



## **Distributed Koopman operator learning from sequential observations**

Downloaded from: <https://research.chalmers.se>, 2026-04-30 15:39 UTC

Citation for the original published paper (version of record):

Azarbahrām, A., Liu, S., Incremona, G. (2026). Distributed Koopman operator learning from sequential observations. *European Journal of Control*, 89.

<http://dx.doi.org/10.1016/j.ejcon.2026.101497>

N.B. When citing this work, cite the original published paper.



# Distributed Koopman operator learning from sequential observations

Ali Azarbahram <sup>a</sup>, Shenyu Liu <sup>b</sup>, Gian Paolo Incremona <sup>c,\*</sup>

<sup>a</sup> The Department of Electrical Engineering, Chalmers University of Technology, Gothenburg, 412 96, Sweden

<sup>b</sup> The School of Automation, Beijing Institute of Technology, China

<sup>c</sup> The Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Milan, 20133, Italy

## ARTICLE INFO

Editor: Prof. T Parisini

### Keywords:

Koopman operator  
Distributed learning  
Multi-agent systems  
Nonlinear system identification

## ABSTRACT

This paper presents a distributed Koopman operator learning framework for modeling unknown nonlinear dynamics using sequential observations from multiple agents. Each agent estimates a local Koopman approximation based on lifted data and collaborates over a communication graph to reach exponential consensus on a consistent distributed approximation. The approach supports distributed computation under asynchronous and resource-constrained sensing. Its performance is demonstrated through simulation results, validating convergence and predictive accuracy under sensing-constrained scenarios and limited communication.

## 1. Introduction

The challenge of modeling and controlling systems with unknown nonlinear dynamics remains central to many engineering and scientific applications. A foundational idea is to represent nonlinear dynamics through the linear evolution of observables under the Koopman operator, originally introduced by Koopman (1931). Although the Koopman operator acts on an (in general) infinite-dimensional function space, its spectral objects (eigenvalues, eigenfunctions, and modes) provide a principled way to analyze and approximate nonlinear behavior using linear-algebraic tools. A modern perspective emphasizes how Koopman spectral structure links to global geometry of state space and to representation learning for dynamical systems; see, e.g., the overview in Mezić (2021).

Building on this operator-theoretic viewpoint, an extensive body of work has developed practical, data-driven approximations of Koopman representations. Early progress in fluid dynamics connected Koopman spectral analysis to coherent structure extraction and modal decompositions (Rowley et al., 2009; Schmid, 2010). These ideas were later unified and extended through snapshot-based operator approximation methods, most notably extended dynamic mode decomposition (EDMD), which constructs a finite-dimensional approximation of the Koopman operator from data and a chosen dictionary of observables (Williams et al., 2015b). Complementary lines of work have deepened the theoretical foundations and clarified when finite-dimensional Koopman-invariant subspaces exist, as well as how lifting choices affect approximation quality and computational cost (Mezić, 2005).

More recently, the Koopman framework has also expanded toward learning spectral information and operator structure in settings where classical eigen-decompositions are challenging, including methods that infer spectral measures and projections directly from measured trajectories (Korda et al., 2020). Collectively, these developments position Koopman-based modeling as a practical bridge between nonlinear dynamics and scalable linear prediction/control pipelines, while highlighting that performance hinges on the chosen lifting and on how data are collected, distributed, and communicated across sensing agents (Mezić, 2021; Williams et al., 2015b).

Koopman-based modeling has been increasingly adopted as a practical bridge between nonlinear dynamics and linear prediction/control tools, with demonstrated impact in diverse application domains. Representative examples include motion-planning formulations built around Koopman linear predictors (Gutow & Rogers, 2020), video-driven learning of dynamical evolution for forecasting (Comas et al., 2021), and robust quadrotor control designs that exploit lifted linear structure for improved performance under uncertainty (Oh et al., 2024). In dynamic environments, Koopman predictors have also been used to forecast obstacle motion efficiently and to embed these forecasts into collision-avoidance strategies (Lu et al., 2024).

On the methodological side, a step toward control-oriented Koopman models is the development of linear predictors for nonlinear controlled systems via lifting and EDMD-type regression, yielding models that can be directly used within model predictive control (MPC) while retaining computational structure comparable to linear MPC (Korda & Mezić, 2018). More recently, Koopman learning has been integrated into safety-critical navigation architectures for UAVs operating in dynamic

\* Corresponding author.

E-mail addresses: [ali.azarbahram@chalmers.se](mailto:ali.azarbahram@chalmers.se) (A. Azarbahram), [shenyuli@bit.edu.cn](mailto:shenyuli@bit.edu.cn) (S. Liu), [gianpaolo.incremona@polimi.it](mailto:gianpaolo.incremona@polimi.it) (G.P. Incremona).

scenes: Azarbahram et al. (2025a) introduces a Koopman-enhanced distributed switched MPC framework in which a localized Koopman operator is learned online to predict moving-obstacle trajectories and enable scalable, collision-free coordination; complementary work in Bueno et al. (2025) leverages real-time LiDAR measurements to learn Koopman predictors of surrounding moving objects and embeds them in an MPC loop. Together, these results highlight both the versatility of Koopman predictors across perception–prediction–planning pipelines and their suitability for real-time, constraint-aware decision making in dynamic environments.

The learning of Koopman operators can be formulated as a data fitting problem, typically solved using least squares algorithms (Hansen et al., 2013). However, as system size and complexity increase, particularly in multi-agent and networked settings, centralized Koopman learning becomes impractical due to communication, privacy, computation, and scalability limitations. Distributed approaches address these challenges by enabling agents to locally estimate Koopman operators using partial observations and limited communication, thereby improving scalability, preserving privacy, and supporting dynamic network conditions. Implementing such distributed learning requires solving the least squares problem in a distributed manner. Toward this goal, recent advances in distributed algorithms, particularly those in Huang et al. (2022), Liu et al. (2019), Wang et al. (2019b), Yang et al. (2020), achieve exponential convergence guarantees. For broader context, the surveys (Wang et al., 2019a; Zheng & Liu, 2022) offer comprehensive reviews of related methods.

Recent work has explored various approaches for distributed Koopman operator learning. In Nandanoori et al. (2021), a block-wise Koopman formulation is proposed for multi-agent systems, but the focus remains on block-structured learning under central data access assumptions. The studied approach in Mukherjee et al. (2022) extends this by exploiting graph sparsity and geometric structure. Deep learning-based methods such as Hao et al. (2024a), Hao et al. (2024b) propose distributed deep Koopman algorithms using neural networks for system identification and control, requiring synchronized neural architectures and backpropagation routines across agents. In Liu et al. (2020), a distributed method is developed under state-based decomposability assumptions to ensure convergence. More recently, Azarbahram et al. (2025b) investigates distributed Koopman learning with an emphasis on perception and safe navigation through information exchange.

Our work focuses on the distributed solution of a Frobenius-norm-based Koopman approximation problem under temporally fragmented data access across agents. This setting arises naturally in sensing-constrained scenarios, such as multi-UAV monitoring or satellite-based observation, where agents are unable to continuously record system trajectories due to bandwidth, energy, or sensing limitations (e.g., Landsat Wulder et al., 2012), or multi-UAV sensing and tracking (e.g., Cao et al., 2012; Schwager et al., 2011). In such environments, agents must observe a shared system sequentially over time, resulting in partial temporal snapshots that are distributed across the network rather than spatially partitioned or centrally available. While Koopman-based representations are attractive in practice due to their linear structure and compatibility with prediction and control, existing distributed formulations typically rely on block-structured subsystems, synchronized learning architectures, or specialized information exchange mechanisms.

In contrast, the problem addressed here considers a shared, nonseparable environment observed intermittently by multiple agents, where no single agent has access to a complete trajectory. The proposed framework enables agents to collaboratively reconstruct a consistent Koopman operator by fusing temporally fragmented observations through local communication over a generic graph. This formulation is particularly relevant in practical sensing missions, where sequential data acquisition is a constraint rather than a design choice. By casting the learning problem as a distributed Frobenius-norm minimization with consensus constraints, the proposed algorithm avoids raw data sharing, deep learn-

ing architectures, and parametric model assumptions, while admitting a unified matrix-theoretic convergence analysis.

The main contributions of this paper are summarized as follows:

- We formulate a distributed Koopman operator learning problem under temporally fragmented observations. These targets sensing-constrained scenarios where agents sequentially observe a shared nonlinear system without access to complete trajectories or subsystem decompositions. Unlike existing literature that often assumes spatial partitioning or synchronized data access (e.g., Azarbahram et al., 2025b; Hao et al., 2024a), our formulation addresses the challenge of reconstructing global dynamics from intermittent, local temporal snapshots.
- We establish a theoretical convergence guarantee for the proposed distributed learning framework. We introduce a novel algorithm based on a proportional-integral (PI) consensus law and provide a rigorous matrix-theoretic analysis to prove its stability. Specifically, we demonstrate that the local Koopman estimates across the network reach a consensus and converge exponentially fast to the optimal centralized Frobenius-norm solution, providing explicit design criteria for the algorithm’s step-size and gains.
- We validate the framework through a practical application to multi-UAV crowd-density monitoring. By simulating a team of UAVs with staggered sensing schedules, we demonstrate that the distributed approach successfully recovers the evolution of a spatially distributed intensity field. The results show that the learned operators achieve high predictive accuracy on unseen data, effectively bridging the gap between local, fragmented sensing and global, high-fidelity forecasting in a resource-constrained multi-agent environment.

The remainder of the paper is structured as follows. The introduction continues with notations and graph definitions; Section 2 presents the problem formulation and preliminaries. Section 3 details the proposed method. Section 4 illustrates simulation results, and Section 5 concludes the paper with future directions.

*Notations and graph definition.* We denote by  $\mathbb{N}$  the set of non-negative integers,  $\mathbb{R}$  the set of real numbers,  $\mathbb{C}$  the set of complex numbers,  $\mathbb{R}^n$  the  $n$ -dimensional real space, and  $\mathbb{R}^{n \times m}$  the space of  $n \times m$  real matrices. In particular,  $0_{n \times m}$ , (resp.  $1_{n \times m}$ ) denotes the  $n \times m$ -dimensional zero matrix (resp. all-ones matrix), while  $I_n$  represents the  $n \times n$  identity matrix. When the dimensions are clear from the context, we remove the subindices. A diagonal matrix composed by elements  $a_1, \dots, a_p$  is denoted by  $\text{diag}(a_1, \dots, a_p)$ , and a block diagonal matrix composed by matrices  $A_1, \dots, A_p$  is denoted by  $\text{diag}(A_1, \dots, A_p)$ . The Kronecker product of two matrices  $A, B$  is denoted by  $A \otimes B$ , and the transpose of matrix  $A$  is denoted by  $A^\top$ . For square matrix  $A \in \mathbb{R}^{n \times n}$ , let  $\det(A)$  be its determinant,  $\text{tr}(A)$  be its trace, and  $\Lambda(A)$  be its spectrum (i.e., set of all eigenvalues). For  $\lambda \in \mathbb{C}$ , let  $\text{Re}(\lambda)$ ,  $\text{Im}(\lambda)$  be its real and imaginary parts, respectively. For any vector  $x \in \mathbb{R}^n$ , let  $\|x\|$  be its 2-norm. For any  $A \in \mathbb{R}^{m \times n}$ , let  $\|A\|_F := \text{tr}(A^\top A)$  be its Frobenius norm.

An undirected graph  $G = (\mathcal{P}, \mathcal{E})$  consists of the vertex set  $\mathcal{P} := \{1, \dots, p\}$ ,  $p \in \mathbb{N}$ , and the edge set  $\mathcal{E} \subseteq \mathcal{P} \times \mathcal{P}$ , such that  $(i, j) \in \mathcal{E}$  if and only if  $(j, i) \in \mathcal{E}$ . A path is a sequence of vertices connected by edges, and the graph  $G$  is *connected* if there is a path between any pair of vertices. For any  $i \in \mathcal{P}$ , the set of neighbors of  $i$  is  $N(i) := \{j \in \mathcal{P} : (i, j) \in \mathcal{E}\}$ . A *Laplacian matrix*  $L = [L_{ij}] \in \mathbb{R}^{p \times p}$  of a graph  $G$  is given by

$$L_{ij} = \begin{cases} -1, & \text{if } (i, j) \in \mathcal{E}, i \neq j, \\ 0, & \text{if } (i, j) \notin \mathcal{E}, i \neq j, \\ -\sum_{k \neq i} L_{ik}, & \text{if } i = j. \end{cases}$$

## 2. Preliminaries and problem statement

The Koopman operator framework enables the study of nonlinear dynamical systems using linear operators acting on functions. Consider

a discrete-time nonlinear system of the form

$$x_{k+1} = f(x_k), \quad x_k \in \mathcal{M} \subseteq \mathbb{R}^q, \quad (1)$$

where  $f : \mathcal{M} \rightarrow \mathcal{M}$  is a nonlinear map and  $q$  is the state dimension. Rather than studying the trajectory of the state  $x_k$  directly, we analyze the evolution of observables  $\psi : \mathcal{M} \rightarrow \mathbb{R}$  that belong to a Hilbert space  $\mathcal{H}$  of scalar-valued functions.

The infinite-dimensional Koopman operator  $\mathcal{K} : \mathcal{H} \rightarrow \mathcal{H}$  is defined as

$$(\mathcal{K}\psi)(x) = \psi(f(x)), \quad \forall \psi \in \mathcal{H}, x \in \mathcal{M}. \quad (2)$$

Although the system dynamics  $f$  is nonlinear, the Koopman operator  $\mathcal{K}$  is linear in the space of observables. We select  $n$  scalar observables and assemble them into a vector-valued function:

$$\Psi(x) = [\psi_1(x), \psi_2(x), \dots, \psi_n(x)]^\top \in \mathbb{R}^n, \quad (3)$$

where  $\Psi : \mathcal{M} \rightarrow \mathbb{R}^n$  maps each state  $x \in \mathcal{M}$  to an  $n$ -dimensional feature space (Williams et al., 2015b). We then collect  $N$  data samples  $\{x_k, f(x_k)\}_{k=1}^N$  and define the data matrices

$$X = [\Psi(x_1), \Psi(x_2), \dots, \Psi(x_N)] \in \mathbb{R}^{n \times N}, \quad (4)$$

$$Y = [\Psi(f(x_1)), \Psi(f(x_2)), \dots, \Psi(f(x_N))] \in \mathbb{R}^{n \times N}. \quad (5)$$

To facilitate computation, the EDMD method (Williams et al., 2015b) projects the action of  $\mathcal{K}$  onto the span of the selected observables. This results in a matrix  $K \in \mathbb{R}^{n \times n}$  that approximates the Koopman operator over the chosen feature space and governs the evolution of lifted states in the observable space. Hence, the finite-dimensional approximation of the Koopman operator  $\mathcal{K}$  via EDMD seeks a matrix  $K \in \mathbb{R}^{n \times n}$  that best satisfies

$$Y \approx KX. \quad (6)$$

This leads to the following least-squares optimization using the Frobenius norm:

$$K^* = \arg \min_{K \in \mathbb{R}^{n \times n}} \|Y - KX\|_F^2. \quad (7)$$

In a distributed setting, each agent  $i \in \{1, \dots, p\}$  collects its own segment of the system observations, resulting in local data matrices  $Y_i \in \mathbb{R}^{n \times m_i}$  and  $X_i \in \mathbb{R}^{n \times m_i}$ , where  $m_i$  denotes the number of temporal snapshots available to agent  $i$ . These data blocks represent lifted observables evaluated at the agent's local state transitions. Such partitioning naturally arises in sequential observation frameworks, where agents take turns sensing the system over time, or when access to data is constrained by communication, privacy, or coverage limitations. To construct a distributed Koopman approximation, the locally collected data are aggregated to form the global matrices

$$Y = [Y_1 \quad Y_2 \quad \dots \quad Y_p], \quad X = [X_1 \quad X_2 \quad \dots \quad X_p],$$

with  $Y, X \in \mathbb{R}^{n \times N}$  and  $\sum_{i=1}^p m_i = N$ . This structure enables decentralized computation while preserving the full representation of the system's evolution.

From the problem formulation, the  $i$ th agent only knows  $Y_i, X_i$ . Because  $\|Y - KX\|_F^2 = \sum_{i=1}^p \|Y_i - KX_i\|_F^2$ , we could study the distributed problem that each agent aims to find a solution  $K_i \in \mathbb{R}^{n \times n}$  for the local problem  $\min_{K_i} \|Y_i - K_i X_i\|_F$ , subject to the constraint that all  $K_i$  must be equal. Furthermore, denote

$$\mathbf{K} := [K_1 \quad K_2 \quad \dots \quad K_p] \in \mathbb{R}^{n \times np}, \quad (8)$$

$$\mathbf{X} := \text{diag}(X_1, X_2, \dots, X_p) \in \mathbb{R}^{np \times np}, \quad (9)$$

$$\mathbf{L} := L \otimes I_n \in \mathbb{R}^{np \times np}, \quad (10)$$

where  $L$  is the Laplacian matrix of the communication graph  $G$ . We further have that  $\|Y - KX\|_F^2 = \sum_{i=1}^p \|Y_i - KX_i\|_F^2 = \|Y - \mathbf{KX}\|_F^2$ , subject to the constraint  $K = K_1 = K_2 = \dots = K_p$ , that is,  $\mathbf{K} = \mathbf{1}_{1 \times p} \otimes K$ , which can be equivalently expressed as  $\mathbf{KL} = 0$  since the communication graph

$G$  is connected. Hence, to solve (7) by a distributed method in this scenario, we can equivalently study the optimization problem

$$\min_{\mathbf{K} \in \mathbb{R}^{n \times np}} \frac{1}{2} \|Y - \mathbf{KX}\|_F^2, \quad (11a)$$

$$\text{subject to } \mathbf{KL} = 0. \quad (11b)$$

### 3. Main results

We propose to solve the problem (7) by a discrete-time distributed algorithm. Essentially, the agents alternate between communication and computation, such that they run the following updating law

$$K_i^+ = K_i - \alpha \left( (K_i X_i - Y_i) X_i^\top + k_p \sum_{j \in \mathcal{N}(i)} (K_i - K_j) + k_I R_i \right), \quad (12a)$$

$$R_i^+ = R_i + \alpha \sum_{j \in \mathcal{N}(i)} (K_i - K_j). \quad (12b)$$

Here,  $\alpha > 0$  is a fixed step-size,  $k_p, k_I > 0$  are some tunable gains,  $K_i \in \mathbb{R}^{n \times n}$  is the  $i$ th agent's individual guess of the optimum of (7),  $R_i \in \mathbb{R}^{n \times n}$  is another internal augmented state variable owned by the  $i$ th agent for facilitating the convergence. The distributed algorithm for Koopman operator learning is summarized in Algorithm 1. The update law is similar to the PI-control studied in Yang et al. (2019), and we provide some intuitive interpretations which help understand Algorithm 1.

- The term  $(Y_i - K_i X_i) X_i^\top$  corresponds to the negative gradient of the local cost  $\frac{1}{2} \|Y_i - K_i X_i\|_F^2$ , indicating that each agent minimizes its own Koopman approximation error; equivalently, the update law is expressed as a gradient-descent step using the gradient  $(K_i X_i - Y_i) X_i^\top$  in (12a).
- The term  $\sum_{j \in \mathcal{N}(i)} (K_i - K_j)$  in (12b) acts as a proportional diffusion term, promoting consensus among neighboring agents.
- Given  $R_i(0) = 0$ , the update in (12b) shows that  $R_i$  accumulates the diffusion term over time, functioning as an integral feedback that enforces consensus across agents.

---

#### Algorithm 1 Distributed Koopman operator learning.

---

**Input:**  $k_p, k_I$  and  $\alpha$  satisfying  $\alpha < \alpha_{\max}$  from (14).

1: Initialize arbitrary  $K_i(0) \in \mathbb{R}^{n \times n}$ ,  $R_i(0) = 0_{n \times n}$  for all  $i \in \mathcal{P}$ .

2: **loop** for  $t = 0, 1, \dots, t_{\max} - 1$ , the  $i$ th agent,  $i \in \mathcal{P}$

3: Broadcast  $K_i(t)$  to its neighbors.

4: Compute  $K_i(t+1), R_i(t+1)$  according to (12).

5: **end loop**

**Output:**  $K_i(t_{\max}), i \in \mathcal{P}$ .

---

The convergence of Algorithm 1 is related to the eigenvalues of the following matrix

$$\mathbf{M} := \begin{bmatrix} -\mathbf{X}\mathbf{X}^\top - k_p \mathbf{L} & \mathbf{L} \\ -k_I \mathbf{I}_{np} & 0 \end{bmatrix} \in \mathbb{R}^{2np \times 2np}, \quad (13)$$

where  $\mathbf{X}, \mathbf{L}$  are defined in (9) and (10). Note that  $\mathbf{M}$  depends on the gains  $k_p, k_I$ . The following theorem provides the design criteria of these parameters and the step-size  $\alpha$  which guarantees the convergence of Algorithm 1, and relates the convergence rate to these parameters.

**Theorem 1.** *Suppose the communication graph  $G$  is undirect and connected. For any gains  $k_p, k_I > 0$ , there exists*

$$\alpha_{\max} := - \max_{\lambda \in \Lambda(\mathbf{M}) \setminus \{0\}} \frac{2\text{Re}(\lambda)}{|\lambda|^2} > 0, \quad (14)$$

*such that as long as the step size  $\alpha < \alpha_{\max}$ , the  $K_i$  components of the algorithm will reach consensus and converge to an optimal solution of the problem (7). Furthermore, the convergence is exponential with rate  $\rho > \rho_{\max}$ , where*

$$\rho_{\max} := \max_{\lambda \in \Lambda(\mathbf{M}) \setminus \{0\}} \sqrt{1 + 2\alpha \text{Re}(\lambda) + \alpha^2 |\lambda|^2}. \quad (15)$$

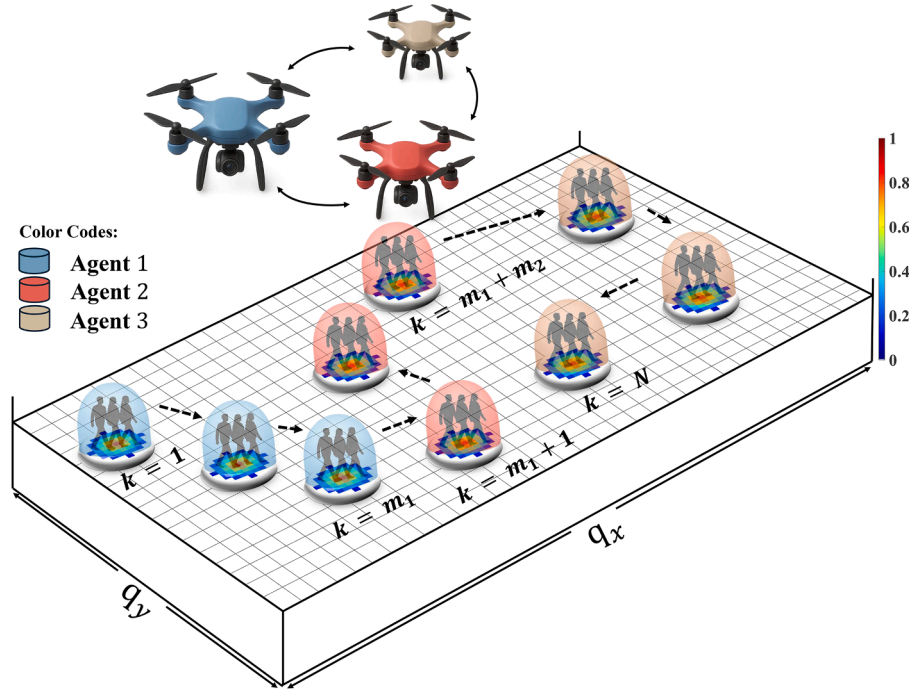


Fig. 1. Conceptual illustration of sequential sensing and distributed Koopman learning.

The proof of [Theorem 1](#) is provided in the Appendix. We note that the upper bound  $\alpha_{\max}$  in (14) on the step size must be computed using the matrix  $\mathbf{M}$ , which depends on the centralized quantities  $X, L$ . Consequently,  $\alpha_{\max}$  cannot be determined a priori in a distributed setting. Our convergence guarantee for [Algorithm 1](#) should be interpreted qualitatively: provided the step size is sufficiently small, the algorithm converges exponentially fast to an optimal solution of (7). Future work will explore the development of distributed algorithms that either incorporate adaptive step sizes or eliminate the need for step-size tuning altogether.

#### 4. Simulation results

We consider a team of UAVs tasked with monitoring the evolution of a spatially distributed crowd-density field over time. Although each UAV is equipped with a camera capable of observing the entire region of interest, continuous high-rate sensing is impractical due to energy limitations, camera usage constraints, and onboard processing costs. As a result, the UAVs operate under a sequential sensing protocol, where data acquisition is staggered across agents over a finite time horizon. Specifically, the sensing task is distributed over  $k \in \{1, \dots, N\}$  discrete time steps such that each UAV records only a subset of the overall trajectory, while the collective measurements cover the full sequence of snapshots. This setting naturally leads to temporally fragmented data across agents, without any single UAV having access to the complete dataset.

We consider  $p = 3$  UAVs connected through a circular, undirected communication graph. The UAVs acquire measurements in sequence: UAV 1 collects the first  $m_1$  snapshots, UAV 2 collects the subsequent  $m_2$  snapshots, and UAV 3 collects the remaining  $m_3$  snapshots, forming one full sensing cycle with  $N = m_1 + m_2 + m_3$  total samples. Each agent processes only its locally acquired data and participates in the distributed update law (12) through neighbor-to-neighbor communication.

A conceptual illustration of this sequential sensing process is shown in [Fig. 1](#). The monitored environment is modeled as a two-dimensional grid of size  $q_x \times q_y$ , yielding a global state dimension  $q = q_x q_y$  in (1). In the simulations, we set  $q_x = q_y = 20$ , resulting in  $q = 400$  spatial states.

At each sensing instant, the active UAV observes the full  $20 \times 20$  intensity field, but only for its assigned time window in the sensing schedule. Each UAV acquires  $m_i = 3$  snapshot pairs before handing off sensing responsibility to the next agent, resulting in  $N = 9$  total samples over one cycle. In addition to the snapshots used for learning, we reserve a subsequent sequence of nine future snapshots that are not used during training and are instead employed to evaluate multi-step prediction performance of the Koopman operator learned from the sequentially acquired data. The evolving crowd generates normalized intensity fields taking values in  $[0, 1]$ , which are visualized using color maps to represent spatial density variations. Shaded regions also indicate which UAV is active at each time interval. For example, a blue-highlighted dome from  $k = 1$  to  $k = m_1$  denotes measurements acquired by UAV 1.

The intensity maps constitute the system state  $x_k$  and serve as inputs to the Koopman learning procedure. For lifting, all agents employ a shared observable structure that directly vectorizes the spatial intensity field, yielding a lifted state dimension  $n = 400$ . The resulting lifted data are used to construct the local matrices  $(X_i, Y_i)$  for each agent, which are then incorporated into the distributed Koopman learning algorithm. The results presented below demonstrate that, despite the absence of centralized data aggregation and the presence of sequentially partitioned observations, the proposed distributed method enables all agents to collectively recover a Koopman operator that closely matches the centralized solution.

Each agent runs 600 iterations of the distributed update, using  $k_p = 150$ ,  $k_l = 50$ , and  $\alpha = 0.5\alpha_{\max}$  with  $\alpha_{\max} = 0.03$  from (14). This ensures exponential convergence with  $\rho_{\max} = 0.96$  as per (15). Let  $K^*$  be the centralized solution to (7) using  $(X, Y)$ , and  $K_i$  the local estimate from agent  $i$ . Define the average distributed operator as  $K_{\text{ave}} := 1/p \sum_{i=1}^p K_i$ . [Fig. 2](#) shows that  $K^*$  and  $K_{\text{ave}}$  share dominant eigenvalues near the unit circle, indicating accurate capture of persistent dynamics. Eigenvalues near the origin in  $K_{\text{ave}}$  reflect suppressed modes from limited observability and averaging, contributing to robustness. We should note that the constraint (11b) enforces agreement only at optimality, i.e., at convergence one has  $K_1 = K_2 = \dots = K_p$ . However, during the transient evolution of [Algorithm 1](#), the local iterates  $K_i(t)$  are generally not identical. For this reason, we define  $K_{\text{ave}}(t)$  as a compact network-level representative of

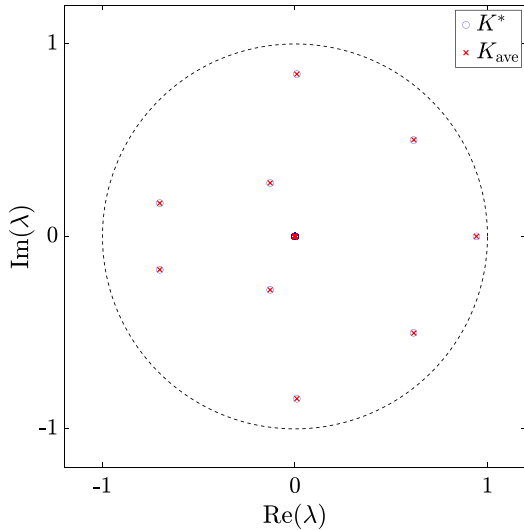


Fig. 2. Spectral comparison between  $K_{\text{ave}}$  and  $K^*$ .

the distributed estimate, which is convenient for visualization and for comparing the collective behavior of the distributed algorithm against the centralized solution  $K^*$ . As  $t \rightarrow \infty$ , the consensus error vanishes and  $K_{\text{ave}}(t)$  coincides with each  $K_i(t)$ .

To assess accuracy, we compute  $1/p \sum_{i=1}^p \|Y - K_i(t)X\|_F$ , measuring how well each distributed Koopman operator  $K_i$  generalizes to the full dataset  $(X, Y)$ . Despite being trained on local data, agents achieve low reconstruction error over time, as shown in Fig. 3, indicating alignment with global measurements and inter-agent consensus. Fig. 4 illustrates the evolution of the distributed objective function  $\frac{1}{2} \sum_{i=1}^p \|Y_i - K_i(t)X_i\|_F^2$  over the algorithm iterations. The results highlight the rapid decrease of the aggregate fitting error, confirming the exponential convergence behavior predicted by Theorem 1.

Fig. 5 reports the evolution of the consensus error among agents, quantified by  $\frac{1}{p} \sum_{i=1}^p \|K_i(t) - K_{\text{ave}}(t)\|_F$ . The monotonic decay indicates that the locally learned Koopman operators progressively align through neighbor communication. This behavior confirms that the distributed update law effectively enforces agreement across agents despite temporally fragmented observations. Fig. 6 also illustrates the distance between the distributed Koopman operators and the centralized solution  $K^*$ . The quantity  $\frac{1}{p} \sum_{i=1}^p \|K_i(t) - K^*\|_F$  decreases steadily over the iterations, demonstrating that the distributed learning process converges not only to consensus, but also toward the optimal centralized least-squares solution.

To assess predictive performance beyond the training horizon, we evaluate the learned Koopman operator on a reserved sequence of snapshots that are not used during the learning phase. Specifically, after estimating the distributed Koopman operator from the first  $N = 9$  sequentially acquired samples, an additional set of nine future snapshots is held out and used exclusively for multi-step prediction. Fig. 7 reports the absolute prediction error between the true system evolution and the distributed Koopman-based forecasts over this future horizon. The error is shown across both time and spatial coordinates, providing a detailed view of how prediction accuracy evolves as the horizon increases. The observed errors remain consistently small (on the order of  $10^{-2}$ ) and exhibit coherent spatial structure, indicating stable propagation of prediction accuracy rather than error accumulation.

To provide a direct and interpretable comparison between centralized and distributed predictions, we report in Table 1 a time-series evaluation of the prediction error over the reserved (unseen) snapshots. The centralized Koopman model and the distributed models are used to predict the system evolution over a future horizon, and at each step the error is computed as the  $\ell_2$ -norm of the difference between the pre-

Table 1  
Time-series prediction error on reserved snapshots.

Horizon	Centralized	Distributed (P-only)	Distributed (PI)
1	$1.82 \times 10^{-9}$	$1.90 \times 10^{-1}$	$1.02 \times 10^{-2}$
2	$3.39 \times 10^{-9}$	$3.80 \times 10^{-1}$	$1.97 \times 10^{-2}$
3	$5.23 \times 10^{-9}$	$4.51 \times 10^{-1}$	$2.82 \times 10^{-2}$
4	$7.50 \times 10^{-9}$	$5.26 \times 10^{-1}$	$3.69 \times 10^{-2}$
5	$9.94 \times 10^{-9}$	$5.80 \times 10^{-1}$	$5.08 \times 10^{-2}$
6	$1.24 \times 10^{-8}$	$6.18 \times 10^{-1}$	$5.48 \times 10^{-2}$
7	$1.53 \times 10^{-8}$	$7.11 \times 10^{-1}$	$5.66 \times 10^{-2}$
8	$1.51 \times 10^{-8}$	$7.74 \times 10^{-1}$	$6.03 \times 10^{-2}$
9	$1.62 \times 10^{-8}$	$8.21 \times 10^{-1}$	$6.45 \times 10^{-2}$

dicted intensity field and the ground-truth snapshot. The table reports results for both the proposed distributed algorithm with proportional-integral (PI) consensus and a proportional-only (P-only) variant obtained by removing the integral state, which is included as an ablation study to assess the role of the integral term. The results show that both distributed variants yield stable multi-step predictions on unseen data, indicating that the learned Koopman operators generalize beyond the training snapshots. While the P-only variant exhibits comparable prediction accuracy over the tested horizons, the PI-based scheme consistently attains lower errors and provides improved agreement with the centralized least-squares solution, in line with the theoretical analysis. Together, these results highlight the practical forecasting capability of the proposed distributed Koopman learning framework under decentralized and sensing-constrained data acquisition.

We benchmark centralized and distributed implementations in MATLAB (Intel i7, 16GB RAM) for the studied  $20 \times 20$  grid. Centralized learning, based on  $K = YX^\dagger$  (with  $X^\dagger$  denoting the Moore–Penrose pseudoinverse), completes in  $\sim 15.2$  ms. Distributed learning takes  $\sim 0.3$  ms for local computation and  $\sim 0.1$  ms for communication per iteration, totaling  $\sim 400$  ms. Although centralized learning appears faster for small datasets, it requires full data aggregation and becomes more computationally expensive as the number of snapshots increases. Moreover due to energy, privacy, bandwidth, or field-of-view constraints, continuous high-resolution data collection from a single agent is often physically infeasible in real-world scenarios.

In our simulations, we employ direct vectorization of the  $20 \times 20$  grid as observables. This choice is natural for representing spatial phenomena such as pedestrian density and preserves locality in the lifted space. It enables interpretable modeling while maintaining computational tractability. Given our focus on distributed Koopman operator learning under distributed data access, this observable structure offers a practical and consistent approach for operator estimation across agents. We note that all agents observe temporally partitioned data from the same environment using a shared lifting function, rather than agent-specific observables. To further examine the role of the lifting dimension, we conducted an additional experiment in which the observable set was enriched by including second-order terms, effectively increasing the expressiveness of the lifted space. Compared to the baseline lifting that uses only first-order observables, the enriched lifting yields a noticeably faster reduction of the distributed objective value. In particular, the quantity  $\frac{1}{p} \sum_{i=1}^p \|Y - K_i(t)X\|_F$  reaches the threshold  $10^{-2}$  after approximately 600 iterations for first-order lifting (Fig. 3), whereas the same level of accuracy is achieved after roughly 300 iterations when second-order terms are included. This behavior highlights a fundamental trade-off in Koopman-based modeling. Richer lifting functions improve the linear representability of nonlinear dynamics, leading to faster convergence in terms of iteration count. However, this improvement comes at the cost of increased lifted dimension, which directly impacts memory requirements, local computation, and inter-agent communication. In the distributed setting, where each agent must exchange lifted quantities, the use of higher-order observables approximately doubles both the local computation time and the communication load per iteration.

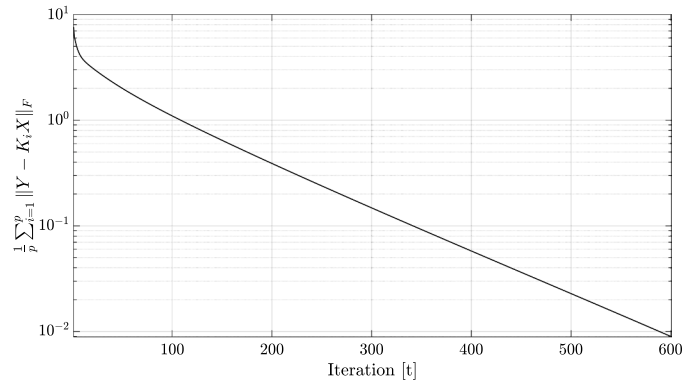


Fig. 3. Prediction error of distributed Koopman operators.

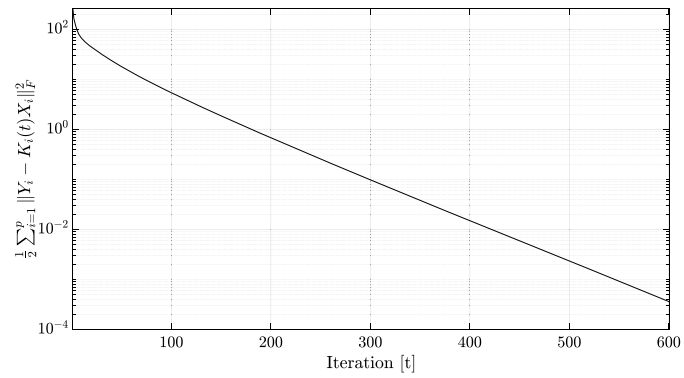


Fig. 4. Convergence of the distributed Koopman learning objective.

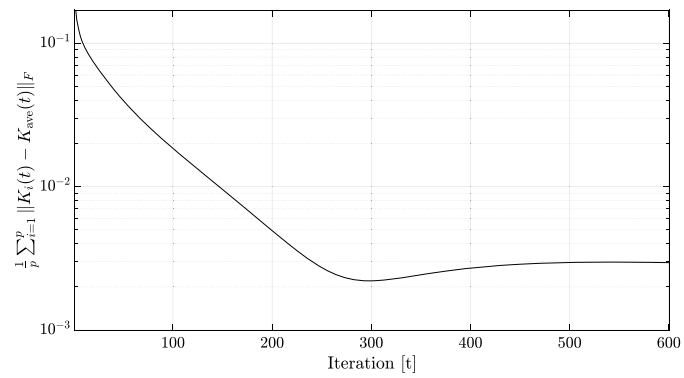


Fig. 5. Consensus among distributed Koopman operators.

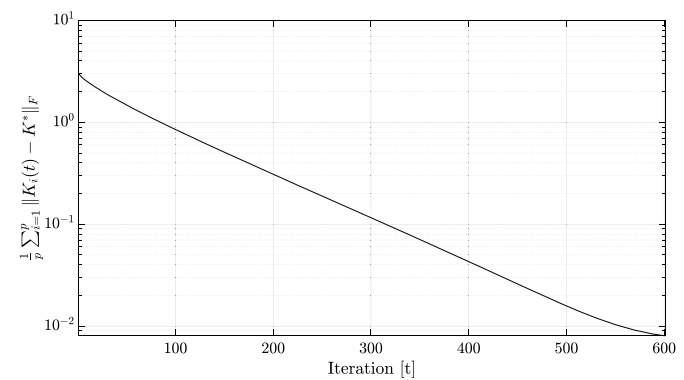


Fig. 6. Convergence to the centralized Koopman solution.

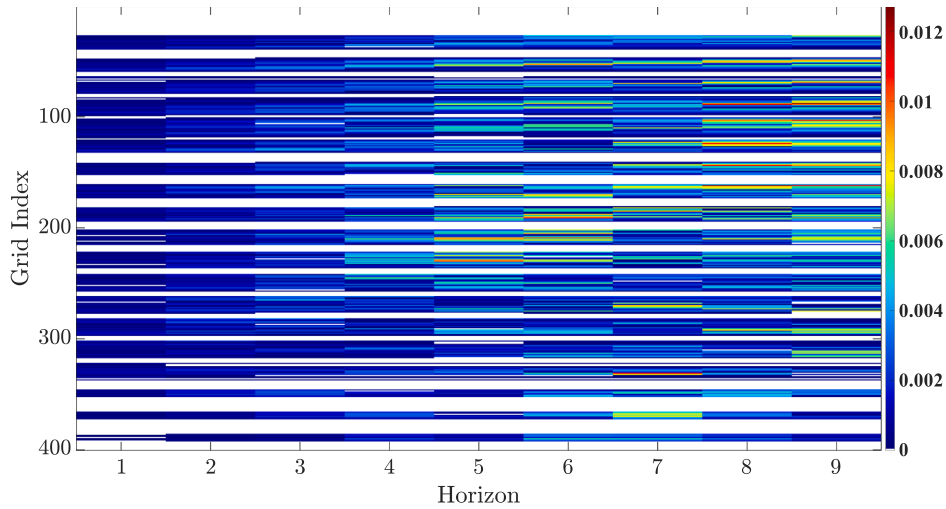


Fig. 7. Heatmap of distributed Koopman prediction error.

From a centralized perspective, increasing the lifting dimension also amplifies the computational burden of forming and inverting large data matrices, potentially offsetting the gains in convergence speed. These results indicate that the choice of lifting structure should balance model expressiveness against computational and communication constraints, particularly in distributed and resource-limited scenarios. The proposed framework accommodates such trade-offs naturally, allowing practitioners to tailor the lifting complexity to the available resources and desired convergence behavior.

**Remark 1.** In [Algorithm 1](#), each agent  $i$  maintains a local Koopman matrix  $K_i \in \mathbb{R}^{n \times n}$  (with  $n$  the lifted dimension) and exchanges it with its neighbors at each communication round; consequently, the per-iteration communication payload scales with the number of transmitted real values, i.e., on the order of  $n^2$  per neighbor. In the simulation setting considered here, the lifting is a direct embedding of a  $20 \times 20$  intensity map, hence  $n = 400$  and  $K_i$  contains  $1.6 \times 10^5$  entries, which remains manageable for episodic learning with sparse neighborhood communication. The intended regime of the proposed method is sensing-constrained scenarios where the lifted dimension is moderate and sequential observations motivate distributed computation without centralized data aggregation. At the same time, we note that as the lifted dimension  $n$  increases, exchanging full dense matrices imposes higher communication requirements, which naturally motivates consideration of additional structure in the Koopman representation for each agent.

**Remark 2.** The convergence of [Algorithm 1](#) is driven by the spectrum of  $\mathbf{M}$  in (13), shaped by  $k_p, k_f$ :

- $k_p$ : enhances consensus by penalizing disagreement. Too high may suppress local fitting.
- $k_f$ : reinforces long-term agreement but risks instability unless  $\alpha$  is small.
- $\alpha$ : must satisfy  $\alpha < \alpha_{\max}$  from (14). Larger  $k_p, k_f$  reduce the spectral radius of  $\mathbf{M}$ , increasing  $\alpha_{\max}$  and improving the convergence rate governed by  $\rho_{\max}$  in (15). Set  $\alpha = \theta \alpha_{\max}$  with  $\theta \in (0.3, 0.6)$ .
- **Practice:** use moderate gains, compute  $\alpha_{\max}$ , and set  $\alpha = 0.5 \alpha_{\max}$ . To accelerate, increase  $k_p$ ; for stability, decrease  $\alpha$  or  $k_f$ . Choose  $t_{\max}$  based on  $\rho_{\max}$ .

**Remark 3.** In practice, each agent collects local data over a brief sequence of time steps, after which [Algorithm 1](#) is executed. This interpretation of sequential observations aligns with physical constraints such as frame-rate or communication delays in agent sensing. [Fig. 3](#) shows that the distributed Koopman operators  $K_i$  achieve low reconstruction error after around 300 iterations, which occur on millisecond time scales

that is significantly faster than the few-second intervals at which video snapshots (e.g., of pedestrian motion) typically arrive. This separation ensures convergence is feasible before new data becomes available. As confirmed in [Fig. 7](#), the average operator  $K_{\text{ave}}$  provides accurate multi-step predictions even under decentralization, enabling online forecasting while new data is being acquired. For example, as three agents collect the next  $N = 9$  snapshots (each with  $m_i = 3$ ), the current  $K_i$ 's can be used for short-term prediction. Once the data is available, the algorithm is rerun to refine the models, making the framework suitable for real-time deployment in applications such as crowd monitoring.

**Remark 4.** [Algorithm 1](#) converges exponentially with rate governed by  $\rho_{\max}$  in (15). As shown in [Fig. 3](#), approximately an accurate reconstruction is achieved after  $\sim 300$  iterations. In online settings, where time and resources are limited, sub-optimal iterates can still be used for short-term prediction. [Fig. 7](#) shows that even intermediate solutions yield reliable forecasts. As new data arrives, the algorithm can resume from the latest iterate, enabling continual refinement. This supports real-time deployment, where learning proceeds alongside sensing and control.

To contextualize the proposed method, we compare our distributed Koopman learning framework against the distributed approach introduced in [Hao et al. \(2024a\)](#), which represents one of the most closely related existing results. Both approaches address the problem of learning Koopman operators from partial and distributed observations, making this comparison particularly relevant and fair. To ensure consistency across methods, the lifting functions are kept fixed and identical for the simulation runs, and both distributed algorithms are executed for the same number of iterations over the same communication graph. Moreover, since the present study focuses on autonomous prediction rather than control, the input-related components in [Hao et al. \(2024a\)](#) are omitted, resulting in a formulation that aligns with our problem setup.

Prediction performance is evaluated using the root mean square error (RMSE) computed over the reserved set of unseen snapshots. Specifically, for a prediction horizon,  $h = 1, \dots, 9$ , the RMSE is defined as

$$\text{RMSE}(h) = \frac{\|\hat{x}_h - x_h\|_2}{\sqrt{q}},$$

where  $\hat{x}_h \in \mathbb{R}^q$  denotes the predicted intensity field,  $x_h$  is the corresponding ground-truth snapshot, and  $q$  is the spatial dimension of the grid. This metric quantifies the average per-cell prediction error and allows for a direct comparison of multi-step forecasting accuracy across methods.

The resulting RMSE curves over the prediction horizon are shown in [Fig. 8](#). As expected, the centralized Koopman model achieves the lowest

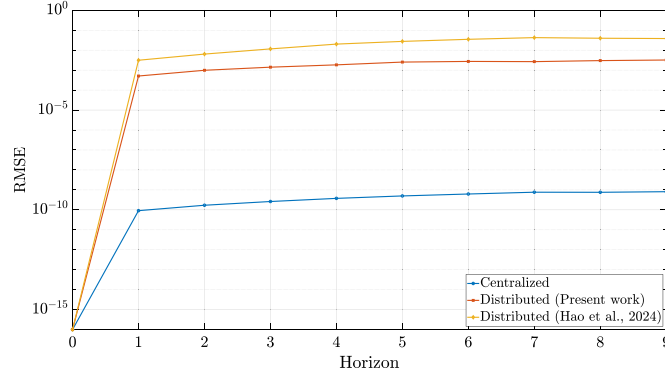


Fig. 8. RMSE over the prediction horizon for centralized and distributed Koopman learning methods: the proposed distributed method vs. the distributed baseline of Hao et al. (2024a)

error across all horizons. Both distributed methods exhibit stable multi-step prediction behavior on unseen data, indicating that the learned operators generalize beyond the training snapshots. Among the distributed approaches, our proposed method consistently attains lower prediction error than the baseline distributed method of Hao et al. (2024a) over the evaluated horizons, while preserving the distributed data acquisition and communication constraints.

## 5. Conclusion

This work proposed a distributed Koopman learning algorithm for modeling unknown nonlinear dynamics from sequential agent-level observations. Local Koopman models are constructed from lifted temporal data and refined via consensus to achieve consistent overall representations under distributed constraints. Simulation results demonstrate convergence and approximate model reconstruction using the proposed distributed learning method. In future, the learned Koopman operator can support real-time prediction of unknown motions, enabling proactive coordination and decision-making for dynamic obstacle avoidance in multi-agent scenarios.

### CRedit authorship contribution statement

**Ali Azarbahram:** Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Project administration, Methodology, Investigation, Formal analysis, Data curation, Conceptualization; **Shenyu Liu:** Writing – review & editing, Writing – original draft, Visualization, Validation, Methodology, Funding acquisition, Formal analysis, Conceptualization; **Gian Paolo Incremona:** Writing – review & editing, Writing – original draft, Visualization, Project administration, Investigation, Funding acquisition, Formal analysis, Conceptualization.

### Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

### Acknowledgements

This research was funded by the Italian Ministry of Enterprises and Made in Italy for the project 4DDS (4D Drone Swarms) under grant no. F/310097/01-04/X56, and the National Natural Science Foundation of China under grant no. 62203053.

### Appendix A.

The proof of Theorem 1 relies on the following Lemma.

**Lemma 1.** Consider a matrix

$$\tilde{\mathbf{M}} = \begin{bmatrix} -\mathbf{X}\mathbf{X}^\top - k_p\mathbf{L} & \sqrt{k_I}\mathbf{L}^{\frac{1}{2}} \\ -\sqrt{k_I}\mathbf{L}^{\frac{1}{2}} & 0 \end{bmatrix}. \quad (\text{A.1})$$

It holds that  $\Lambda(\mathbf{M}) = \Lambda(\tilde{\mathbf{M}})$ , where  $\mathbf{M}$  is defined in (13).

**Proof.** Since the matrix  $\mathbf{L}^{\frac{1}{2}}$  and  $I$  commute, we use Schur complement and determinantal formula (Horn & Johnson, 2012, Chapter 0.8.5) to conclude

$$\begin{aligned} \det(sI - \mathbf{M}) &= \det \left( \begin{bmatrix} sI + \mathbf{X}\mathbf{X}^\top + k_p\mathbf{L} & -\mathbf{L} \\ k_I I & sI \end{bmatrix} \right) \\ &= \det(s(sI + \mathbf{X}\mathbf{X}^\top + k_p\mathbf{L}) + k_I \mathbf{L}) \\ &= \det \left( \begin{bmatrix} sI + \mathbf{X}\mathbf{X}^\top + k_p\mathbf{L} & -\sqrt{k_I}\mathbf{L}^{\frac{1}{2}} \\ \sqrt{k_I}\mathbf{L}^{\frac{1}{2}} & sI \end{bmatrix} \right) \\ &= \det(sI - \tilde{\mathbf{M}}). \end{aligned}$$

□

**Proof of Theorem 1.** Similar to the  $\mathbf{K}$  defined in (8), define

$$\mathbf{R} := [R_1 \quad R_2 \quad \dots \quad R_p] \in \mathbb{R}^{n \times np}.$$

The update law (12b) and the initial state  $R_i(0) = 0_{n \times n}$  imply the existence of  $\tilde{\mathbf{R}}(k) \in \mathbb{R}^{n \times np}$  such that  $\mathbf{R}(k) = \frac{1}{\sqrt{k_I}} \tilde{\mathbf{R}}(k) \mathbf{L}^{\frac{1}{2}}$  for all  $k \in \mathbb{N}$ . Therefore, the update law (12) can be rewritten as

$$[\mathbf{K}^+ \quad \tilde{\mathbf{R}}^+] = [\mathbf{K} \quad \tilde{\mathbf{R}}](I + \alpha\tilde{\mathbf{M}}) + \alpha[Y\mathbf{X}^\top \quad 0], \quad (\text{A.2})$$

where we recall that the matrices  $\mathbf{X}, \tilde{\mathbf{M}}$  are defined in (9), (A.1). Meanwhile, it follows from the KKT condition (Boyd & Vandenberghe, 2004) that for any optimal solution  $\mathbf{K}^* \in \mathbb{R}^{n \times np}$  of the constrained problem (11a), there exists  $\Lambda^* \in \mathbb{R}^{n \times np}$  such that

$$\begin{aligned} -(Y - \mathbf{K}^*\mathbf{X})\mathbf{X}^\top + \Lambda^*\mathbf{L} &= 0, \\ \mathbf{K}^*\mathbf{L} &= 0. \end{aligned}$$

These equations can be compactly written as

$$\begin{bmatrix} \mathbf{K}^* & -\frac{1}{\sqrt{k_I}}\Lambda^*\mathbf{L}^{\frac{1}{2}} \end{bmatrix} \tilde{\mathbf{M}} + [Y\mathbf{X}^\top \quad 0] = 0. \quad (\text{A.4})$$

For each  $k \in \mathbb{N}$ , define  $\mathbf{E}(k) := [\mathbf{E}_K(k) \quad \mathbf{E}_R(k)]$ , where

$$\mathbf{E}_K(k) := \mathbf{K}(k) - \mathbf{K}^*, \quad \mathbf{E}_R(k) := \tilde{\mathbf{R}}(k) + \frac{1}{\sqrt{k_I}}\Lambda^*\mathbf{L}^{\frac{1}{2}}.$$

It follows from (A.2) and (A.4) that

$$\begin{aligned} \mathbf{E}^+ &= [\mathbf{K}^+ \quad \tilde{\mathbf{R}}^+] - \begin{bmatrix} \mathbf{K}^* & -\frac{1}{\sqrt{k_I}}\Lambda^*\mathbf{L}^{\frac{1}{2}} \end{bmatrix} \\ &= [\mathbf{K} \quad \tilde{\mathbf{R}}](I + \alpha\tilde{\mathbf{M}}) + \alpha[Y\mathbf{X}^\top \quad 0] - \begin{bmatrix} \mathbf{K}^* & \frac{1}{\sqrt{k_I}}\Lambda^*\mathbf{L}^{\frac{1}{2}} \end{bmatrix} \end{aligned}$$

$$\begin{aligned}
&= \begin{bmatrix} \mathbf{K} & \tilde{\mathbf{R}} \end{bmatrix} (I + \alpha \tilde{\mathbf{M}}) - \alpha \begin{bmatrix} \mathbf{K}^* & -\frac{1}{\sqrt{k_I}} \Lambda^* \mathbf{L}^{\frac{1}{2}} \end{bmatrix} \tilde{\mathbf{M}} - \begin{bmatrix} \mathbf{K}^* & \frac{1}{\sqrt{k_I}} \Lambda^* \mathbf{L}^{\frac{1}{2}} \end{bmatrix} \\
&= \begin{bmatrix} \mathbf{K} & \tilde{\mathbf{R}} \end{bmatrix} (I + \alpha \tilde{\mathbf{M}}) - \begin{bmatrix} \mathbf{K}^* & -\frac{1}{\sqrt{k_I}} \Lambda^* \mathbf{L}^{\frac{1}{2}} \end{bmatrix} (I + \alpha \tilde{\mathbf{M}}) \\
&= \mathbf{E}(I + \alpha \tilde{\mathbf{M}}), \tag{A.5}
\end{aligned}$$

which is a matrix-valued linear time-invariant system. Hence, we only need to analyze the stability properties of the matrix  $I + \alpha \tilde{\mathbf{M}}$ .

For any eigenvalue  $\lambda'$  of  $I + \alpha \tilde{\mathbf{M}}$ ,  $\lambda' = 1 + \alpha \text{Re}(\lambda) + \alpha \text{Im}(\lambda)j$ , where  $\lambda$  is an eigenvalue of  $\tilde{\mathbf{M}}$ . Hence

$$|\lambda'|^2 = (1 + \alpha \text{Re}(\lambda))^2 + \alpha^2 \text{Im}(\lambda)^2 = 1 + 2\alpha \text{Re}(\lambda) + \alpha^2 |\lambda|^2.$$

Meanwhile, by Liu (2024) (Lemma 7),  $\tilde{\mathbf{M}}$  is semi-Hurwitz; that is, all its eigenvalues either have negative real parts, or are 0 and non-defective (algebraic multiplicity equals to geometric multiplicity). Additionally, because of Lemma 1 and the assumption  $\alpha < \alpha_{\max}$ , where  $\alpha_{\max}$  is defined in (14), we either have  $\lambda = 0$  and non-defective so that  $\lambda' = 1$  and non-defective, or  $\text{Re}(\lambda) < 0$  and  $|\lambda'| < 1$ . This implies that  $I + \alpha \tilde{\mathbf{M}}$  has a Jordan decomposition of the form

$$I + \alpha \tilde{\mathbf{M}} = \begin{bmatrix} V_1 & V_0 \end{bmatrix} \begin{bmatrix} J_1 & \\ & I \end{bmatrix} \begin{bmatrix} U_1 \\ U_0 \end{bmatrix} =: V J U, \tag{A.6}$$

such that  $UV = I$ ,  $J_1 \in \mathbb{R}^{n' \times n'}$  for some  $n' < 2np$  is a block-diagonal matrix of Jordan blocks. Moreover,  $J_1$  is Schur (all eigenvalues are in the open unit disk) and the spectral radius of  $J_1$  is  $\rho_{\max}$ , defined in (15). Since  $\rho > \rho_{\max}$ ,  $\frac{1}{\rho} J_1$  is still Schur and hence by the discrete-time Lyapunov equation, there exists a symmetric positive definite matrix  $P \in \mathbb{R}^{n' \times n'}$  such that

$$\frac{1}{\rho^2} J_1 P J_1^T - P < 0, \tag{A.7}$$

where  $< 0$  means the matrix is negative definite. For each  $k \in \mathbb{N}$ , denote  $\mathbf{F}_1^+(k) := \mathbf{E}(k)V_1$ ,  $\mathbf{F}_0(k) := \mathbf{E}(k)V_0$ .

It follows from (A.5) and (A.6) that

$$\begin{aligned}
\mathbf{F}_0^+ &= \mathbf{E}^+ V_0 = \mathbf{E} \begin{bmatrix} V_1 & V_0 \end{bmatrix} \begin{bmatrix} J_1 & \\ & I \end{bmatrix} \begin{bmatrix} U_1 \\ U_0 \end{bmatrix} V_0 \\
&= \mathbf{E} V_0 = \mathbf{F}_0.
\end{aligned}$$

Hence  $\mathbf{F}(k) = \mathbf{F}(0)$ . Similarly,

$$\begin{aligned}
\mathbf{F}_1^+ &= \mathbf{E}^+ V_1 = \mathbf{E} \begin{bmatrix} V_1 & V_0 \end{bmatrix} \begin{bmatrix} J_1 & \\ & I \end{bmatrix} \begin{bmatrix} U_1 \\ U_0 \end{bmatrix} V_1 \\
&= \mathbf{E} V_1 J_1 = \mathbf{F}_1 J_1.
\end{aligned}$$

Then, it further follows from (A.7) that

$$\begin{aligned}
\|\mathbf{F}_1^+ P^{\frac{1}{2}}\|_F^2 &= \text{tr}(\mathbf{F}_1^+ P (\mathbf{F}_1^+)^T) = \text{tr}(\mathbf{F}_1 J_1 P J_1^T \mathbf{F}_1^T) \\
&\leq \rho^2 \text{tr}(\mathbf{F}_1 P \mathbf{F}_1^T) = \rho^2 \|\mathbf{F}_1 P^{\frac{1}{2}}\|_F^2,
\end{aligned}$$

where we have used the fact that the trace of positive semi-definite matrix is non-negative for the inequality above. Hence,  $\|\mathbf{F}_1(k) P^{\frac{1}{2}}\|_F \leq \rho^k \|\mathbf{F}_1(0) P^{\frac{1}{2}}\|_F$ . In other words,  $\mathbf{F}_1(k)$  converges to 0 exponentially with rate  $\rho$ . Because  $\mathbf{E} = \begin{bmatrix} \mathbf{F}_1 & \mathbf{F}_0 \end{bmatrix} U = \mathbf{F}_1 U_1 + \mathbf{F}_0 U_0 = \mathbf{F}_1 U_1 + \mathbf{E} V_0 U_0$ , we conclude that  $\mathbf{E}(k)$  converges to  $\mathbf{E}_\infty := \mathbf{E}(0) V_0 U_0$  exponentially with rate  $\rho$ . In addition,

$$\begin{aligned}
\mathbf{E}_\infty (I + \alpha \tilde{\mathbf{M}}) &= \mathbf{E}(0) V_0 U_0 \begin{bmatrix} V_1 & V_0 \end{bmatrix} \begin{bmatrix} J_1 & \\ & I \end{bmatrix} \begin{bmatrix} U_1 \\ U_0 \end{bmatrix} \\
&= \mathbf{E}(0) \begin{bmatrix} 0 & V_0 \end{bmatrix} \begin{bmatrix} J_1 & \\ & I \end{bmatrix} \begin{bmatrix} U_1 \\ U_0 \end{bmatrix} = \mathbf{E}(0) V_0 U_0 = \mathbf{E}_\infty,
\end{aligned}$$

which implies

$$\mathbf{E}_\infty \tilde{\mathbf{M}} = 0. \tag{A.8}$$

Recall the definition of  $\mathbf{E}$ . The solution of (12),  $\begin{bmatrix} \mathbf{K}(k) & \tilde{\mathbf{R}}(k) \end{bmatrix}$ , converges to

$$\begin{bmatrix} \mathbf{K}_\infty & \mathbf{R}_\infty \end{bmatrix} := \mathbf{E}_\infty + \begin{bmatrix} \mathbf{K}^* & -\frac{1}{\sqrt{k_I}} \Lambda^* \mathbf{L}^{\frac{1}{2}} \end{bmatrix}.$$

It then follows from (A.4) and (A.8) that

$$\begin{bmatrix} \mathbf{K}_\infty & \mathbf{R}_\infty \end{bmatrix} \tilde{\mathbf{M}} + \begin{bmatrix} Y X^T & 0 \end{bmatrix} = 0.$$

In other words,  $\mathbf{K}_\infty, \mathbf{R}_\infty$  also satisfy the KKT condition of optimality for (11a). Hence,  $\mathbf{K}_\infty$  is an optimal solution for the problem (11a), and  $\mathbf{K}_\infty = \begin{bmatrix} K_\infty & K_\infty & \cdots & K_\infty \end{bmatrix}$ , where  $K_\infty \in \mathbb{R}^{n \times n}$  is an optimal solution for the problem (7). This completes the proof.  $\square$

## References

- Azarbahram, A., Huanca, C. P. Y., Incremona, G. P., & Colaneri, P. (2025a). Distributed switching model predictive control meets Koopman operator for dynamic obstacle avoidance. Technical Report arXiv:2511.17186. arXiv preprint.
- Azarbahram, A., Liu, S., & Incremona, G. P. (2025b). Distributed Koopman operator learning for perception and safe navigation. Technical Report arXiv preprint.
- Boyd, S., & Vandenberghe, L. (2004). Convex optimization. Cambridge University Press.
- Bueno, V., Azarbahram, A., Farina, M., & Fagiano, L. (2025). Koopman-based dynamic environment prediction for safe UAV navigation. Technical Report arXiv:2511.06990. arXiv preprint.
- Cao, Y., Yu, W., Ren, W., & Chen, G. (2012). An overview of recent progress in the study of distributed multi-agent coordination. *IEEE Transactions on Industrial Informatics*, 9, 427–438.
- Comas, A., Ghimire, S., Li, H., Szanier, M., & Camps, O. (2021). Self-supervised decomposition, disentanglement and prediction of video sequences while interpreting dynamics: A Koopman perspective. Technical Report arXiv preprint.
- Gutow, G., & Rogers, J. D. (2020). Koopman operator method for chance-constrained motion primitive planning. *IEEE Robotics and Automation Letters*, 5, 1572–1578.
- Hansen, P. C., Pereyra, V., & Scherer, G. (2013). Least squares data fitting with applications. Johns Hopkins University Press.
- Hao, W., Lu, Z., Upadhyay, D., & Mou, S. (2024a). Distributed Koopman learning using partial trajectories for control. Technical Report arXiv preprint.
- Hao, W., Wang, L., Rai, A., & Mou, S. (2024b). Distributed Koopman learning with incomplete measurements. Technical Report arXiv preprint.
- Horn, R. A., & Johnson, C. R. (2012). Matrix analysis. New York, USA. Cambridge University Press.
- Huang, Y., Meng, Z., & Sun, J. (2022). Scalable distributed least square algorithms for large-scale linear equations via an optimization approach. *Automatica*, 146, 110572.
- Koopman, B. O. (1931). Hamiltonian systems and transformation in Hilbert space. *Proceedings of the National Academy of Sciences of the United States of America*, 17, 315–318. <https://doi.org/10.1073/pnas.17.5.315>
- Korda, M., & Mezić, I. (2018). Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control. *Automatica*, 93, 149–160. <https://doi.org/10.1016/j.automatica.2018.03.046>
- Korda, M., Putinar, M., & Mezić, I. (2020). Data-driven spectral analysis of the Koopman operator. *Applied and Computational Harmonic Analysis*, 48, 599–629. <https://doi.org/10.1016/j.acha.2018.08.002>
- Liu, S. (2024). Scalable distributed least squares algorithm for linear algebraic equations via scheduling. Technical Report arXiv preprint.
- Liu, Y., Lageman, C., Anderson, B. D. O., & Shi, G. (2019). An arrow-hurwitz-uzawa type flow as least squares solver for network linear equations. *Automatica*, 100, 187–193. <https://doi.org/10.1016/j.automatica.2018.10.007>
- Liu, Z., Ding, G., Chen, L., & Yeung, E. (2020). Towards scalable Koopman operator learning: Convergence rates and a distributed learning algorithm. In *2020 American control conference (ACC)* (pp. 3983–3990). IEEE.
- Lu, Y., Yao, W., Xiao, Y., & Xu, X. (2024). Vector field-guided learning predictive control for motion planning of mobile robots with unknown dynamics. Technical Report arXiv preprint.
- Mezić, I. (2005). Spectral properties of dynamical systems, model reduction and decompositions. *Nonlinear Dynamics*, 41, 309–325. <https://doi.org/10.1007/s11071-005-2824-x>
- Mezić, I. (2021). Koopman operator, geometry, and learning of dynamical systems. *Notices of the American Mathematical Society*, 68, 1087–1105. <https://doi.org/10.1090/noti2306>
- Mukherjee, S., Nandanoori, S. P., Guan, S., Agarwal, K., Sinha, S., Kundu, S., Pal, S., Wu, Y., Vrabie, D. L., & Choudhury, S. (2022). Learning distributed geometric Koopman operator for sparse networked dynamical systems. Technical Report PMLR.
- Nandanoori, S. P., Pal, S., Sinha, S., Kundu, S., Agarwal, K., & Choudhury, S. (2021). Data-driven distributed learning of multi-agent systems: A Koopman operator approach. In *2021 60th IEEE conference on decision and control* (pp. 5059–5066). IEEE.
- Oh, Y., Lee, M. H., & Moon, J. (2024). Koopman-based control system for quadrotors in noisy environments. *IEEE Access*, 12, 71675–71684.
- Rowley, C. W., Mezić, I., Bagheri, S., Schlatter, P., & Henningson, D. S. (2009). Spectral analysis of nonlinear flows. *Journal of Fluid Mechanics*, 641, 115–127. <https://doi.org/10.1017/S0022112009992059>
- Schmid, P. J. (2010). Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics*, 656, 5–28. <https://doi.org/10.1017/S0022112010001217>
- Schwager, M., Slotine, J. J., & Rus, D. (2011). Unifying geometric, probabilistic, and potential field approaches to multi-robot coverage control. In C. Pradaliere, R. Siegwart, & G. Hirzinger (Eds.), *Robotics research Berlin Heidelberg*: Berlin, Heidelberg (pp. 21–38). Springer.
- Wang, P., Mou, S., Lian, J., & Ren, W. (2019a). Solving a system of linear equations: From centralized to distributed algorithms. *Annual Reviews in Control*, 47, 306–322.

- Wang, X., Zhou, J., Mou, S., & Corless, M. J. (2019b). A distributed algorithm for least squares solutions. *IEEE Transactions on Automatic Control*, *64*, 4217–4222. <https://doi.org/10.1109/TAC.2019.2894588>
- Williams, M. O., Kevrekidis, I. G., & Rowley, C. W. (2015b). A data-driven approximation of the Koopman operator: Extending dynamic mode decomposition. *Journal of Nonlinear Science*, *25*, 1307–1346. <https://doi.org/10.1007/s00332-015-9258-5>
- Wulder, M. A., Masek, J. G., Cohen, W. B., Loveland, T. R., & Woodcock, C. E. (2012). Opening the archive: How free data has enabled the science and monitoring promise of LandsAT. *Remote Sensing of Environment*, *122*, 2–10.
- Yang, S., Wang, J., & Liu, Q. (2019). Consensus of heterogeneous nonlinear multiagent systems with duplex control laws. *IEEE Transactions on Automatic Control*, *64*, 5140–5147. <https://doi.org/10.1109/TAC.2019.2912533>
- Yang, T., George, J., Qin, J., Yi, X., & Wu, J. (2020). Distributed least squares solver for network linear equations. *Automatica*, *113*, 108798. <https://doi.org/10.1016/j.automatica.2019.108798>
- Zheng, Y., & Liu, Q. (2022). A review of distributed optimization: Problems, models and algorithms. *Neurocomputing*, *483*, 446–459. <https://doi.org/10.1016/j.neucom.2021.06.097>