol{z}}_i) + \mathrm{tr}\big(\mathbf{M}_i^\top (\tilde{\mathbf{L}}_i - \tilde{\mathbf{W}}_i)\big) + \frac{\rho}{2}\|\tilde{\boldsymbol{\nu}}_i - \tilde{\boldsymbol{z}}_i\|_2^2 + \frac{\mu}{2}\|\tilde{\mathbf{L}}_i - \tilde{\mathbf{W}}_i\|_2^2 \Big]$$

where $\boldsymbol{\lambda}_i$ and $\mathbf{M}_i$ are the dual variables that correspond to the consensus constraints (27a) and (27b), respectively, and ρ and μ are the penalty parameters. Given the AL, we can now derive the following ADMM algorithm consisting of the updates:

$$\{\tilde{\boldsymbol{\nu}}_i, \tilde{\mathbf{L}}_i\}^{l+1} = \arg\min_{\tilde{\boldsymbol{\nu}}_i, \tilde{\mathbf{L}}_i} \Big(\hat{\mathcal{J}}_i(\tilde{\boldsymbol{\nu}}_i, \tilde{\mathbf{L}}_i) + \boldsymbol{\lambda}_i^{l^\top} \tilde{\boldsymbol{\nu}}_i + \mathrm{tr}(\mathbf{M}_i^{l^\top} \tilde{\mathbf{L}}_i) + \frac{\rho}{2}\|\tilde{\boldsymbol{\nu}}_i - \tilde{\boldsymbol{z}}_i^l\|_2^2 + \frac{\mu}{2}\|\tilde{\mathbf{L}}_i - \tilde{\mathbf{W}}_i^l\|_2^2 \Big) \tag{30a}$$

$$\boldsymbol{z}_g^{l+1} = \frac{1}{k_g} \sum_{g=G(i,j)} \Big((\tilde{\boldsymbol{\nu}}_i^{l+1})_j + \frac{1}{\rho}(\boldsymbol{\lambda}_i^l)_j \Big) \tag{30b}$$

$$\mathbf{W}_g^{l+1} = \frac{1}{k_g} \sum_{g=G(i,j)} \Big((\tilde{\mathbf{L}}_i^{l+1})_j + \frac{1}{\mu}(\mathbf{M}_i^l)_j \Big) \tag{30c}$$

$$\boldsymbol{\lambda}_i^{l+1} = \boldsymbol{\lambda}_i^l + \rho(\tilde{\boldsymbol{\nu}}_i^{l+1} - \tilde{\boldsymbol{z}}_i^{l+1}) \tag{30d}$$

$$\mathbf{M}_i^{l+1} = \mathbf{M}_i^l + \mu(\tilde{\mathbf{L}}_i^{l+1} - \tilde{\mathbf{W}}_i^{l+1}) \tag{30e}$$

where l indicates the iteration number. Clearly, the primal (30a) and dual (30d-30e) updates can be executed in parallel and independently by each agent i. In addition, based on the assumption that each agent can communicate with the agents belonging in $N_i \cup P_i$, then it can also perform the global variable component updates (30b-30c) that correspond to itself, i.e. $\boldsymbol{z}_i$ and $\mathbf{W}_i$, independently. To achieve this, after the primal updates, each agent that belongs in P_i must send its local (primal and dual) variables to agent i. After the global updates, the neighbors of agent i, i.e. the ones that belong in N_i, must send the global variable components that they just computed to agent i so that it can construct the variables $\tilde{\boldsymbol{z}}_i$ and $\tilde{\boldsymbol{W}}_i$. This parallelizability of the updates (30a-30e) characterizes the distributed nature of our method. Furthermore, to warmstart the algorithm, we initialize the global variable components by solving the single-agent covariance steering problems of every agent - without copy variables and inter-agent constraints - once before initiating the ADMM updates. The dual variables can be initialized with zero values. The primal update of each agent i can also be written as:

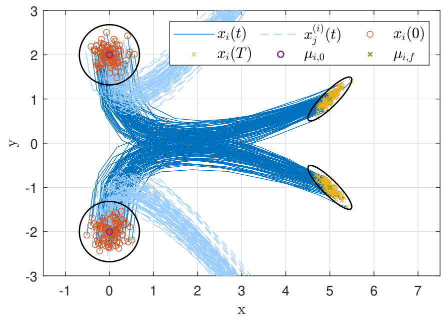
$$\{\tilde{\boldsymbol{\nu}}_i, \tilde{\mathbf{L}}_i\} = \arg\min \Big(\mathcal{J}_i(\boldsymbol{\nu}_i, \mathbf{L}_i) + \boldsymbol{\lambda}_i^{l^\top} \tilde{\boldsymbol{\nu}}_i + \mathbf{M}_i^{l^\top} \tilde{\mathbf{L}}_i + \frac{\rho}{2}\|\tilde{\boldsymbol{\nu}}_i - \tilde{\boldsymbol{z}}_i^l\|_2^2 + \frac{\mu}{2}\|\tilde{\mathbf{L}}_i - \tilde{\mathbf{W}}_i^l\|_2^2 \Big) \tag{31a}$$

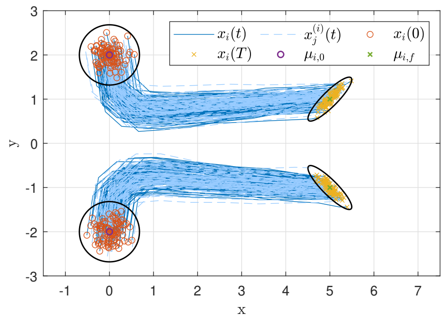
$$\text{s.t.} \quad \mathcal{F}_i(\boldsymbol{\nu}_i, \mathbf{L}_i) = 0, \tag{31b}$$

$$\mathcal{G}_i(\boldsymbol{\nu}_i, \mathbf{L}_i) \in \mathbb{S}_n^+, \tag{31c}$$

$$\mathcal{D}_i(\tilde{\boldsymbol{\nu}}_i, \tilde{\mathbf{L}}_i) \leq 0. \tag{31d}$$

This local optimization problem that each agent has to solve can be seen as the single-agent covariance steering problem that was initially introduced in Section II but with three fundamental modifications. First, each agent now also optimizes for


(a)


(b)

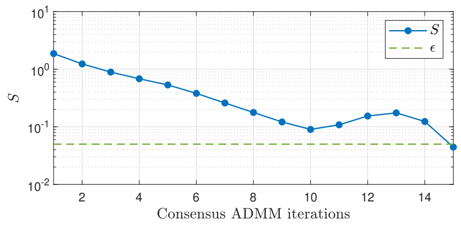
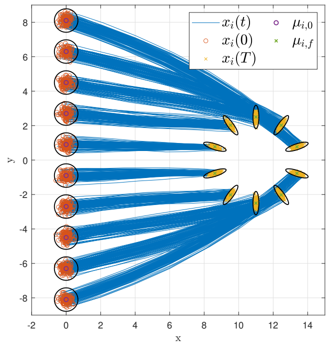
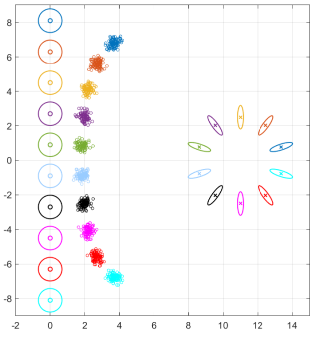
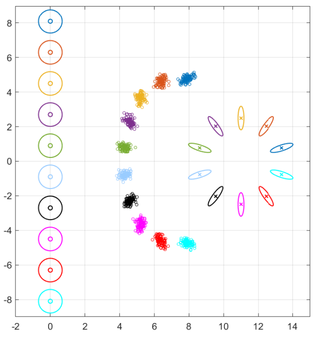
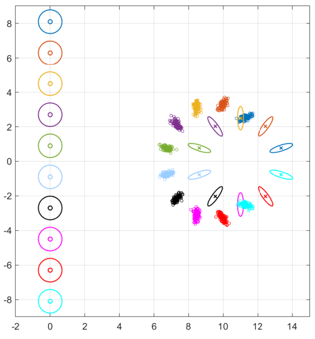

(c)

Fig. 3: Case 1. A two-agent scenario that demonstrates the effectiveness of the inter-agent collision avoidance constraints. a) After 1st ADMM iteration. b) After 31st ADMM iteration. c) Sum of consensus constraints residual norms.


(a)


(b)


(c)


(d)

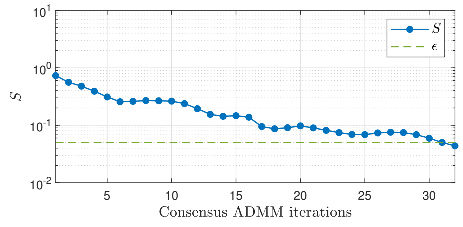

(e)

Fig. 4: Case 2. A team of 10 agents reaches a particular circle formation specified by the prescribed terminal state distributions. a) The trajectories of the distribution of each agent. b-d) Snapshots of the distributions of the agents at time instants $t = 4, 6, 8$, respectively. e) Sum of consensus constraints residual norms.

the copy variables it contains regarding its neighbors. Second, through these copy variables, the agent can account for the satisfaction of inter-agent constraints. Finally, the performance index now also contains some additional terms for reaching consensus with the other agents. Moreover, if the constraint (31d) is convex, then the problem (31) remains a convex SDP.

After the first ADMM iteration, the global updates (30b) and (30c) can be further simplified [9] to the following form:

$$\boldsymbol{z}_g^{l+1} = \frac{1}{k_g} \sum_{g=G(i,j)} (\tilde{\boldsymbol{\nu}}_i^{l+1})_j \tag{32a}$$

$$\mathbf{W}_g^{l+1} = \frac{1}{k_g} \sum_{g=G(i,j)} (\tilde{\mathbf{L}}_i^{l+1})_j. \tag{32b}$$

Intuitively, one can interpret the global updates as an averaging between the local variable components that correspond to $\boldsymbol{z}_g$ and $\mathbf{W}_g$. The algorithm terminates when the sum of the consensus constraints residual norms divided by the number of all local variable components - so that we get a normalized criterion irrespective of the number of agents and neighbors - gets below a prespecified threshold ϵ:

$$S = \frac{\sum_{i=1}^{N} \left(\|\tilde{\boldsymbol{\nu}}_i - \tilde{\boldsymbol{z}}_i\|_2^2 + \|\tilde{\mathbf{L}}_i - \tilde{\mathbf{W}}_i\|_2^2 \right)}{\sum_{i=1}^{N} (m_i + 1)} \leq \epsilon. \tag{33}$$

The satisfaction of this criterion indicates that the agents have reached a sufficient consensus.

V. Simulation Results

We apply our distributed covariance steering method on a multi-vehicle scenario, where the goal is to propagate each vehicle from an initial state Gaussian distribution to a prescribed one. The vehicles operate under double integrator dynamics on a plane with $x_i(t) = \begin{bmatrix} \mathrm{x}_i(t) & \mathrm{y}_i(t) & \dot{\mathrm{x}}_i(t) & \dot{\mathrm{y}}_i(t) \end{bmatrix}^\top$, where $\mathrm{x}_i(t), \mathrm{y}_i(t)$ are the coordinates of the i-th agent at time step t. Inter-agent collision avoidance constraints between the mean positions of the vehicles are also imposed. Note that this type of constraints will be non-convex, but we can overcome this issue by taking a first-order Taylor approximation around the previous mean state trajectories at each ADMM iteration as in [24]. In particular, for the local problem of each agent, the linearization is performed about the mean state trajectories that occur from its actual and copy control policy variables regarding its neighbors.

We examine two cases. In the first one, we present a scenario with two agents reaching a goal distribution while the anti-collision constraints are guaranteed to be satisfied after a sufficient number of ADMM iterations. In the second one, we show a formation example with 10 agents. In all examples, the time horizon is $T = 10$ and the process noise covariance is $W = \mathrm{diag}(0.02, 0.02, 0.2, 0.2)^2$. The minimum allowed distance between the mean positions of the agents is $d = 1.5$. The AL penalty parameters are tuned to $\rho, \mu = 10^3$ and the stopping criterion tolerance is assigned to be $\epsilon = 0.05$.

Case 1: In the first case, the two agents start from initial state distributions with means $\mu_{1,0} = \begin{bmatrix} 0 & 2 & 0 & -25 \end{bmatrix}^\top$, $\mu_{2,0} = \begin{bmatrix} 0 & -2 & 0 & 25 \end{bmatrix}^\top$ and covariances $\Sigma_{1,0} = \Sigma_{2,0} = \mathrm{diag}(0.2, 0.2, 0.5, 0.5)^2$. These initial mean velocities will drive the agents into a collision course that our method will have to handle. The prescribed terminal means and covariances are $\mu_{1,f} = \begin{bmatrix} 5 & 1 & 0 & 0 \end{bmatrix}^\top$, $\mu_{2,f} = \begin{bmatrix} 5 & -1 & 0 & 0 \end{bmatrix}^\top$, $\Sigma_{1,f} = R(-\pi/4)\mathrm{diag}(0.2, 0.05, 0.5, 0.5)^2 R(-\pi/4)^\top$ and $\Sigma_{2,f} = R(\pi/4)\mathrm{diag}(0.2, 0.05, 0.5, 0.5)^2 R(\pi/4)^\top$ where $R(\theta)$ is the rotation matrix. Figure 3a shows 100 realizations of the trajectory of each agent after the first ADMM iteration where consensus has not been reached, resulting to a high probability of collision. We also demonstrate the trajectories that would occur for each agent if the copy control policy variables of its neighbor were applied on it. After the first iteration, each agent has computed their actual and copy variables so that they would satisfy the collision avoidance constraints (from its perspective), while solving its own covariance steering problem and not taking into account the one of its neighbor. It is through the soft consensus constraints that the actual and copy control policies concerning the same agent will start converging to the same one during the following ADMM iterations. In Fig. 3b, the agents have reached consensus after 15 ADMM iterations and successfully handle the collision avoidance constraints. In each figure, the 99.7% confidence regions of the initial and target distributions are displayed with a black ellipsoid.

Case 2: Next, we consider a team of 10 agents that start from the initial position distributions shown in Fig. 4a with zero mean initial velocities. More specifically, the initial state covariances are $\Sigma_{i,0} = \mathrm{diag}(0.2, 0.2, 0.5, 0.5)^2, \forall i = 1, \dots, N$ and the final target state covariances are $\Sigma_{i,f} = R(-\phi_i)\mathrm{diag}(0.2, 0.05, 0.5, 0.5)^2 R(-\phi_i), \forall i = 1, \dots, N$ where $\phi_i = (2i-1)2\pi/N$. Each agent has $m_i = 4$ neighbors. As demonstrated, the agents reach the desired circle formation defined by the prescribed terminal distributions. Note that as in Case 1, the terminal distributions are within the prescribed ones, placing an upper bound on the uncertainty of the final state. Moreover, as shown in Figs. 4b-d, where some snapshots of the distributions are provided for $t = 4, 6, 8$, the agents achieve the formation while successfully avoiding collisions.

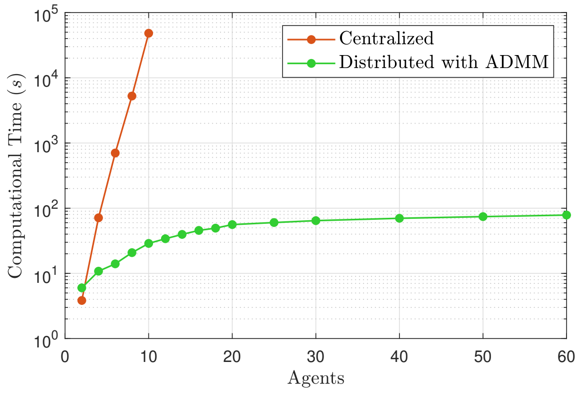


Fig. 5: Comparison of computational times between our distributed and a centralized approach.

Scalability to large-scale systems: Next, we demonstrate the applicability of our method on large-scale stochastic multi-agent systems in terms of computational demands. In particular, we repeat Case 2 with a varying number of agents and we compare the computational times of our distributed method and an equivalent centralized approach. For $N \leq 20$, we suppose that $m_i = N/2$, while for $N > 20$, we fix the number of neighbors to $m_i = 10$. The computational times for both approaches are shown in Fig. 5. The simulations were performed in Matlab R2020b using CVX [14] as the modeling software, MOSEK 9.1.9 [1] as the solver and an Intel Core i5-8279U CPU @ 2.40GHz. The increased computational efficiency of our ADMM-based approach is mainly due to the fact that each agent solves in parallel a local covariance steering (SDP) problem where the terminal semidefinite covariance constraint only involves its own covariance. On the other hand, a centralized scheme would result to an SDP where the size of the terminal semidefinite constraint increases with the total number of agents.

VI. Conclusions

In this work, we proposed multi-agent covariance steering as a method that can provide probabilistic safety guarantees and an upper bound on the uncertainty of the terminal states of the agents. To deal with the excessive computational demands of centralized multi-agent covariance control, we suggested an ADMM-based approach that solves the problem in a distributed fashion. This framework leads to each agent solving a modified version of the single-agent covariance steering problem in parallel. Simulation results on teams of vehicles verify the effectiveness of our approach and, most importantly, demonstrate its scalability to large-scale stochastic multi-agent systems.

In future works, we aim to address the multi-agent covariance control problem for agents with nonlinear dynamics. Furthermore, we plan to explore how distributed optimization architectures, such as ADMM, can be used in other stochastic control problems beyond covariance steering, leading to scalable multi-agent control algorithms. Finally, we will also consider problems with multi-agent systems under partial state information and time-varying topologies.

Acknowledgments

The work of the first and fourth author was supported by awards NSF CMMI-1936079 and ARO W911NF2010151. The work of the second and third author was supported in part by NSF awards ECCS-1924790 and CMMI-1937957. Alexandros Tsolovikos acknowledges support by the A. Onassis Foundation scholarship.

References

[1] MOSEK ApS. *The MOSEK optimization toolbox for MATLAB manual. Version 9.0.*, 2019. URL http://docs.mosek.com/9.0/toolbox/index.html.

[2] Efstathios Bakolas. Optimal covariance control for stochastic linear systems subject to integral quadratic state constraints. In *2016 American Control Conference (ACC)*, pages 7231–7236. IEEE, 2016.

[3] Efstathios Bakolas. Optimal covariance control for discrete-time stochastic linear systems subject to constraints. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 1153–1158. IEEE, 2016.

[4] Efstathios Bakolas. Finite-horizon covariance control for discrete-time stochastic linear systems subject to input constraints. *Automatica*, 91:61–68, 2018.

[5] Efstathios Bakolas. Dynamic output feedback control of the Liouville equation for discrete-time SISO linear systems. *IEEE Transactions on Automatic Control*, 64 (10):4268–4275, 2019.

[6] Efstathios Bakolas. Minimum variance and covariance steering based on affine disturbance feedback control parameterization. *arXiv preprint arXiv:2011.05394*, 2020.

[7] Isin M Balci and Efstathios Bakolas. Covariance steering of discrete-time stochastic linear systems based on Wasserstein distance terminal cost. *IEEE Control Systems Letters*, 2020.

[8] Isin M Balci and Efstathios Bakolas. Covariance control of discrete-time Gaussian linear systems using affine disturbance feedback control policies. *arXiv preprint arXiv:2103.14428*, 2021.

[9] Stephen Boyd, Neal Parikh, and Eric Chu. *Distributed optimization and statistical learning via the alternating direction method of multipliers*. Now Publishers Inc, 2011.

[10] Yongxin Chen, Tryphon T Georgiou, and Michele Pavon. Optimal steering of a linear stochastic system to a final probability distribution, Part I. *IEEE Transactions on Automatic Control*, 61(5):1158–1169, 2015.

[11] Yongxin Chen, Tryphon T Georgiou, and Michele Pavon. Optimal steering of a linear stochastic system to a final probability distribution, Part II. *IEEE Transactions on Automatic Control*, 61(5):1170–1180, 2015.

[12] Yongxin Chen, Tryphon T Georgiou, and Michele Pavon. Optimal steering of a linear stochastic system to a final probability distribution, Part III. *IEEE Transactions on Automatic Control*, 63(9):3112–3118, 2018.

[13] Zilong Cheng, Jun Ma, Xiaoxue Zhang, Clarence W de Silva, and Tong Heng Lee. Admm-based parallel optimization for multi-agent collision-free model predictive control. *arXiv preprint arXiv:2101.09894*, 2021.

[14] Michael Grant and Stephen Boyd. CVX: Matlab software for disciplined convex programming, version 2.1, 2014.

[15] Karolos M Grigoriadis and Robert E Skelton. Minimum-energy covariance controllers. *Automatica*, 33(4):569–578, 1997.

[16] Abhishek Halder and Eric DB Wendel. Finite horizon linear quadratic Gaussian density regulator with Wasserstein terminal cost. In *2016 American Control Conference (ACC)*, pages 7249–7254. IEEE, 2016.

[17] Anthony Hotz and Robert E Skelton. Covariance control theory. *International Journal of Control*, 46(1):13–32, 1987.

[18] Georgios Kotsalis, Guanghui Lan, and Arkadi S Nemirovski. Convex optimization for finite-horizon robust covariance control of linear stochastic systems. *SIAM Journal on Control and Optimization*, 59(1):296–319, 2021.

[19] Viet-Anh Le and Truong X Nghiem. Gaussian process based distributed model predictive control for multi-agent systems using sequential convex programming and ADMM. In *2020 IEEE Conference on Control Technology and Applications (CCTA)*, pages 31–36. IEEE, 2020.

[20] Wei Li and Christos G Cassandras. Distributed cooperative coverage control of sensor networks. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 2542–2547. IEEE, 2005.

[21] Kazuhide Okamoto and Panagiotis Tsiotras. Optimal stochastic vehicle path planning using covariance steering. *IEEE Robotics and Automation Letters*, 4(3):2276–2281, 2019.

[22] Dimitra Panagou, Dušan M Stipanović, and Petros G Voulgaris. Distributed coordination control for multi-robot networks using Lyapunov-like barrier functions. *IEEE Transactions on Automatic Control*, 61(3):617–632, 2015.

[23] Soon-Seo Park, Youngjae Min, Jung-Su Ha, Doo-Hyun Cho, and Han-Lim Choi. A distributed ADMM approach to non-myopic path planning for multi-target tracking.

IEEE Access, 7:163589–163603, 2019.

[24] Felix Rey, Zhoudan Pan, Adrian Hauswirth, and John Lygeros. Fully decentralized ADMM for coordination and collision avoidance. In *2018 European Control Conference (ECC)*, pages 825–830. IEEE, 2018.

[25] Jack Ridderhof and Panagiotis Tsiotras. Uncertainty quantication and control during Mars powered descent and landing using covariance steering. In *2018 AIAA Guidance, Navigation, and Control Conference*, page 0611, 2018.

[26] Jack Ridderhof, Kazuhide Okamoto, and Panagiotis Tsiotras. Nonlinear uncertainty control with iterative covariance steering. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 3484–3490. IEEE, 2019.

[27] Jack Ridderhof, Kazuhide Okamoto, and Panagiotis Tsiotras. Chance constrained covariance control for linear stochastic systems with output feedback. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pages 1758–1763. IEEE, 2020.

[28] Anam Tahir, Jari Böling, Mohammad-Hashem Haghbayan, Hannu T Toivonen, and Juha Plosila. Swarms of unmanned aerial vehicles—a survey. *Journal of Industrial Information Integration*, 16:100106, 2019.

[29] Wentao Tang and Prodromos Daoutidis. Distributed nonlinear model predictive control through accelerated parallel ADMM. In *2019 American Control Conference (ACC)*, pages 1406–1411. IEEE, 2019.

[30] Herbert G Tanner, George J Pappas, and Vijay Kumar. Leader-to-formation stability. *IEEE Transactions on robotics and automation*, 20(3):443–455, 2004.

[31] Alexandros Tsolovikos and Efstathios Bakolas. Nonlinear covariance steering using variational gaussian process predictive models. *arXiv preprint arXiv:2010.00778*, 2020.

[32] Ruben Van Parys and Goele Pipeleers. Distributed model predictive formation control with inter-vehicle collision avoidance. In *2017 11th Asian Control Conference (ASCC)*, pages 2399–2404. IEEE, 2017.

[33] J-H Xu and Robert E Skelton. An improved covariance assignment theory for discrete systems. *IEEE transactions on Automatic Control*, 37(10):1588–1591, 1992.

[34] Zeji Yi, Zhefeng Cao, Evangelos Theodorou, and Yongxin Chen. Nonlinear covariance control via differential dynamic programming. In *2020 American Control Conference (ACC)*, pages 3571–3576. IEEE, 2020.