Optimization and Learning in Open Multi-Agent Systems

Diego Deplano, *Member, IEEE*, Nicola Bastianello, *Member, IEEE*
Mauro Franceschelli, *Senior, IEEE*, and Karl H. Johansson, *Fellow, IEEE*

***Abstract*— Modern artificial intelligence relies on networks of agents that collect data, process information, and exchange it with neighbors to collaboratively solve optimization and learning problems. This article introduces a novel distributed algorithm to address a broad class of these problems in *open networks*, where the number of participating agents may vary due to several factors, such as autonomous decisions, heterogeneous resource availability, or DoS attacks. Extending the current literature, the convergence analysis of the proposed algorithm is based on the newly developed *Theory of Open Operators*, which characterizes an operator as open when the set of components to be updated changes over time, yielding time-varying operators acting on sequences of points of different dimensions and compositions. The mathematical tools and convergence results developed here provide a general framework for evaluating distributed algorithms in open networks, enabling characterization of their performance in terms of the punctual distance to the optimal solution, in contrast with regret-based metrics that assess cumulative performance over a finite-time horizon. As illustrative examples, the proposed algorithm is used to solve dynamic consensus and tracking problems on different metrics, such as average, median, and min/max, as well as classification problems with logistic loss functions.**

***Index Terms*— Open Operator Theory, ADMM, Open Networks, Open Multiagent Systems, Distributed optimization, Distributed Learning, Dynamic Consensus.**

I. Introduction

Many real-world systems consist of multiple interacting agents in a network, where new agents may join (start interacting) and others may leave (stop interacting), forming what is called an *open* multi-agent system [1–3]. Notions of open systems can be found in computer networks and communication systems, for instance, when referring to software agents and problems related to trust evaluation [4] and network management under churn [5, 6]. Dynamically evolving populations have also been considered in network games [7, 8] as, for instance, the Bitcoin network [9] where miners compete to gain a reward by adding a block of transactions to the blockchain, leaving after winning or when the profitability does not justify the costs, but also in social networks [10, 11] where people participate to share their thoughts and learn from others, leaving when they lose interest. Despite observing this dynamic behavior in practice, traditional cooperation schemes, optimization algorithms, game strategies, and learning techniques often assume a static network with a fixed set of participants. Under this simplified assumption, it is typically possible to prove stability and characterize performance in reaching equilibrium points, representing best-response strategies in games, optimal solutions in optimization, or finest trained models in machine learning. Such guarantees may lose their relevance in the context of open networks, because an equilibrium may never be reached and divergent behavior could arise due to the join–leave events which the network experiences.

Within the control and optimization community, only recently there has been growing interest in the study of open networks, as it is demonstrated by the high number of papers published within the last decade. In multi-robot systems, where adaptivity to addition/removal of robots is crucial, some architectures accommodate for dynamics teams but offer no performance guarantees [15] and only a few directly address the open scenario to ensure collision avoidance and connectivity maintenance [16–18]. Under the simplified assumption that the time-varying agent set belongs to a time-invariant superset, majority consensus [19] and practical consensus [20] were analyzed, as well as generalization of stability results for passivity systems [21] and input-to-state stable systems [22] to the open scenario. From a different perspective, scale-invariant quantities were exploited to study consensus problems in open networks, as for the max-consensus problem in [1] where the infinity norm is used to define a distance metric that does not depend on the number of agents, and the average consensus problem in [23, 24] where the normalized mean and variance metrics are used to study the expected asymptotic network's behavior under gossip interactions. Another interesting strategy is that of considering continuous approximations of large-graphs called graphons [25, 26], which may accommodate for

The work of Diego Deplano was supported by the project e.INS-Ecosystem of Innovation for Next Generation Sardinia (cod. ECS 00000038) funded by the Italian Ministry for Research and Education (MUR) under the National Recovery and Resilience Plan (NRRP) - MISSION 4 COMPONENT 2, “From research to business” INVESTMENT 1.5, “Creation and strengthening of Ecosystems of innovation” and construction of “Territorial R& D Leaders”.

The work of Nicola Bastianello and Karl H. Johansson was partially supported by the European Union's Horizon Research and Innovation Actions programme under grant agreement No. 101070162, and partially by Swedish Research Council Distinguished Professor Grant 2017-01078 Knut and Alice Wallenberg Foundation Wallenberg Scholar Grant.

Diego Deplano and Mauro Franceschelli are with DIEE, University of Cagliari, 09123 Cagliari, Italy. Emails: {diego.deplano,mauro.franceschelli}@unica.it

Nicola Bastianello and Karl H. Johansson are with the School of Electrical Engineering and Computer Science, and Digital Futures, KTH Royal Institute of Technology, Stockholm, Sweden. Emails: {nicolba,kallej}@kth.se

TABLE I
COMPARISON WITH THE STATE OF THE ART FOR DISTRIBUTED OPTIMIZATION AND LEARNING IN OPEN MULTI-AGENT SYSTEMS WITH DISCRETE-TIME DYNAMICS.

[Ref.]	Algorithm	Assumptions on the problem	Assumptions on the network	Time independent parameters	Convergence metric	Convergence rate
[12] Hendrickx *et al.* (2020)	DGD	Static + Smooth + Strongly convex + Minimizers in a ball	✗ Only replacement	✓	Distance from minimizers	Linear (inexact)
[13] Hsieh *et al.* (2021)	Dual averaging	Static + Lipschitz + Convex + Shared convex constraint set	≈ Vertex-connected† (jointly)	✗	Regret	Sublinear (if the network's size is known)
[14] Hayashi (2023)	Sub-gradient	Time-varying + Lipschitz + Convex Shared compact constraint set	≈ Vertex-connected† (jointly)	✗	Regret	Sublinear (if the network's size is bounded)
[This work]	ADMM	Time-varying + Semicontinuous + Convex Unconstrained	✓ Connected	✓	Distance from minimizers	Linear (exact)

†A graph $\mathcal{G}_k = (\mathcal{V}_k, \mathcal{E}_k)$ is said to be jointly vertex connected if there exists $B, \kappa \geq 1$ such that at least κ nodes need to be removed to disrupt the connectivity of the union graph $\left(\bigcup_{t=k-B}^{k} \mathcal{V}_t, \bigcup_{t=k-B}^{k} \mathcal{E}_t\right)$ for all $k \in \mathbb{N}$.

open agent sets. Major topics of interest are represented by distributed consensus protocols, both in continuous-time [27, 28] and in discrete-time [2, 29–37], distributed optimization algorithms [12–14, 38, 39], distributed resource allocation [40–42], and learning problems [43, 44]. The difficulty in designing distributed algorithms that can be deployed over open networks, and in providing formal performance guarantees, is mainly due to the lack of mathematical tools to analyze the dynamics of systems with a varying number of state variables.

The **first contribution** of this manuscript is to fill this theoretical gap by formalizing fundamental concepts for what we call *open operators*, i.e., time-varying operators acting on sequences of points of different dimensions and compositions. We extend concepts for standard operators [45, 46] – such as distance between points and sets, projections of points into sets, fixed points, and convergence of the iteration toward fixed points – to the set-up of open operators. Pioneering ideas and results in this direction appeared in [2], where the distance between points of different dimensions was defined by accounting for elements in the union of their domains, whereas this manuscript considers the intersection, as recently proposed in [1]. Moreover, this manuscript accounts for agents with vector states rather than just scalar states as in [2]. Since exact convergence cannot be reached in general due to the varying nature of the set of fixed points, this manuscript provides sufficient conditions for the class of paracontractive operators – a superclass of contractive operators considered in [2] – ensuring the convergence of the iteration to a bounded distance from the fixed point set.

The **second contribution** of this manuscript is the presentation and characterization of an open and distributed version of ADMM to solve optimization and learning problems over open networks. We call the proposed algorithm *Open ADMM* which enjoys the following advantages compared with other state-of-the-art algorithms (see Table I): 1) the algorithm accommodates arbitrary changes in the network with a time-varying number of nodes that can grow unbounded; 2) it never needs a centralized re-initialization procedure as the algorithm parameters are neither vanishing nor time-varying; 3) it works with time-varying cost functions that are only convex and not necessarily strongly convex; and 4) it converges linearly to the set of minimizers, achieving exact convergence if the network composition and the local costs remain unchanged for a sufficiently long time. Open ADMM is an extension of DOT-ADMM (presented by us in [47]) for the scenario of open multi-agent systems of interest in this manuscript. Assuming all cost functions are static, smooth, strongly convex, and that their minimizers lie within a given ball, the authors of [12] have shown that the Decentralized Gradient Descent (DGD) Algorithm remains stable in the constrained scenario where departing agents are immediately replaced by new arrivals. Relaxing strong convexity to convexity and smoothness to Lipschitz continuity, the authors of [13] proposed a dual averaging method and proved sublinear convergence of the running loss, provided the running ratio of the quadratic mean to the average number of active agents is bounded, which accommodates unbounded network growth. Under the stronger assumption that the network's size is bounded from above, the author of [14] introduced a subgradient approach and proved sublinear convergence of the running loss while accounting with time-varying local loss functions. Unlike this paper and [12], these works analyze performance using a regret-based metric and require the network to be (jointly) vertex-connected. Instead, this work only assumes the network to be connected and characterizes the punctual distance of the local estimates from the global optimum, making the performance evaluation less coarse and more informative than regret-based metrics: this claim follows by the fact that sublinear regret alone does not guarantee convergence of the estimator to the global optimum, nor that it remains within a bounded distance of it.

As a **third contribution**, the proposed Open ADMM algorithm is applied to solve dynamic consensus problems over open networks. Specifically, we derive closed-form updates for Open ADMM and establish sufficient conditions on the signals being tracked to ensure the correct tracking of three different metrics of interest: the maximum, the median, and the average of a set of signals, each of which is locally accessible to the agents. Notably, for the median metric, this work

This article has been accepted for publication in IEEE Transactions on Automatic Control. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TAC.2026.3653296

introduces the first discrete-time protocol in the literature designed for open scenarios. Additionally, the proposed approach demonstrates superior performance for both the maximum and average metrics. We refer the interested reader to Section IV-A and Remark 3 for a comparison of performance and working assumption between our algorithms and those in the literature.

As a **fourth contribution**, the proposed algorithm is applied to supervised learning problems with logistic loss functions, and numerical simulations are carried out to demonstrate performance in different scenarios: (i) networks with Poisson arrivals and departures, the distance from the optimal solution is proportional to the Poisson rates but remains bounded; (ii) eventually closed, fixed networks, the algorithm achieves exact convergence at steady state; (iii) networks with only node replacements, larger networks exhibit higher robustness. Open ADMM has been benchmarked against state-of-the-art algorithms, demonstrating superior performance in terms of both convergence rate and tracking error. Furthermore, the use of stochastic gradients and alternative initialization methods is explored to address cases where computing local optima is computationally expensive, as in very large datasets.

Structure of the manuscript: Section II introduces all relevant concepts of open operator theory, generalizing the concepts of fixed points, sequences and their convergence, projections, and distances. Section III formalizes the problem of interest together with the working assumptions, then presents and characterizes the proposed Open ADMM algorithm. Section IV exploits Open ADMM to solve the tracking problem (or dynamic consensus) over the maximum, median, and average functions, and supervised learning problems with logistic loss function. Section V concludes the paper by discussing some future research directions.

II. Open Operator Theory

The set of real and integer numbers are denoted by $\mathbb{R}$ and $\mathbb{Z}$, respectively, while $\mathbb{R}_+$ and $\mathbb{N}$ denote their restriction to positive entries. Matrices are denoted by uppercase letters, vectors and scalars by lowercase letters, while sets and spaces are denoted by uppercase calligraphic letters. The identity matrix is denoted by I_n, $n \in \mathbb{N}$, while the vectors of ones and zeros are denoted by $\mathbb{1}_n$ and $\mathbb{0}_n$; subscripts are omitted if clear from the context. We use the symbol $\otimes$ to denote the Kronecker product. Letting $\mathcal{I} \neq \emptyset$ be a finite set of labels, we adopt the non-standard yet intuitive notation $\mathbb{R}^{\mathcal{I}}$ to denote a vector space of finite dimension equal to the number of elements of $\mathcal{I}$, and by $x \in \mathbb{R}^{\mathcal{I}}$ we denote a vector with labeled components $x_i \in \mathbb{R}$ where each label $i \in \mathcal{I}$ corresponds to a label in $\mathcal{I}$. We limit our discussion to finite-dimensional Euclidean normed spaces $(\mathbb{R}^{\mathcal{I}}, \|\cdot\|_2)$ and denote the distance between two points $x, y \in \mathbb{R}^{\mathcal{I}}$ with the same labeled components by $d : \mathbb{R}^{\mathcal{I}} \times \mathbb{R}^{\mathcal{I}} \to \mathbb{R}_+$, given by $d(x, y) = \|x - y\|_2$. In turn, the distance of a point $x \in \mathbb{R}^{\mathcal{I}}$ from a set $\mathcal{X} \subseteq \mathbb{R}^{\mathcal{I}}$ and the distance between two sets $\mathcal{X}, \mathcal{Y} \subseteq \mathbb{R}^{\mathcal{I}}$ are defined as follows

$$d(x, \mathcal{Y}) = \inf_{y \in \mathcal{Y}} d(x, y), \quad d(\mathcal{X}, \mathcal{Y}) = \inf_{x \in \mathcal{X}} d(x, \mathcal{Y}). \tag{1}$$

In the remainder of this section, we introduce all relevant concepts about operators $\mathsf{T} : \mathcal{X} \to \mathcal{Y}$ mapping spaces $\mathcal{X} \subseteq \mathbb{R}^{\mathcal{I}_\mathcal{X}}$ into spaces $\mathcal{Y} \subseteq \mathbb{R}^{\mathcal{I}_\mathcal{Y}}$ where $\mathcal{I}_\mathcal{X}$, $\mathcal{I}_\mathcal{Y}$ are sets of labels with different cardinality and composition: such operators will be called *open* when $\mathcal{X} \neq \mathcal{Y} \neq 0$ or, equivalently, $\mathcal{I}_\mathcal{X} \neq \mathcal{I}_\mathcal{Y}$.

A. Open-operators and sets of interest

Let $\mathcal{I}_k$ denote the set of labels at time $k \in \mathbb{N}$. According to Fig. 1, we identify the following subsets:

- *Remaining labels* $\mathcal{R}_k = \mathcal{I}_k \cap \mathcal{I}_{k-1}$: labels present both at time $k-1$ and time k;
- *Arriving labels* $\mathcal{A}_k = \mathcal{I}_k \setminus \mathcal{I}_{k-1}$: labels present at time k but not at time $k-1$;
- *Departing labels* $\mathcal{D}_k = \mathcal{I}_k \setminus \mathcal{I}_{k+1}$: labels present at time k but not at time $k+1$.

By convention, $\mathcal{I}_{-1} = \emptyset$, yielding $\mathcal{R}_0 = \emptyset$ and $\mathcal{A}_0 = \mathcal{I}_0$. Note that, in general, the set of departing labels may contain both remaining and arriving labels, which are instead disjoint:

$$\mathcal{D}_k \subset \mathcal{I}_k = \mathcal{R}_k \cup \mathcal{A}_k, \qquad \mathcal{R}_k \cap \mathcal{A}_k = \emptyset.$$

Example 1. *Consider the sets of labels* $\mathcal{I}_0 = \{a, b, c\}$, $\mathcal{I}_1 = \{a, b, c, d\}$, $\mathcal{I}_2 = \{b, c, d\}$, $\mathcal{I}_3 = \{e, f\}$. *Then:*

- *(at the initial step* 0*)* $D_{-1} = \emptyset$, $\mathcal{A}_0 = \{a, b, c\}$, $\mathcal{R}_0 = \emptyset$;
- *(from step* 0 *to step* 1*)* $D_0 = \emptyset$, $\mathcal{A}_1 = \{d\}$, $\mathcal{R}_1 = \{a, b, c\}$;
- *(from step* 1 *to step* 2*)* $D_1 = \{a\}$, $\mathcal{A}_2 = \emptyset$, $\mathcal{R}_2 = \{b, c, d\}$;
- *(from step* 2 *to step* 3*)* $D_2 = \{b, c, d\}$, $\mathcal{A}_3 = \{e, f\}$, $\mathcal{R}_3 = \emptyset$.

An *open* operator $\mathsf{T}_k : \mathbb{R}^{\mathcal{I}_{k-1}} \to \mathbb{R}^{\mathcal{I}_k}$ maps points x_{k-1} with components x_{k-1}^i labeled by $i \in \mathcal{I}_{k-1}$ into points x_k with components x_k^i labeled by $i \in \mathcal{I}_k$, which may have different size. The sequence $\{x_k : k \in \mathbb{N}\}$ generated by the iteration of T_k is such that the points $x_k \in \mathbb{R}^{\mathcal{I}_k}$ may have different dimensions for $k \in \mathbb{N}$, and thus are called *open sequences*. Since x_k takes values in $\mathbb{R}^{\mathcal{I}_k}$ at all times k, departing components with label $i \in \mathcal{D}_{k-1}$ are simply left out from x_k. Instead, new components with label $i \in \mathcal{A}_k$ are initialized to some value $x_k^{\mathrm{A},i} \in \mathbb{R}$, and remaining components with label $i \in \mathcal{R}_k$ are updated according to a scalar operator $\mathsf{F}_k^i : \mathbb{R}^{\mathcal{I}_{k-1}} \to \mathbb{R}$. Thus, the iteration of the open operator T_k can be written component-wise for each $i \in \mathcal{I}_k$, as follows:

$$x_k^i = \mathsf{T}_k^i(x_{k-1}) = \begin{cases} \mathsf{F}_k^i(x_{k-1}) & \text{if } i \in \mathcal{R}_k \\ x_k^{\mathrm{A},i} & \text{if } i \in \mathcal{A}_k, \end{cases} \tag{2}$$

which, in compact form, becomes

$$x_k = \mathsf{T}_k(x_{k-1}), \qquad x_k \in \mathbb{R}^{\mathcal{I}_k}. \tag{3}$$

Note that if the components of the system remain unchanged, that is, $\mathcal{I}_k = \mathcal{I}_{k-1}$, then the operator $\mathsf{F}_k : \mathbb{R}^{\mathcal{I}_{k-1}} \to \mathbb{R}^{\mathcal{I}_{k-1}}$ rules the so-called *standard iteration*:

$$x_k = \mathsf{F}_k(x_{k-1}), \quad \text{when} \quad \mathcal{I}_k = \mathcal{I}_{k-1}. \tag{4}$$

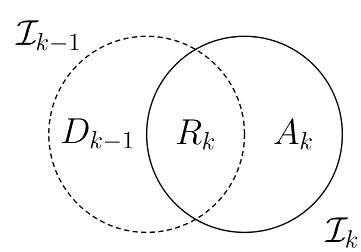

Fig. 1. Venn diagram of labels at two consecutive steps $k-1$ and k.

This article has been accepted for publication in IEEE Transactions on Automatic Control. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TAC.2026.3653296

Example 2. *Consider the open operator* T_k *as in* (2) *ruled by the averaging operator* $\mathsf{F}_k^i : x_{k-1} \mapsto F_k^i x_k$ *defined by*

$$F_k^i = \frac{1}{|\mathcal{I}_{k-1}|}\mathbb{1}_{|\mathcal{I}_{k-1}|}^\top, \qquad \forall i \in \mathcal{I}_k.$$

If the set of labels evolves as in Example 1, then a possible open sequence generated by the iteration (3) *is:*

$$\underbrace{\begin{bmatrix} x_0^a \\ x_0^b \\ x_0^c \end{bmatrix}}_{x_0} = \begin{bmatrix} 2 \\ 3 \\ 4 \end{bmatrix}, \quad \underbrace{\begin{bmatrix} x_1^a \\ x_1^b \\ x_1^c \\ x_1^d \end{bmatrix}}_{x_1} = \begin{bmatrix} 3 \\ 3 \\ 3 \\ 7 \end{bmatrix}, \quad \underbrace{\begin{bmatrix} x_2^b \\ x_2^c \\ x_2^d \end{bmatrix}}_{x_2} = \begin{bmatrix} 4 \\ 4 \\ 4 \end{bmatrix}, \quad \underbrace{\begin{bmatrix} x_3^e \\ x_3^f \end{bmatrix}}_{x_3} = \begin{bmatrix} 0 \\ 1 \end{bmatrix},$$

where:

- *(from step* 0 *to step* 1*) all components associated with remaining labels update their value to the average of* x_0*, that is* $(2+3+4)/3 = 3$*; label* d *arrives, and component* x_1^d *is initialized at 7;*
- *(from step* 1 *to step* 2*) all components associated with remaining labels update their value to the average of* x_1*, that is* $(3+3+3+7)/4 = 4$*; label* a *departs, and component* x_2^a *is left out;*
- *(from step* 2 *to step* 3*) labels* b, c, d *depart, and components* x_3^b*,* x_3^c*,* x_3^d *are left out; labels* e, f *arrive, and components* x_3^e*,* x_3^f *are initialized at* 0*,* 1*, respectively.*

The need to generalize the usual concept of *set of fixed points* $\hat{\mathcal{X}} = \{x : x = \mathsf{F}(x)\}$ for time-invariant standard operators $\mathsf{F} : \mathbb{R}^{\mathcal{I}} \to \mathbb{R}^{\mathcal{I}}$ arises naturally. Since for open operators $\mathsf{T}_k \in \mathbb{R}^{\mathcal{I}_{k-1} \to \mathcal{I}_k}$ the fixed point relation $x_{k-1} = \mathsf{T}_k(x_{k-1})$ is not well-posed if $\mathcal{I}_{k-1} \neq \mathcal{I}_k$, it is necessary to consider instead the standard iteration $\mathsf{F}_k : \mathbb{R}^{\mathcal{I}_{k-1}} \to \mathbb{R}^{\mathcal{I}_{k-1}}$, for which $x_{k-1} = \mathsf{F}_k(x_{k-1})$ is always well-posed. This yields a trajectory of sets containing fixed points of different dimension, which we call the *trajectory of sets of interest*.

Definition 1 (Trajectory of sets of interest). *Consider the iteration of an open operator* T_k *as in* (2) *and let* $\hat{\mathcal{X}}_k$ *be the set of fixed points of the operator* F_{k+1} *ruling the standard iteration, i.e.,*

$$\hat{\mathcal{X}}_k := \{x \in \mathbb{R}^{\mathcal{I}_k} \mid x = \mathsf{F}_{k+1}(x)\}. \tag{5}$$

Then, the sequence $\{\hat{\mathcal{X}}_k : k \in \mathbb{N}\}$ *is called the "trajectory of sets of interest" (TSI) of the operator* T_k*.*

If each operator F_k has a unique equilibrium point $\hat{\mathcal{X}}_k = \{\hat{x}_k\}$, then the sequence $\{\hat{x}_k : k \in \mathbb{N}\}$ of these points forms the *trajectory of points of interest* [2, Definition 3.1].

Example 3. *Consider the iteration described in Example 2. Then, the TSI is given by*

$$\hat{\mathcal{X}}_k = \{\alpha\mathbb{1} \in \mathbb{R}^{\mathcal{I}_k} \mid \alpha \in \mathbb{R}\}.$$

Indeed, at each $k \in \mathbb{N}$*, the set of fixed points of the standard iteration ruled by* $\mathsf{F}_k : x \mapsto F_k x$ *consists of the eigenvectors of* F_k *corresponding to the (unique) unitary eigenvalue.*

We formalize the concept of convergence for open sequences, emphasizing that comparing distances in spaces of different dimensions is unfair without normalization by the corresponding space's dimension. Without normalization, iterations of an open operator with increasing number of components may produce sequences diverging from the TSI, even if the distance between new components and their TSI counterparts remains bounded. We define convergence based on normalized distance from the TSI and provide an example clarifying the importance of this normalization.

Definition 2 (Convergence of open sequences). *Consider the open sequence* $\{x_k \in \mathbb{R}^{\mathcal{I}_k} : k \in \mathbb{N}\}$ *generated by the iteration of an open operator as in* (3) *whose TSI is* $\{\hat{\mathcal{X}}_k \subseteq \mathbb{R}^{\mathcal{I}_k} : k \in \mathbb{N}\}$*. The open sequence "converges" to the TSI within a radius* $R \geq 0$ *if*

$$\limsup_{k \to \infty} \frac{d(x_k, \hat{\mathcal{X}}_k)}{\sqrt{|\mathcal{I}_k|}} \leq R.$$

Example 4. *Consider the iteration described in Example 2 and assume that the components are totally renewed at each step and increase in number by 2 at each step, namely,* $D_k = \mathcal{I}_k$ *and* $|\mathcal{A}_k| = |\mathcal{I}_{k-1}| + 2$*. A possible open sequence generated by the iteration* $x_k = \mathsf{T}_k(x_{k-1})$ *is*

$$x_0 = \begin{bmatrix} +1 \\ -1 \end{bmatrix}, \quad x_1 = \begin{bmatrix} +1 \\ +1 \\ -1 \\ -1 \end{bmatrix}, \quad x_2 = \begin{bmatrix} +1 \\ +1 \\ +1 \\ -1 \\ -1 \\ -1 \end{bmatrix}, \quad \cdots,$$

where half of the components are initialized at $+1$ *and the other half at* -1*. The point* $\hat{x}_k \in \hat{\mathcal{X}}_k$ *in the TSI attaining the minimum distance from* x_k *is the null vector* $\hat{x}_k = \mathbb{0}_{2(k+1)}$ *and therefore*

$$d(x_k, \hat{\mathcal{X}}_k) = d(x_k, \hat{x}_k) = d(x_k, \mathbb{0}_{2(k+1)}) = \sqrt{2(k+1)},$$

which, as $k \to \infty$*, diverges even though the new components have bounded distance of* 1 *from the corresponding component of the TSI. Instead, when normalization is considered, it is possible to find a finite upper bound*

$$\limsup_{k \to \infty} \frac{d(x_k, \hat{\mathcal{X}}_k)}{\sqrt{2(k+1)}} = 1.$$

B. Open distances and projections

A notion of *open distance* is necessary to evaluate the distance between points with labeled components that belong to vector spaces of different dimensions. We propose to evaluate this distance by considering only the common labeled components, disregarding the others.

Definition 3 (Open distance). *Let* $\mathcal{I}_1$ *and* $\mathcal{I}_2$ *be two finite sets of labels. If* $\mathcal{I}_1 \cap \mathcal{I}_2 \neq \emptyset$*, the "open distance"* $d : \mathbb{R}^{\mathcal{I}_1} \times \mathbb{R}^{\mathcal{I}_2} \mapsto \mathbb{R}_+$ *between points* $x \in \mathbb{R}^{\mathcal{I}_1}$ *and* $y \in \mathbb{R}^{\mathcal{I}_2}$ *is defined by*

$$d(x,y) = \|\tilde{x} - \tilde{y}\|_2, \quad \text{where} \quad \begin{cases} \tilde{x} &= [x^i \text{ for } i \in \mathcal{I}_1 \cap \mathcal{I}_2], \\ \tilde{y} &= [y^i \text{ for } i \in \mathcal{I}_1 \cap \mathcal{I}_2]. \end{cases}$$

Otherwise, if $\mathcal{I}_1 \cap \mathcal{I}_2 = \emptyset$ *then* $d(x,y) = 0$*.*

This article has been accepted for publication in IEEE Transactions on Automatic Control. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TAC.2026.3653296

Example 5. *Consider the sets of labels $\mathcal{I}_1 = \{a, b, c\}$ and $\mathcal{I}_2 = \{b, c, d\}$ and let*

$$x = \begin{bmatrix} x^a \\ x^b \\ x^c \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \in \mathbb{R}^{\mathcal{I}_1}, \quad y = \begin{bmatrix} y^b \\ y^c \\ y^d \end{bmatrix} = \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix} \in \mathbb{R}^{\mathcal{I}_1}.$$

Since the common components are those labeled by b and c, the distance between them is given by

$$d(x, y) = \left\| \begin{bmatrix} x^b \\ x^c \end{bmatrix} - \begin{bmatrix} y^b \\ y^c \end{bmatrix} \right\| = \left\| \begin{bmatrix} 2 \\ 3 \end{bmatrix} - \begin{bmatrix} 4 \\ 5 \end{bmatrix} \right\| = \sqrt{8}.$$

For the sake of simplicity, we overload the notation of standard distance in a way that the distance of a point $x \in \mathbb{R}^{\mathcal{I}_1}$ from a set $\mathcal{X} \subseteq \mathbb{R}^{\mathcal{I}_2}$ and the distance between two sets $\mathcal{X} \subseteq \mathbb{R}^{\mathcal{I}_2}$, $\mathcal{Y} \subseteq \mathbb{R}^{\mathcal{I}_3}$ of different dimension are naturally defined as in (1). The projection of a point $x \in \mathbb{R}^{\mathcal{I}_1}$ over a non-empty set $\mathcal{Y} \subseteq \mathbb{R}^{\mathcal{I}_2}$ is a set of points $y \in \mathcal{Y}$ of minimum distance from x, given by the *projection operator* defined as

$$\text{proj}(x, \mathcal{Y}) = \left\{ y^\star \in \mathcal{Y} : \ d(x, y^\star) = \inf_{y \in \mathcal{Y}} \ d(x, y) \right\}$$

When $\mathcal{I}_1 \supseteq \mathcal{I}_2$ the projection reduces to a singleton (see Fig. 2), but when $\mathcal{I}_2 \setminus \mathcal{I}_1 \neq 0$ it is indeed a set (see Fig. 3) because components with label in $\mathcal{I}_2$ that are not in $\mathcal{I}_1$ can be arbitrary within $\mathcal{Y}$.

Example 6. *Let $\mathcal{I}_1 = \{a, b\}$, $\mathcal{I}_2 = \{a\}$, and $\mathcal{Y} = \{y \in \mathbb{R}^{\mathcal{I}_2} \mid y \in [1, 3]\}$ as in Fig. 2. Then, the distance of $x = [x^a, x^b]^\top = [5, 1]^\top \in \mathbb{R}^{\mathcal{I}_1}$ from $\mathcal{Y}$ is*

$$d(x, \mathcal{Y}) = d(x^a, \mathcal{Y}) = \min_{y \in \mathcal{Y}} |5 - y| = |5 - 3| = 2,$$

and $\text{proj}(x, \mathcal{Y}) = 3 \in \mathbb{R}^{\mathcal{I}_2}$ is a singleton. Instead, the distance of $x = [x^a, x^b]^\top = [2, 1]^\top \in \mathbb{R}^{\mathcal{I}_1}$ from $\mathcal{Y}$ is

$$d(x, \mathcal{Y}) = d(x^a, \mathcal{Y}) = \min_{y \in \mathcal{Y}} |2 - y| = 0,$$

and $\text{proj}(x, \mathcal{Y}) = 2 \in \mathbb{R}^{\mathcal{I}_2}$ is a singleton.

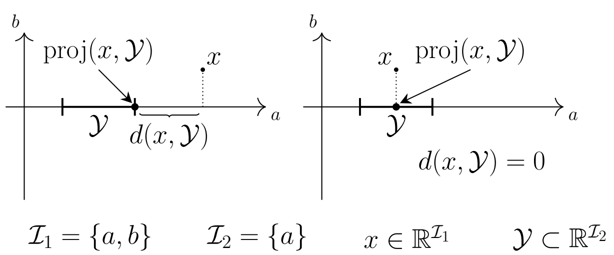

Fig. 2. Representation of the open distance (curly brace) and the open projection (bold dot) in two examples with $\mathcal{I}_2 \subseteq \mathcal{I}_1 \neq \emptyset$.

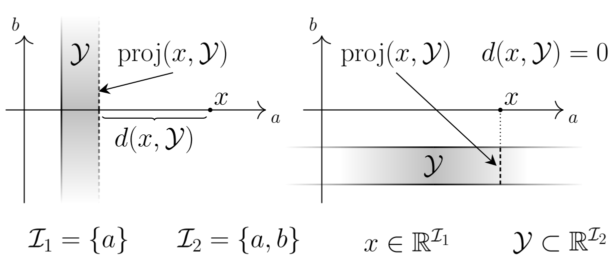

Fig. 3. Representation of the open distance (curly brace) and the open projection (dashed line) in two examples with $\mathcal{I}_2 \setminus \mathcal{I}_1 \neq \emptyset$.

Example 7. *Let $\mathcal{I}_1 = \{a\}$, $\mathcal{I}_2 = \{a, b\}$, and $x = 5 \in \mathbb{R}^{\mathcal{I}_1}$ as in Fig. 3. Then, the distance of x from $\mathcal{Y} = \{[y^a, y^b] \in \mathbb{R}^{\mathcal{I}_2} \mid y^a \in [1, 2]\}$ is*

$$d(x, \mathcal{Y}) = \min_{[y^a, y^b] \in \mathcal{Y}} |5 - y^a| = |5 - 2| = 3,$$

and $\text{proj}(x, \mathcal{Y}) = \{[2, z^b]^\top \in \mathbb{R}^{\mathcal{I}_2}\}$ is a set. Instead, the distance of x from $\mathcal{Y} = \{[y^a, y^b] \in \mathbb{R}^{\mathcal{I}_2} \mid y^b \in [-2, -1]\}$ is

$$d(x, \mathcal{Y}) = d(x^a, \mathcal{Y}) = \min_{[y^a, y^b] \in \mathcal{Y}} |2 - y^a| = 0,$$

and $\text{proj}(x, \mathcal{Y}) = \{[5, z^b]^\top \in \mathbb{R}^{\mathcal{I}_2} \mid z^b \in [-2, -1]\}$ is a set.

The projection operator is a particular case of the proximal operator which, for a point $x \in \mathbb{R}^{\mathcal{I}_1}$, a function $f : \mathbb{R}^{\mathcal{I}_2} \to \mathbb{R}$, and penalty parameter $\varepsilon > 0$, is defined by

$$\text{prox}_f^\varepsilon(x) = \left\{ y^\star : f(y^\star) + \frac{d^2(x, y^\star)}{2\varepsilon} = \min_{y \in \mathbb{R}^{\mathcal{I}_2}} \left\{ f(y) + \frac{d^2(x, y)}{2\varepsilon} \right\} \right\}.$$

Indeed, it holds $\text{proj}(x, \mathcal{Y}) = \text{prox}_{\iota_\mathcal{Y}}^\varepsilon(x)$ where $\iota_\mathcal{Y} : \mathbb{R}^n \to \mathbb{R}^n \cup \{+\infty\}$ is the indicator function defined as $\iota_\mathcal{Y}(x) = 0$ if $y \in \mathcal{Y} \subseteq \mathbb{R}^{\mathcal{I}_2}$, and $\iota_\mathcal{Y}(y) = +\infty$ otherwise.

We also introduce the concept of *shadow distance* between two spaces $\mathcal{X}, \mathcal{Y}$ of different cardinality or composition, that is the maximum distance between any pair of projections.

Definition 4 (Shadow distance). *Let $\mathcal{I}_1$ and $\mathcal{I}_2$ be two finite sets of labels. If $\mathcal{I}_1 \cap \mathcal{I}_2 \neq \emptyset$, the "shadow distance" $d : \mathbb{R}^{\mathcal{I}_1} \times \mathbb{R}^{\mathcal{I}_2} \mapsto \mathbb{R}_+$ between points $x \in \mathbb{R}^{\mathcal{I}_1}$ and $y \in \mathbb{R}^{\mathcal{I}_2}$ is defined by*

$$d_{\text{SH}}(\mathcal{X}, \mathcal{Y}) = \sup_{z \in \mathbb{R}^{\mathcal{I}_1 \cup \mathcal{I}_2}} d(\text{proj}(z, \mathcal{X}), \text{proj}(z, \mathcal{Y})).$$

The concept of shadow distance is essential as it allows formulating the following version of triangle inequality:

$$d(z, \mathcal{X}) \leq d(z, \mathcal{Y}) + d_{\text{SH}}(\mathcal{X}, \mathcal{Y}). \quad (6)$$

Example 8. *Consider the set of labels $\mathcal{I} = \{a\}$ and the sets $\mathcal{X} = \{x \in \mathbb{R}^\mathcal{I} : x \in [1, 2]\}$, $\mathcal{Y} = \{y \in \mathbb{R}^\mathcal{I} : y \in [5, 6]\}$. The standard distance between these sets is attained by the points $x = 2$ and $y = 5$, namely,*

$$d(\mathcal{X}, \mathcal{Y}) = \inf_{x \in \mathcal{X}} \ \inf_{y \in \mathcal{Y}} \ d(x, y) = |2 - 5| = 3.$$

On the other hand, the shadow distance (see Fig. 4) is attained for all points $z \in \mathbb{R}^\mathcal{I}$ such that $z \in (-\infty, 1] \cup [6, +\infty)$. For instance, the projection of $z \leq 1$ onto $\mathcal{X}$ is 1 and the projection onto $\mathcal{Y}$ is 5, yielding

$$d_{\text{SH}}(\mathcal{X}, \mathcal{Y}) = \sup_{z \in \mathbb{R}^\mathcal{I}} d(\text{proj}(z, \mathcal{X}), \text{proj}(z, \mathcal{Y})) = |1 - 5| = 4.$$

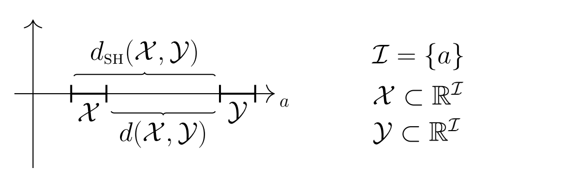

Fig. 4. Representation of the shadow distance compared to the standard distance in an example with $\mathcal{I} = \mathcal{I}_1 = \mathcal{I}_2$.

This article has been accepted for publication in IEEE Transactions on Automatic Control. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TAC.2026.3653296

C. A convergence result for paracontractive operators

In this section, we provide sufficient conditions ensuring the convergence of open sequences generated by the iteration of open operators – in the sense of Definition 2 – whose standard iteration is ruled by a *paracontractive* operator [48–50]. Paracontractive operators are a class of operators that generalizes that of contractive operators for two different reasons: 1) they allow for multiple fixed points, rather than implying just a unique fixed point; 2) they enjoy a contractivity property between trajectories and fixed points, rather than between trajectories in general.

Definition 5. *An operator $\mathsf{F}_k : \mathbb{R}^{\mathcal{I}} \to \mathbb{R}^{\mathcal{I}}$ is said to be "paracontractive" if there exists $\gamma \in [0, 1)$ such that for all $k \geq 0$ and for all $x \in \mathbb{R}^{\mathcal{I}}$ it holds*

$$d(\mathsf{F}_{k+1}(x), \hat{\mathcal{X}}_k) \leq \gamma \cdot d(x, \hat{\mathcal{X}}_k), \tag{7}$$

where $\hat{\mathcal{X}}_k$ is the set of fixed points of F_{k+1} for all $k \in \mathbb{N}$.

Proposition 1. *If the operator $\mathsf{F} : \mathbb{R}^n \to \mathbb{R}^n$ is:*

- *α-averaged, i.e., there is $\alpha \in (0, 1)$ such that $d^2(\mathsf{F}_{k+1}(x), \hat{\mathcal{X}}_k) \leq d^2(x, \hat{\mathcal{X}}_k) - \frac{1-\alpha}{\alpha} d^2(x, \mathsf{F}_{k+1}(x))$;*
- *μ-metric subregular, i.e., there is $\mu > 0$ such that $d(x, \hat{\mathcal{X}}_k) \leq \mu d(x, \mathsf{F}_{k+1}(x))$;*

then it is paracontractive with $\gamma := \sqrt{(1 - (1 - \alpha)/(\alpha\mu^2))}$.

Proof: The statement follows from the following steps:

$$d^2(\mathsf{F}_{k+1}(x), \hat{\mathcal{X}}_k) \leq d^2(x, \hat{\mathcal{X}}_k) - \frac{1-\alpha}{\alpha} d^2(x, \mathsf{F}_{k+1}(x)) \leq$$
$$\leq d^2(x, \hat{\mathcal{X}}_k) - \frac{1-\alpha}{\alpha\mu^2} d^2(x, \hat{\mathcal{X}}_k) \leq \left(1 - \frac{1-\alpha}{\alpha\mu^2}\right) d^2(x, \hat{\mathcal{X}}_k).$$

■

The sufficient conditions that we enforce correspond to limits on the variation of the TSI and on the process by which the labels components join and leave during the iteration. In plain words, the normalized shadow distance between two consecutive sets of interests (see Definition 6) and the normalized distance of the arriving components from the set of interest (see Definition 8) must be bounded from above. Moreover, the dimension of the state space cannot decrease too fast (see Definition 7), i.e., there must be an upper bound to the ratio between the number of components at consecutive steps. These limits are formally defined next.

Definition 6 (Bounded TSI). *Consider the TSI $\{\hat{\mathcal{X}}_k \subseteq \mathbb{R}^{\mathcal{I}_k} : k \in \mathbb{N}\}$ of an open operator as in* (3)*. The TSI is said to have "bounded variation" if*

$$\exists B \geq 0 : \quad \frac{d_{\text{SH}}(\hat{\mathcal{X}}_k, \hat{\mathcal{X}}_{k-1})}{\sqrt{|\mathcal{R}_k|}} \leq B, \quad \forall k \in \mathbb{N}.$$

Definition 7 (Bounded departure process). *Consider the open sequence $\{x_k \in \mathbb{R}^{\mathcal{I}_k} : k \in \mathbb{N}\}$ generated by the iteration of an open operator as in* (3)*. The departure process is said to be "bounded" if*

$$\exists \beta \in (0, 1) : \quad \frac{\sqrt{|\mathcal{I}_k|}}{\sqrt{|\mathcal{I}_{k-1}|}} \geq \beta, \quad \forall k \in \mathbb{N}.$$

Definition 8 (Bounded arrival process). *Consider the TSI $\{\hat{\mathcal{X}}_k \subseteq \mathbb{R}^{\mathcal{I}_k} : k \in \mathbb{N}\}$ of an open operator as in* (3) *and let $x_k^{\text{A}} \in \mathbb{R}^{\mathcal{A}_k}$ be the vector stacking the components of all arriving labels. The arrival process is said to be "bounded" if*

$$\exists H \geq 0 : \quad \frac{d(x_k^{\text{A}}, \hat{\mathcal{X}}_k)}{\sqrt{|\mathcal{A}_k|}} \leq H, \quad \forall k \in \mathbb{N}.$$

Example 9. *Consider the iteration described in Example 2 whose TSI is that described in Example 3 and assume that the components change as outlined in Example 4. Then:*

- *the TSI is bounded with $B = 0$ because there are never remaining components;*
- *the departure process is bounded for any $\beta \in [0, 1]$ because the number of components never decreases;*
- *the arrival process is bounded with $H = 1$ because all arriving components initialized at $\{-1, +1\}$ and the corresponding components of the TSI are zeros.*

We now state and prove the main result of this section. A key novelty of this result, compared to with respect to existing works on open multi-agent systems – namely [2, Theorem 3.8] and [1, Theorem 1] – lies in allowing multiple fixed points at each step, forming what we have defined in Definition 1 as the *trajectory of sets of interest* (TSI). In contrast, [1, 2] assume the existence of a *trajectory of points of interest* (TPI), which is a special case of TSI corresponding to a unique fixed point at each step. This extension is far from being trivial as it required introducing a new notion of distance between sets with potentially different cardinalities or compositions, which we have called *shadow distance* in Definition 4, where the name is reminiscent of the projection operation. The need to handle standard iterations with multiple fixed points naturally arises in optimization and learning problems, particularly when the objective function is convex but not strongly convex. This is precisely the case in the present work: in Section III we present a new algorithm, Open ADMM, designed to solve optimization problems with convex (not necessarily strongly convex) functions in the scenario of open networks, and we characterize its stability and performance through the following general results.

Theorem 1. *Consider the iteration of a time-varying open operator $\mathsf{T}_k : \mathbb{R}^{\mathcal{I}_{k-1}} \to \mathbb{R}^{\mathcal{I}_k}$ given component-wise $\forall i \in \mathcal{I}_k$ by*

$$x_k^i = \mathsf{T}_k^i(x_{k-1}) = \begin{cases} \mathsf{F}_k^i(x_{k-1}) & \text{if } i \in \mathcal{R}_k = \mathcal{I}_{k-1} \setminus \mathcal{D}_{k-1}, \\ x_k^{\text{A},i} & \text{if } i \in \mathcal{A}_k = \mathcal{I}_k \setminus \mathcal{I}_{k-1}. \end{cases}$$

and assume that

(a) F_k is paracontractive with $\gamma \in (0, 1)$;
(b) the TSI has bounded variation $B \geq 0$;
(c) the arrival process is bounded with $H \geq 0$.
(d) the departure process is bounded with $\beta \in (\gamma, 1)$.

Then, the open sequence $\{x_k \in \mathbb{R}^{\mathcal{I}_k} : k \in \mathbb{N}\}$ converges linearly with rate $\theta = \gamma/\beta \in (0, 1)$ to the TSI within a radius

$$R = \frac{B + H}{1 - \theta}, \tag{8}$$

according to the following punctual upper bound

$$\frac{d(x_k, \hat{\mathcal{X}}_k)}{\sqrt{|\mathcal{I}_k|}} \leq \theta^k \frac{d(x_0, \hat{\mathcal{X}}_0)}{\sqrt{|\mathcal{I}_0|}} + \frac{1 - \theta^k}{1 - \theta}(B + H).$$

This article has been accepted for publication in IEEE Transactions on Automatic Control. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TAC.2026.3653296

Proof: Let us split the state $x_k = [x_k^{\mathrm{R}}; x_k^{\mathrm{A}}]$ into two vectors such that:

- $x_k^{\mathrm{R}} \in \mathbb{R}^{\mathcal{R}_k}$ is the vector of the remaining components;
- $x_k^{\mathrm{A}} \in \mathbb{R}^{\mathcal{A}_k}$ is the vector of the new components.

For any two consecutive steps $k-1, k$ with $k \in \mathbb{N}$, it holds:

$$\begin{aligned}
d(x_k, \hat{\mathcal{X}}_k) &= \sqrt{d^2(x_k^{\mathrm{R}}, \hat{\mathcal{X}}_k) + d^2(x_k^{\mathrm{A}}, \hat{\mathcal{X}}_k)}, \\
&\leq \sqrt{d^2(x_k^{\mathrm{R}}, \hat{\mathcal{X}}_k)} + \sqrt{d^2(x_k^{\mathrm{A}}, \hat{\mathcal{X}}_k)}, \\
&= d(x_k^{\mathrm{R}}, \hat{\mathcal{X}}_k) + d(x_k^{\mathrm{A}}, \hat{\mathcal{X}}_k), \overset{(i)}{\leq} d(x_k^{\mathrm{R}}, \hat{\mathcal{X}}_k) + \sqrt{|\mathcal{I}_k|}H, \\
&\overset{(ii)}{\leq} d(x_k^{\mathrm{R}}, \hat{\mathcal{X}}_{k-1}) + d_{\mathrm{SH}}(\hat{\mathcal{X}}_k, \hat{\mathcal{X}}_{k-1}) + \sqrt{|\mathcal{I}_k|}H, \\
&\overset{(iii)}{\leq} d(x_k^{\mathrm{R}}, \hat{\mathcal{X}}_{k-1}) + \sqrt{|\mathcal{I}_k|}(B+H), \\
&\overset{(iv)}{\leq} d(\mathsf{F}_k(x_{k-1}), \hat{\mathcal{X}}_{k-1}) + \sqrt{|\mathcal{I}_k|}(B+H), \\
&\overset{(v)}{\leq} \gamma d(x_{k-1}, \hat{\mathcal{X}}_{k-1}) + \sqrt{|\mathcal{I}_k|}(B+H),
\end{aligned}$$

where (i) holds by assumption (c) and $|\mathcal{A}_k| \leq |\mathcal{I}_k|$; (ii) holds by triangle inequality in (6) for the shadow distance in Definition 4; (iii) holds by assumption (b) and $|\mathcal{R}_k| \leq |\mathcal{I}_k|$; (iv) holds because the vector x_k^{R} is entirely contained into $\mathsf{F}_k(x_{k-1})$ by definition; (v) holds by assumption (a). Thus, we have shown that,

$$\frac{d(x_k, \hat{\mathcal{X}}_k)}{\sqrt{|\mathcal{I}_k|}} \leq \gamma \frac{d(x_{k-1}, \hat{\mathcal{X}}_{k-1})}{\sqrt{|\mathcal{I}_k|}} + B + H.$$

which, by assumption (d), becomes

$$\frac{d(x_k, \hat{\mathcal{X}}_k)}{\sqrt{|\mathcal{I}_k|}} \leq \frac{\gamma}{\beta} \frac{d(x_{k-1}, \hat{\mathcal{X}}_{k-1})}{\sqrt{|\mathcal{I}_{k-1}|}} + B + H,$$

By iterating over $k \in \mathbb{N}$ one gets

$$\frac{d(x_k, \hat{\mathcal{X}}_k)}{\sqrt{|\mathcal{I}_k|}} \leq \left(\frac{\gamma}{\beta}\right)^k \frac{d(x_0, \hat{\mathcal{X}}_0)}{\sqrt{|\mathcal{I}_0|}} + (B+H) \sum_{i=0}^{k-1} \left(\frac{\gamma}{\beta}\right)^i.$$

Since in the limit $k \to \infty$ the term $(\gamma/\beta)^k$ goes to 0 and the geometric series equals to $(1-\theta^k)/(1-\theta)$ and goes to $1/(1-\gamma/\beta)$, it holds

$$\limsup_{k\to\infty} \frac{d(x_k, \hat{\mathcal{X}}_k)}{\sqrt{|\mathcal{I}_k|}} \leq \frac{B+H}{1-\frac{\gamma}{\beta}} = R,$$

thus concluding the proof. ■

Corollary 1. *Consider the setting of Theorem 1 and the following simplified cases:*

(a) *The iteration is not open;*

(b) *The iteration is time-invariant.*

Then, the results of Theorem 1 become:

- (a) *implies that the convergence rate is equal to the paracontractivity constant* $\theta = \gamma$ *because* $\beta = 1$*, so the radius reduces to* $R = B/(1-\gamma)$ *because* $H = 0$*;*
- $(a) \wedge (b)$ *implies that the radius reduces to zero because* $B = 0$*, i.e., the sequence converges to a fixed point for any initial condition.*

III. Problem of Interest and Open ADMM Algorithm

With the theoretical framework developed in Section II, we have now the tools to design and analyze algorithms for optimization and learning over open networks. In particular, we are interested in algorithms that can solve the following optimization problem

$$\min_{y \in \mathbb{R}^p} \sum_{i \in \mathcal{V}_k} f_k^i(y), \tag{9}$$

where $p \in \mathbb{N}$ denotes the number of variables, $f_k^i : \mathbb{R}^p \mapsto \mathbb{R}$ denotes the local objective function of an agent $i \in \mathcal{V}_k$ in the network at time k, where $\mathcal{V}_k$ represents the time-varying set of agents. Most of the literature addresses this problem under the assumption that the set of agents in the network remains constant over time, i.e., $\mathcal{V} = \mathcal{V}_0 = \mathcal{V}_1, \cdots$, which facilitates the application of various results from operator theory to study the convergence of custom-designed algorithms. Instead, this work considers the problem without this assumption, thus letting the agents be able to leave and join the network arbitrarily, which results in a time-varying number of agents $n_k = |\mathcal{V}_k| \in \mathbb{N}$: such networks are usually called *Open Multi-Agent Systems* (OMAS). We propose a distributed algorithm called *Open ADMM* to solve the problem in (9) in OMAS and we carry out a convergence analysis by means of *open operator theory*.

Remark 1 (Relationship with online optimization). *We remark that* (9) *is an online optimization problem because the optimal solution is time-varying due to two different factors: 1) the local costs are time-varying [51]; 2) the agents participating in the network change over time, thus yielding a change in the set of the local costs. Consequently, even when the local costs are static, the problem solution is still time-varying due to the open nature of the network.*

A. Problem set-up and working assumptions

An OMAS consists of a time-varying set of agents $\mathcal{V}_k$ which may leave and join the network at any time $k \in \mathbb{N}$. We define the sets of remaining, arriving, departing agents as follows

$$\mathcal{V}_k^{\mathrm{R}} = \mathcal{V}_k \cap V_{k-1},\;\; \mathcal{V}_k^{\mathrm{A}} = \mathcal{V}_k \setminus \mathcal{V}_{k-1},\;\; \mathcal{V}_k^{\mathrm{D}} = \mathcal{V}_k \setminus \mathcal{V}_{k+1},$$

By convention, $\mathcal{V}_{-1} = \emptyset$, yielding $\mathcal{V}_0^{\mathrm{R}} = \emptyset$ and $\mathcal{V}_0^{\mathrm{A}} = \mathcal{V}_0$. The agents are linked according to a graph $\mathcal{G}_k = (\mathcal{V}_k, \mathcal{E}_k)$, where $\mathcal{E}_k \subseteq \mathcal{V}_k \times \mathcal{V}_k$ represents the set of agents pairs that are linked by a point-to-point communication channel. The set of agents that can communicate with the i-th agent at time k is denoted by $\mathcal{N}_k^i = \{j \in \mathcal{V}_k : (i,j) \in \mathcal{E}_k\}$ and its cardinality is denoted by $\eta_k^i = |\mathcal{N}_k^i|$; note that graphs are assumed to be without self-loops, i.e., $i \notin \mathcal{N}_k^i$. We also denote by $\xi_k = 2|\mathcal{E}_k| = \eta_k^1 + \cdots + \eta_k^{n_k}$ twice the total number of communication channels. We formalize next our assumptions on the communication graph among the agents.

Assumption 1. *The communication graph* $\mathcal{G}_k = (\mathcal{V}_k, \mathcal{E}_k)$ *of the OMAS satisfies the following at all times* $k \in \mathbb{N}$*:*

- *undirected, i.e.,* $(i,j) \in \mathcal{E}_k$ *if and only if* $(j,i) \in \mathcal{E}_k$*;*
- *connected, i.e., there is a sequence of consecutive pairs* $(i,a), (a,b), \cdots, (y,z), (z,j)$ *in* $\mathcal{E}_k$ *for all* $i, j \in \mathcal{V}_k$.

An OMAS can actually solve (9) under suitable assumptions on the local objective functions and the corresponding local and global solutions. We formalize our set of assumptions in Assumption 2, which makes use of the following notation for the set of solutions to the problem in (9),

$$\mathcal{Y}_k^\star = \left\{ y_k^\star \in \mathbb{R}^p : \sum_{i\in\mathcal{V}_k} f_k^i(y_k^\star) = \min_{y\in\mathbb{R}^p} \sum_{i\in\mathcal{V}_k} f_k^i(y) \right\},$$

and for the minimizers of each local cost function,

$$\mathcal{Y}_k^{i,\star} = \left\{ y_k^{i,\star} \in \mathbb{R}^p : f_k^i(y_k^{i,\star}) = \min_{y\in\mathbb{R}^p} f_k^i(y) \right\}. \tag{10}$$

Assumption 2. *The problem in* (9) *is such that,* $\forall k \in \mathbb{N}$*:*

(i) the local cost functions f_k^i *are proper [46, Definition 1.4], lower semi-continuous [46, Lemma 1.24], and convex [46, Definition 8.1] for all* $i \in \mathcal{V}_k$*;*

(ii) the set of minimizers $\mathcal{Y}_k^{i,\star} \subseteq \mathbb{R}^p$ *for each local cost function* f_k^i *is not empty;*

(iii) the distance between two consecutive global solutions $y_k^\star \in \mathcal{Y}_k^\star$ *and* $y_{k-1}^\star \in \mathcal{Y}_{k-1}^\star$ *is upper bounded by a constant* $\sigma \geq 0$*;*

(iv) the distance between any local solution $y_k^{i,\star} \in \mathcal{Y}_k^{i,\star}$ *and any global solution* $y_k^\star \in \mathcal{Y}_k^\star$ *is upper bounded by* $\omega \geq 0$.

Assumption 2(i) and (ii) are standard in the context of convex distributed optimization. Assumption 2(iii) is standard in online optimization [51], since it requires consecutive problems to be similar, thus ensuring that warm-starting the algorithm at time k with the output at time $k-1$ leads to good performance. Assumption 2(iv), on the other hand, bounds the heterogeneity of the local cost functions at time k. This implies that all agents have similar data, thus motivating them to cooperate rather than minimizing the local cost functions independently. See section III-C.3 for further discussions.

B. Open ADMM and convergence analysis

To solve the problem in eq. (9) in a distributed way over an open network, we propose the open version of ADMM, which we call *Open ADMM* and whose implementation is detailed in Algorithm 1. Open ADMM requires each agent $i \in \mathcal{V}_k$ to initialize and update per-edge variables $x_k^{ij} \in \mathbb{R}^p$ with $j \in \mathcal{N}_k^i$, while making an estimate y_l^i of the global solution via the proximal step as in eq. (12), tuned with a proximal scaling factor $\rho > 0$, applied to its local objective and evaluated at the average of its local per-edge variables. Each per-edge variable is symmetrically updated as in eq. (11) as a convex combination, with coefficient $\alpha \in (0,1)$, of its previous value and the neighbor's message, which implements the Peaceman–Rachford splitting agreement step.

By defining the set of remaining neighbors $\mathcal{R}_k^i = \mathcal{N}_k^i \cap \mathcal{N}_{k-1}^i$ and the set of arriving neighbors $\mathcal{A}_k^i = \mathcal{N}_k^i \setminus \mathcal{N}_{k-1}^i$, the open operator $\mathsf{T} : \mathbb{R}^{\mathcal{I}_k} \to \mathbb{R}^{\mathcal{I}_k}$ with $\mathcal{I}_k = \{(i,j,\ell) : (i,j) \in \mathcal{E}_k \text{ and } \ell = 1,\dots,p\}$ describing Open ADMM can be formalized block-wise as follows,

$$x_k^{ij} = \mathsf{T}_k^{ij}(x_{k-1}) = \begin{cases} \mathsf{F}_k^{ij}(x_{k-1}) & \text{if } i\in\mathcal{V}_k^{\mathrm{R}} \text{ and } j\in\mathcal{R}_k^i, \\ \rho y_k^{i,\star} & \text{if } j\in\mathcal{A}_k^i \text{ or } i\in\mathcal{A}_k^j \end{cases}$$

where F_k^{ij} is the operator as in eqs. (11)-(12).

Algorithm 1 Open and distributed ADMM

Input: The relaxation $\alpha \in (0,1)$ and the penalty $\rho > 0$

Output: Each agent returns y_k^i which is an approximate solution to the optimization problem in (9)

for $k = 0,1,2,\dots$ **each agent** $i \in \mathcal{V}_k$**:**

  if $i \in \mathcal{V}_k^{\mathrm{A}}$ **is an arriving agent:**

    initializes the state variables to a local optimum

$$x_k^{ij} = \rho y_k^{i,\star} \text{ (see (10))}, \quad \forall j \in \mathcal{N}_k^i$$

  else if $i \in \mathcal{V}_k^{\mathrm{R}}$ **is a remaining agent:**

    receives y_{k-1}^j, x_{k-1}^{ji} from each neighbor $j \in \mathcal{R}_k^i$

    updates the remaining state variable according to

$$x_k^{ij} = (1-\alpha)x_{k-1}^{ij} - \alpha x_{k-1}^{ji} + 2\rho\alpha y_{k-1}^j, \quad \forall j\in\mathcal{R}_k^i \tag{11}$$

    initializes the new state variables to a local optimum

$$x_k^{ij} = \rho y_k^{i,\star} \text{ (see (10))}, \quad \forall j \in \mathcal{A}_k^i$$

  end if

  updates the output variable

$$y_k^i = \mathrm{prox}_{f_k^i}^{1/\rho\eta_k^i} \left(\frac{1}{\rho\eta_k^i} \sum_{j\in\mathcal{N}_k^i} x_k^{ij} \right) \tag{12}$$

  transmits y_k^i, x_k^{ij} to each neighbor $j \in \mathcal{N}_k^i$

end for

We construct the global network state $x_k = [\cdots, x^{ij}, \cdots] \in \mathbb{R}^{\mathcal{I}_k}$ by concatenating all individual agent states associated with edges in the network. Specifically, we stack all vectors $x_k^{ij} \in \mathbb{R}^p$ corresponding to edges $(i,j) \in \mathcal{E}_k$ according to lexicographic ordering, i.e., first by i and then by j. Thus, the standard iteration of Open ADMM ruled by the operator $\mathsf{F}_k = [\cdots, \mathsf{F}_k^{ij}, \cdots]^\top$ becomes that of DOT-ADMM presented by the same authors in [47], given by

$$x_k = \mathsf{F}_k(x_{k-1}) = [(1-\alpha)I - \alpha P_{k-1}]x_{k-1} + 2\alpha\rho P_{k-1}A_{k-1}y_{k-1}$$
$$y_k = \mathsf{P}_k(x_k) = \mathrm{prox}_{f_k}^{1/\rho\eta_k}(D_k A_k^\top x_k)$$

where: the operator $\mathrm{prox}_{f_k}^{1/\rho\eta_k} : \mathbb{R}^{\mathcal{J}_k} \to \mathbb{R}^{\mathcal{J}_k}$ with $\mathcal{J}_k = \{(i,\ell) : i \in \mathcal{V}_k \text{ and } \ell = 1,\dots,p\}$ applies block-wise the proximal of the local costs f_k^i; the matrix $A_k = \Lambda \otimes I_p \in \{0,1\}^{\mathcal{I}_k \times \mathcal{J}_k}$ is given by $\Lambda \in \{0,1\}^{\mathcal{E}_k \times \mathcal{V}_k}$ defined block-wise for $i \in \mathcal{V}_k$ and $j \in \mathcal{N}_k^i$ by the matrices $\Lambda^{ij} = [e_i, e_j]^\top$ for $j > i$ where $e_\ell \in \{0,1\}^{\mathcal{V}_k}$ denotes a canonical vector; the matrix $D_k \in \mathbb{R}^{\mathcal{J}_k \times \mathcal{J}_k}$ is given by $D_k = \mathrm{blkdiag}\{(\rho\eta_k^i)^{-1} I_p\}_{i=1}^n$; the matrix $P_k \in \{0,1\}^{\mathcal{I}_k \times \mathcal{I}_k}$ is a squared block-diagonal matrix given by $P_k = I_{\xi_k/2} \otimes (\Pi \otimes I_p)$ with $\Pi = [0\ 1; 1\ 0]$.

To prove the convergence of Open ADMM, we resort to our main convergence result in Theorem 1 for open operators. We first show that Assumptions 1-2 are sufficient to guarantee the existence of a TSI and its boundedness (Lemma 1) as well as the boundedness of the arrival process (Lemma 2). Theorem 2 builds on these lemmas to establish that Theorem 1 holds for Open ADMM; specifically, it proves that Open ADMM is open stable, meaning that the open sequences it generates converge linearly to the trajectory of sets of interest within a certain stability radius, as formalized in Definition 2.

Lemma 1. *Consider an OMAS executing Open ADMM to distributedly solve the optimization problem* (9) *under Assumptions 1-2. Then, there is a TSI* $\{\hat{\mathcal{X}}_k : k \in \mathbb{N}\}$ *given by*

$$\hat{\mathcal{X}}_k = \{\hat{x}_k \,|\, (I+P_k)\hat{x}_k = 2\rho P_k A_k(\mathbb{1}_{n_k} \otimes y_k^\star),\ y_k^\star \in \mathcal{Y}_k^\star\}, \quad (13)$$

where $\mathcal{Y}_k^\star$ *is the solution set, while the TSI is bounded with*

$$B = \rho\sigma. \quad (14)$$

Proof: See Section A in the Appendix. ■

Lemma 2. *Consider an OMAS executing Open ADMM to distributedly solve the optimization problem* (9) *under Assumptions 1-2. Then, the arrival process is bounded with*

$$H = \rho\omega. \quad (15)$$

Proof: See Section B in the Appendix. ■

Theorem 2. *Consider an OMAS executing Open ADMM to distributedly solve an optimization problem as in* (9) *under Assumptions 1-2. If the standard iteration is paracontractive with* $\gamma \in (0,1)$ *and the departure process is bounded with* $\beta \in (\gamma, 1)$*, then the open sequence* $\{x_k : k \in \mathbb{N}\}$ *generated by the open operator of Open ADMM converges with linear rate* $\theta = \gamma/\beta \in (0,1)$ *within a radius* R*,*

$$\limsup_{k\to\infty} \frac{d(x_k, \hat{\mathcal{X}}_k)}{\sqrt{p\xi_k}} \le \rho\frac{(\sigma+\omega)}{(1-\theta)} =: R.$$

Proof: The proof consists in showing that conditions $(a)-(d)$ of Theorem 1 hold: Paracontractivity of the operator ruling the standard dynamics holds by assumption with $\gamma \in (0,1)$; Boundedness of the variation of the TSI holds by Lemma 1 with $B = \rho\sigma/\sqrt{p}$; Boundedness of the arrival process holds by Lemma 2 with $H = \rho\sigma/\sqrt{p}$; Boundedness of the departure process holds by assumption with $\beta \in (\gamma, 1)$. The thesis follows by substituting H and B into (8). ■

Remark 2. *Proposition 1 offers a practical way to verify the paracontractivity of the operator* F *ruling the standard iteration. If* F *admits affine lower and upper bounds then it is metric subregular [47, Proposition 1], which, together with averagedness*[1] *implies paracontractivity by Proposition 1. Remarkably, this is the case for the logistic loss function considered in the numerical simulations [47, Proposition 4].*

We now characterize the performance of Open ADMM in terms of the normalized distance between the open sequence of the stacking of all agents' states,

$$\{\widetilde{y}_k \;:\; k \in \mathbb{N}\} \quad \text{with} \quad \begin{cases} \widetilde{y}_k = [\widetilde{y}_k^1, \cdots, \widetilde{y}_k^{n_k}]^\top, \\ \widetilde{y}_k^i = \mathbb{1}_{\eta_k^i} \otimes y_k^i, \end{cases} \quad (16)$$

from the consensus state on the optimal solutions,

$$\{\widetilde{\mathcal{C}}_k^\star \;:\; k \in \mathbb{N}\} \quad \text{with} \quad \widetilde{\mathcal{C}}_k^\star := \{\mathbb{1}_{\xi_k} \otimes y_k^\star \;\mid\; y_k^\star \in \mathcal{Y}_k^\star\}. \quad (17)$$

Theorem 3. *In the scenario of Theorem 2, the open sequence of agents' estimates* (16) *converges linearly to the consensus state on the optimal solutions* (17) *within a radius* Δ*,*

$$\limsup_{k\to\infty} \frac{d(\widetilde{y}_k, \widetilde{\mathcal{C}}_k^\star)}{\sqrt{p\xi_k}} = \frac{(\sigma+\omega)}{(1-\theta)} =: \Delta.$$

Proof: See Section C in the Appendix. ■

[1] Because the standard iteration results from the relaxed Peaceman-Rachford Splitting to the dual of the distributed version of the problem in (9) [52, 53]

C. Discussion on the proposed algorithm and the set-up

Let us discuss the set-up and Algorithm 1 from the perspective of learning. Each agent has access to data sampled from a local distribution, which are used to define the local cost as

$$f_k^i(y) = \tfrac{1}{m_{i,k}} \textstyle\sum_{h=1}^{m_{i,k}} \ell(y \mid d_{ih,k})$$

where $d_{ih,k}$ are the local data, $m_{i,k}$ is the number of data, and ℓ is a loss function. In principle, each agent could compute its model on the local data only, i.e., $y_k^{i,\star} \in \mathcal{Y}_k^{i,\star}$, which always exists according to Assumptions 2(i)-(ii). However, this model in general can have *poor accuracy and generalization*. The poor accuracy is due to the limited amount of data that an agent can collect and store, while the poor generalization is due to the skewed/biased perspective that the local distribution has of the phenomenon being analyzed. Thus, each agent has an incentive to participate in the cooperative learning process to train a model that is more accurate (all data of all agents are involved) and more general (all distributions together offer a better perspective on the phenomenon).

1) Initialization procedure: This interpretation then motivates the fact that Open ADMM requires that agents' state variables x_k^{ij} are initialized to their local minimizer $y_k^{i,\star}$. Essentially, Open ADMM assumes that any arriving agent has already pre-trained its local model as well as possible using its available data, and then joins the network to collaboratively refine this model into a more accurate and general one. During execution, each agent tracks its local minimizer to initialize new state variables as the network topology evolves, which ensures smooth adaptation without the cost of re-training the local model every time. In practice, the local optimum may not be computable exactly and can only be approximated within a bounded error. This approximation does not compromise the functioning of Open ADMM; however, it affects the stability radius by effectively increasing the bound on the arrival process in Lemma 2.

Alternative initialization strategies are also possible, such as the "blind" initialization to a zero vector, i.e., $y_k^i = \mathbb{0}_p$, or initialization based on the average of neighboring estimates, i.e., $x_k^{ij} = \sum_{j\in\mathcal{N}_k^i} y_{k-1}^j$. Both approaches are computationally cheaper for the agents, since they avoid the overhead of pre-training and tracking updates of local models. However, each comes with limitations. The blind initialization typically yields poorer performance, as it ignores both local information and the current global estimate. In contrast, the neighbor-averaging approach exhibits good performance as it enables a form of knowledge transfer between agents [13], but it introduces additional communication and coordination among agents when new agents arrives or new links are established. A rigorous convergence analysis of Open ADMM under these different initialization schemes is left for future work, while Section IV-B.4 provides comparative numerical results for the two methods discussed above.

2) Arrival/departure process: Agents benefit from joining cooperative learning to refine their local models, but internal decision rules or resource constraints may also lead them to leave, establishing a feedback loop. For instance, agents could decide to leave the cooperative learning once the trained model exceeds a high enough test accuracy on its local distribution; in the case of networks of battery-powered devices, agents may choose to, or be forced to, stop participating to preserve battery, and join again only once its charge is restored; in multi-robot exploration tasks where each robot builds a local map and shares updates only with neighbors within its radio range, some robots may move so far away that they cannot communicate with any others, leaving them excluded from the shared model until they return close enough to rejoin the team. In these examples, the interplay between an agent's state, its actions, and its network membership closes the feedback loop.

3) Working assumptions: Since the arrival and departure of agents might disrupt the cooperative learning, resulting in trained models with poor accuracy. Some assumptions on the time-variability of the optimization problem are needed. First, Assumption 2(iii) bounds the variation of the global solution over time. This ensures that changes in the set of participating agents and their local datasets do not drastically alter the global model. Second, we need to guarantee that suitable bounds on the departure and arrival process are satisfied. Assumption 2(iv) ensures that local models (which are the initializations of edge variables of arriving agents and links) are not too far from the global model. This condition effectively bounds the distance between local and global distributions: heterogeneity of local data is still allowed, but within a controlled level that prevents excessive divergence in local models. Finally, Theorem 2 requires a bounded rate of departure to mitigate the drifting effect of the global solution toward the local ones due to agent departures. Together, these assumptions delineate a class of open learning problems for which stability and convergence guarantees can be established.

4) Hyperparameter tuning: Algorithm 1 has two tunable hyperparameters, the relaxation $\alpha \in (0,1)$ and penalty $\rho > 0$. Importantly, Theorem 2 holds for any choice of these parameters from the allowed ranges. How to (distributedly) tune these parameters for optimal performance is still an open problem, especially in open networks.

IV. Applications and Numerical Simulations

In this section, we demonstrate Open ADMM performance by showcasing its application to dynamic consensus problems and supervised learning problems in OMAS.

A. Dynamic consensus algorithms for open networks

Consider an OMAS in which the i-th agent with $i \in \mathcal{V}_k$ has access to a scalar, time-varying reference signal $u_k^i \in \mathbb{R}$. The dynamic consensus problems on the average, the maximum, and the median values of these signals can be recast as a time-varying optimization problem in (9) with local scalar cost functions with $p = 1$, satisfying Assumption 2(i)-(ii), given by

$$f_k^i(y) := \frac{1}{q}\left|y - u_k^i\right|^q + \iota_{\mathcal{S}_k^i}(y) \tag{18}$$

Proposition 2. *[57, Proposition 1] Consider an OMAS distributedly solving the optimization problem* (9) *with local costs* (18) *under Assumption 1. Then, there is a unique solution $y_k^* \in \mathcal{Y}_k^\star$ to the problem such that:*

i) If $q = 2$ and $\mathcal{S}_k^i = \mathbb{R}$, then $y_k^\star = \operatorname{avg}(u_k)$;
ii) If $q = 2$ and $\mathcal{S}_k^i = \{x \geq u_i(k)\}$, then $y_k^\star = \max(u_k)$;
iii) If $q = 1$ and $\mathcal{S}_k^i = \mathbb{R}$, then $y_k^\star = \operatorname{med}(u_k)$.

The updates of the Open ADMM for the tracking (or alternatively, dynamic consensus) problems of the average, maximum, and median values can be written in closed-form. In particular, the initialization of the new state variables becomes $x_k^{ij} = \rho u_k^i$ because the local optimal solution is unique and equal to $y_k^{i,\star} = u_k^i$; the updates of the state variables in (11) remain unchanged; the updates of the output variables in (12) becomes (cfr. [57, Lemmas 1-2-3]):

$$\text{(average)} \quad y_k^i = \frac{u_k^i + \sum_{j \in \mathcal{N}_k^i} x_k^{ij}}{1 + \rho\eta_k^i},$$

$$\text{(maximum)} \quad y_k^i = \max\left\{u_k^i, \frac{u_k^i + \sum_{j \in \mathcal{N}_k^i} x_k^{ij}}{1 + \rho\eta_k^i}\right\},$$

$$\text{(median)} \quad y_k^i = u_k^i + \max\{\theta_k^{i,-} - u_k^i, 0\} + \min\{\theta_k^{i,+} - u_k^i, 0\},$$

$$\text{with } \theta_k^{i,\pm} = \frac{1}{\rho\eta_k^i}\Big[\sum_{j \in \mathcal{N}_k^i} x_{k-1}^{ij} \pm 1\Big].$$

It can also be verified that the general conditions required by Assumption 2(iii)-(iv) hold if the reference signals $u_k^i \in \mathbb{R}$ with $i \in \mathcal{V}_k$ are such that [57, 58]: their absolute variation is bounded by a constant $\sigma \geq 0$, i.e., $\left|u_k^i - u_{k-1}^i\right| \leq \sigma$; they lie within a set of size $\omega \geq 0$, $|\overline{u}_k - \underline{u}_k| \leq \omega$.

Remark 3 (Comparison with the state-of-the-art). *A comparison of the working assumptions of the proposed protocols derived from Open ADMM and their performance with the state-of-the-art is detailed in Table II. The only algorithms accounting for directed communications are those provided in [54–56], but their tracking error is not formally characterized. This is the most common case as the network is usually assumed to be eventually closed and the algorithm is characterized only at steady state. In contrast, the proposed Open ADMM and the algorithms proposed in [1], [2] work under the stronger assumption of undirected communications, but enjoy an eventually bounded tracking error. Also, the proposed Open ADMM is the only algorithm guaranteeing a null steady state error for closed, fixed networks and constant reference signals (as a consequence of Corollary 1), while [1] ensures it can be made arbitrarily small.*

Figure 5 shows a typical realization of a network of agents executing the proposed Open ADMM algorithm and compare it with the OPDC algorithm [2], the OpenRC algorithm [54], and Algorithm 1 in [55] in the scenario described next. We used the tuning parameters $\rho = 1$ and $\alpha = 0.99$ for Open ADMM, and $\alpha = \varepsilon = 0.01$ for OPDC. The network starts with 200 agents whose state is initialized uniformly at random in the interval $[0, 500]$. The initial graph is randomly generated with edge probability $p = 0.05$. At any subsequent step $k \geq 0$, there is a probability $p_k^{\text{join}} \in [0,1]$ that one node joins the network and there is a probability $p_k^{\text{leave}} \in [0,1]$ that one node leaves the network. The probabilities are chosen time-varying so that the network first increases in size by approximately 50% and then decreases by approximately 20%.

TABLE II
COMPARISON WITH THE STATE OF THE ART FOR DISTRIBUTED TRACKING IN OPEN MULTI-AGENT SYSTEMS WITH DISCRETE-TIME DYNAMICS.

[Ref.]	Problem	Assumptions on the network	Assumptions on the signals	Bounded tracking error	Null steady-state error
[1] Deplano *et al.* (2024)	Max/Min	Undirected + Connected + Bounded diameter + Dwell time	Bounded variation + Bounded span	✓	≈ (arbitrarily small)
[33] Abdelrahim *et al.* (2017)	Max/Min	Complete + Eventually closed	Constant	✗	✓
[54] Makridis *et al.* (2024)	Avg	Directed + Strongly connected + Unbalanced + Eventually closed	Constant	✗	✓
[55] Hadjicostis *et al.* (2024)	Avg	Directed + Strongly connected + Eventually closed	Constant	✗	✓
[34] Oliva *et al.* (2023)	Avg	Undirected + Connected + Eventually closed	Constant	✗	✓
[2] Franceschelli *et al.* (2020)	Avg	Undirected + Connected + Bounded departures	Bounded variation + Bounded span	✓	✗
[56] Zhu *et al.* (2010)	Avg	Directed + Balanced + jointly strongly connected	Relative bounded variation	✗	✗
[31] Dashti *et al.* (2022)	Avg/Mode	Undirected + Connected + Eventually closed	Constant	✗	✓
[THIS WORK]	Avg, Max/Min, Median, $\cdots$	Undirected + Connected + Bounded departures	Bounded variation + Bounded span	✓	✓

We model these events in a way that the network remains connected whichever event occurs. Consequently, the set of network agents is frequently renewed and the number of agents changes according to Fig. 5(bottom). The arriving agent creates random communication channels with all other agents with probability $p = 0.1$. Input reference signals are randomly sampled in the interval $[\underline{u}_k, \underline{u}_k + \omega]$ with $\omega = 5$ when agents join the network and evolve with bounded variation $\sigma = 0.02$. Figure 5(top) shows the normalized distance of the network state from the consensus on the average, namely

$$e_k := \frac{d(\widetilde{y}_k, \widetilde{\mathcal{C}}_k^\star)}{\sqrt{p\xi_k}}, \quad \text{with} \quad \widetilde{y}_k, \widetilde{\mathcal{C}}_k^\star \quad \text{as in (16)-(17).} \tag{19}$$

One can verify that the OpenRC algorithm in [54] and Algorithm 1 in [55] initially converge close to the optimal solution, but then lose the tracking because they do not account for time-varying reference signals. Even though they behave very similarly, as they are both based on a ratio-consensus mechanism, we observe larger transient errors with OpenRC, possibly due to its additional acknowledgement-based and mass-redistribution mechanisms tailored for unbalanced directed graphs.

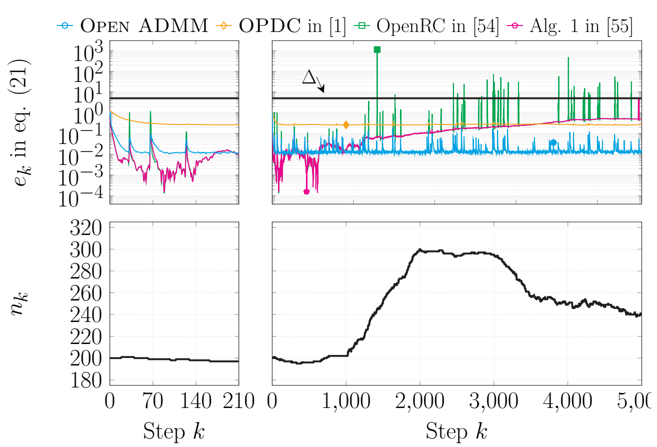

Fig. 5. Comparison between Open ADMM, OPDC in [2], OpenRC in [54], and Alg. 1 in [55] in an open network, where $\boldsymbol{\Delta}$ is the stability radius as in Theorem 3.

Instead, both Open ADMM and the OPDC algorithm converge linearly despite the continuous network changes and approach a bounded tracking error. In this simulation, Open ADMM outperforms OPDC both for the faster convergence rate and the smaller tracking error, which is bounded coherently with Theorem 3. In particular, we considered the best possible convergence radius $\Delta = (\sigma + \omega) = 5.02$. It is evident that this bound on the tracking error is a quite conservative estimate of the actual error.

B. Supervised learning over open networks

In this subsection we apply the Open ADMM to solve *supervised* learning problems. Each local dataset contains labeled pairs (a_{ih}, b_{ih}) where a_{ih} are the features and b_{ih} are the associated labels. The goal is to learn a model y that predicts the label b_{ih} from the sample a_{ih}. In particular, we consider a classification problem characterized by the loss

$$f_k^i(y) = \frac{1}{m_i} \sum_{h=1}^{m_i} \log\left(1 + \exp\left(-b_{ih} a_{ih}^\top y\right)\right) + \frac{\epsilon}{2}\|y\|^2$$

where $a_{ih} \in \mathbb{R}^p$ and $b_{ih} \in \{-1, 1\}$ are the pairs of feature vector and label. Datasets of dimension $m_i = 150$ are randomly generated with `scikit-learn` given $p = 5$ and $\epsilon = 0.05$.

This article has been accepted for publication in IEEE Transactions on Automatic Control. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TAC.2026.3653296

We note that the local costs do not have a closed form prox for the selected logistic loss; rather, the agents approximate it with accelerated gradient descent, up to precision of 10^{-10}. Since this makes it impractical to numerically evaluate the distance of the agents' estimates from the global optimal solution, we instead use the following metric as a proxy for the algorithm's convergence to the solution of (9):

$$\varepsilon_k := \left\| \frac{1}{n_k} \sum_{i \in \mathcal{V}_k} \nabla f_i(\overline{y}_k) \right\|^2, \quad \text{with} \quad \overline{y}_k = \frac{1}{n_k} \sum_{i \in \mathcal{V}_k} y_k^i \qquad (20)$$

We consider arrival and departure events that occur according to the Poisson distributions $\text{Pois}(\lambda^{\text{join}})$ and $\text{Pois}(\lambda^{\text{leave}})$, respectively, resulting in λ^{join} and λ^{leave} arrivals and departures in mean. Additionally, arriving agents are connected to a number of remaining agents equal to the average degree in the network, and the network starts with $n_0 = 50$ agents.

1) Comparison with the state-of-the-art: Our first simulation considers a scenario in which the network is characterized by four modes for the arrival/departure processes: 1) same arrivals and departures $\lambda^{\text{join}} = \lambda^{\text{leave}} = 1$; 2) more arrivals $\lambda^{\text{join}} = 2\lambda^{\text{leave}} = 1$; 3) more departures $\lambda^{\text{leave}} = 2\lambda^{\text{join}} = 1$, 4) fixed network $\lambda^{\text{leave}} = \lambda^{\text{join}} = 0$. In Figure 6 we compare Open ADMM with the dual-averaging approach proposed in [13] and the distributed subgradient method proposed in [14], and report the evolution of ε_k and the number of agents n_k over the course of a simulation for the three algorithms. The step-size for the algorithms in [13, 14] has been hand-tuned to a fixed constant in the order of $\mathcal{O}(1/\sqrt{T})$ to achieve the best performance while meeting their theoretical requirements. After the initial transient, ε_k remains upper bounded for all three algorithms, with Open ADMM exhibiting the best performance with an average tracking error of $\approx 5 \cdot 10^{-3}$ – within $k \in [100, 1000]$, thus excluding the initial transient and the final steady state – followed by [14] with $\approx 2 \cdot 10^{-1}$ and by [13] with $\approx 2 \cdot 10^{0}$. Moreover, when the network becomes fixed, Open ADMM is the only algorithm that converges asymptotically to the optimal solution with no error.

2) The impact of arrival and departure rates: The values of λ^{join} and λ^{leave} of course affect that magnitude of the fluctuations in n_k that happen over the course of the simulation. To explore how this in turn impacts ε_k, we run a set of simulations with $\lambda^{\text{join}} = \lambda^{\text{leave}} = \lambda$ and different choices of $\lambda \in \{0.1, 1, 10, 100\}$. The results are reported in Figure 7. We notice that the larger the mean of the arrival/departure events, the larger ε_k is, due to the wider fluctuations in the number of agents. This is further verified by the results of Table III, which report the minimum, maximum, mean and standard deviation of ε_k in the second half of the simulation (to exclude transient behaviors due to the initialization). The results are averaged over 10 Monte Carlo iterations. The results confirm that the more arrival/departure events, the larger ε_k, in line with our theoretical results. We remark that the algorithm does not diverge even in the challenging scenario $\lambda = 100$.

3) The special scenario of only replacements: The open network models used in the previous sections allowed for the number of agents n_k to vary over time. We now consider the special case in which the network has a fixed number of agents n [3, 12, 23], but allow a number of them – drawn from $\text{Pois}(\lambda)$, $\lambda = 1$ – to be replaced throughout the simulation. Table IV reports the minimum, maximum, mean and standard deviation of ε_k in the second half of the simulation, with $n \in \{50, 100, 500\}$. The results are averaged over 10 Monte Carlo iterations. We notice that the higher the number of agents, the smaller the value of ε_k. Indeed, in larger networks the impact of replacing a few nodes (1 on average) is lesser,

TABLE III
PERFORMANCE OF OPEN ADMM IN TERMS OF ε_k IN (20) FOR DIFFERENT ARRIVAL/DEPARTURE RATES.

λ	Min	Mean ± Std	Max
0.1	6.163×10^{-5}	$(6.765 \pm 7.988) \times 10^{-4}$	4.192×10^{-3}
1	3.178×10^{-3}	$(1.701 \pm 1.012) \times 10^{-2}$	5.655×10^{-2}
10	6.550×10^{-2}	$(1.765 \pm 0.968) \times 10^{-1}$	5.698×10^{-1}
100	9.282×10^{-1}	1.692 ± 0.624	3.046

TABLE IV
PERFORMANCE OF OPEN ADMM IN TERMS OF ε_k IN (20) FOR DIFFERENT NETWORK SIZES, AND ONLY REPLACEMENT EVENTS.

n	Min	Mean ± Std	Max
50	1.661×10^{-3}	$(1.253 \pm 0.980) \times 10^{-2}$	6.159×10^{-2}
100	3.997×10^{-4}	$(3.685 \pm 4.535) \times 10^{-3}$	3.382×10^{-2}
500	7.493×10^{-5}	$(4.676 \pm 3.045) \times 10^{-4}$	1.817×10^{-3}

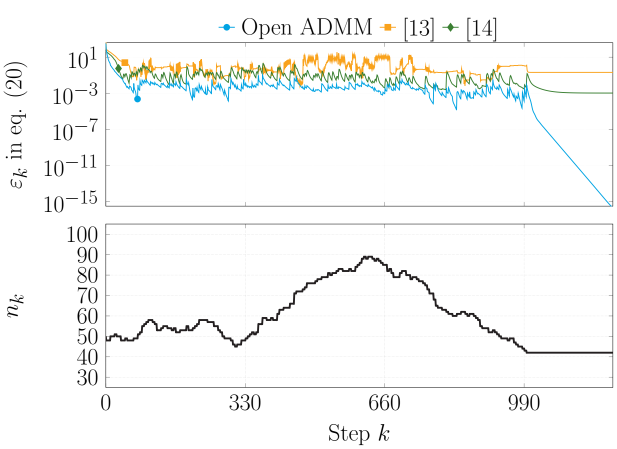

Fig. 6. Comparison of Open ADMM with the state-of-the-art for solving a supervised learning problem in an open network with four modes: $\lambda^{\text{join}}=\lambda^{\text{leave}}=1, \lambda^{\text{join}}=2\lambda^{\text{leave}}=1, \lambda^{\text{join}}=0.5\lambda^{\text{leave}}=1, \lambda^{\text{join}}=\lambda^{\text{leave}}=0$.

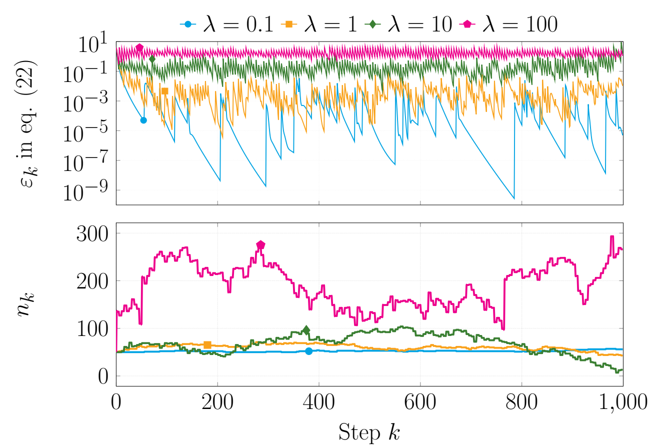

Fig. 7. Comparison of Open ADMM performance while solving a supervised learning problem in an open network with $\lambda^{\text{join}} = \lambda^{\text{leave}} = \lambda \in \{0.1, 1, 10, 100\}$.

as the large number of cost functions in (9) ensure smaller sensitivity to individual changes in the agents. However, ε_k does not converge to zero, as changes in the network still act as persistent perturbations.

4) Alternative initialization strategies: The agents executing Open ADMM use their local optimum as an initialization when they join the network (or when they connect to a new neighbor). However, in practice the agents may not have access to a locally optimal model (especially if their cost function changes over time). Thus, in this section we test the performance of Open ADMM with different local initializations: (a) local optimum $x_k^{ij} = \rho y^{i,\star}$; (b) zero vector $x_k^{ij} = 0$; (c) the average of neighbors' states $x_k^{ij} = \sum_{j \in \mathcal{N}_k^i} y_{k-1}^j$.

Table V reports the minimum, maximum, mean and standard deviation of ε_k (in the second half of the simulation) with the different initializations. The results are averaged over 10 Monte Carlo iterations. First of all, we remark that the naïve choice (b) leads to the worst performance overall, due to the fact that it causes the largest transient effect. Indeed, the zero vector is (on average) much farther from the fixed point than (a) and (c). On the other hand, these two better informed initializations have very close performance, with (c) being slightly worse. As mentioned above, (a) may be inaccessible due to the computational cost of computing the local optimum, making (c) a less expensive alternative that relies on communication rather than local computation.

5) Inexact proximal evaluations: Open ADMM requires the (approximate) computation of local proximals in (12), which in the previous sections was done by accelerated gradient descent up to high precision. However, this requires the agents to compute full gradients, which might be impractical especially if $m_i \gg 1$. Thus in this section we evaluate the performance of Open ADMM when the agents use τ steps of stochastic gradient descent, evaluating gradients on a subset $\mathcal{B}_i \subset \{1, \dots, m_i\}$ of local datapoints. We select a network with $n_0 = 50$ and Poisson arrivals/departures with mean λ, with agents storing $m_i = 25$ datapoints. The results in terms of asymptotic ε_k are reported in Table VI for different τ, λ and $|\mathcal{B}_i|$. There are several interesting phenomena to observe in this figure, which future theoretical research will address.

TABLE V

PERFORMANCE OF OPEN ADMM IN TERMS OF $\boldsymbol{\varepsilon_k}$ IN (20) FOR DIFFERENT INITIALIZATIONS OF ARRIVING AGENTS.

Init.	Min	Mean ± Std	Max
(a)	3.178×10^{-3}	$(1.701 \pm 1.012) \times 10^{-2}$	5.655×10^{-2}
(b)	2.665×10^{-3}	$(2.214 \pm 3.854) \times 10^{-1}$	1.692
(c)	3.212×10^{-3}	$(2.134 \pm 1.630) \times 10^{-2}$	7.773×10^{-2}

TABLE VI

PERFORMANCE OF OPEN ADMM WHEN EMPLOYING STOCHASTIC GRADIENTS DURING PROXIMAL EVALUATIONS.

	$\lambda = 0$			$\lambda = 1$		
$\lvert\mathcal{B}_i\rvert$	$\tau = 1$	$\tau = 10$	$\tau = 100$	$\tau = 1$	$\tau = 10$	$\tau = 100$
1	0.127	0.236	0.368	0.545	0.159	0.193
5	0.040	0.092	0.149	0.502	0.096	0.045
10	0.017	0.020	0.051	0.454	0.077	0.012
15	**0.007**	**0.004**	0.015	0.411	**0.043**	**0.006**
20	0.009	0.006	0.003	0.356	0.050	**0.006**
m_i	0.017	0.021	$\mathbf{7 \times 10^{-6}}$	**0.344**	0.059	**0.006**

First, when τ is small, selecting a larger batch might not be the best choice (see *e.g.* columns 1 and 2 with $\lambda = 0$). This is due to the fact that stochastic gradients might reach a small neighborhood of the exact proximal faster than full gradients allow. When τ is large, a larger batch is beneficial, since the stochastic gradient noise reduces and the proximal is more accurately computed. Secondly, we observe that with $\lambda = 1$, convergence is of course not exact, since both the network openness and the use of stochastic gradients contribute to it. Interestingly, when $\tau = 100$, the openness is the dominant factor, since $|\mathcal{B}_i| = 15, 20, m_i$ all achieve the same error.

V. CONCLUSIONS

This article presents the Open ADMM algorithm to solve distributed optimization and learning problems in networks where agents may join or leave during the execution of the algorithm. The stability and performance of Open ADMM are discussed in the light of the newly introduced *open operator theory*, which are corroborated through extensive simulations on dynamic consensus problems with different metrics, as well as on classification tasks using logistic regression. The superiority of our approach with respect to the state-of-the-art has been discussed both in terms of working assumptions and performance, as detailed in Tables I-II.

Many interesting future research directions originate from this manuscript, mainly involving the development of new technical tools within the framework of *open operator theory* to describe and analyze more complex and realistic scenarios, such as asynchronous communications among agents, unreliable or limited links accounting for packet losses and quantization effects, inexact local computations, and so on.

REFERENCES

[1] D. Deplano, M. Franceschelli, and A. Giua, "Stability of paracontractive open multi-agent systems," in *63rd Conference on Decision and Control*, IEEE, 2024, pp. 3031–3036.

[2] M. Franceschelli and P. Frasca, "Stability of open multiagent systems and applications to dynamic consensus," *IEEE Transactions on Automatic Control*, vol. 66, no. 5, pp. 2326–2331, 2020.

[3] R. Vizuete, C. Monnoyer de Galland, P. Frasca, E. Panteley, and J. M. Hendrickx, "Trends and questions in open multi-agent systems," in *Hybrid and Networked Dynamical Systems: Modeling, Analysis and Control*, Springer, 2024, pp. 219–252.

[4] I. Pinyol and J. Sabater-Mir, "Computational trust and reputation models for open multi-agent systems: A review," *Artificial Intelligence Review*, vol. 40, no. 1, pp. 1–25, 2013.

[5] Z. Li, G. Xie, K. Hwang, and Z. Li, "Churn-resilient protocol for massive data dissemination in p2p networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 8, pp. 1342–1349, 2011.

[6] F. Wuhib, R. Yanggratoke, and R. Stadler, "Allocating compute and network resources under management objectives in large-scale clouds," *Journal of Network and Systems Management*, vol. 23, pp. 111–136, 2015.

[7] T. Lykouris, V. Syrgkanis, and É. Tardos, "Learning and efficiency in games with dynamic population," in *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*, SIAM, 2016, pp. 120–129.

[8] F. A. Taha, K. Rokade, and F. Parise, "Learning in time-varying monotone network games with dynamic populations," *arXiv preprint arXiv:2408.06253*, 2024.

[9] M. Warren, *Bitcoin: A Game-Theoretic Analysis*. 2023, pp. 1–328.

[10] H. Rainer and U. Krause, "Opinion dynamics and bounded confidence: Models, analysis and simulation," 2002.

[11] M. Zhu and G. Xie, "Model and simulation of krause model in dynamic open network," in *AIP Conference Proceedings*, AIP Publishing, vol. 1864, 2017.

[12] J. M. Hendrickx and M. G. Rabbat, "Stability of decentralized gradient descent in open multi-agent systems," in *59th Conference on Decision and Control*, Jeju Island, Korea (South): IEEE, 2020, pp. 4885–4890.

[13] Y.-G. Hsieh, F. Iutzeler, J. Malick, and P. Mertikopoulos, "Optimization in Open Networks via Dual Averaging," in *2021 60th IEEE Conference on Decision and Control (CDC)*, Austin, TX, USA: IEEE, 2021, pp. 514–520.

[14] N. Hayashi, "Distributed Subgradient Method in Open Multiagent Systems," *IEEE Transactions on Automatic Control*, vol. 68, no. 10, pp. 6192–6199, 2023.

[15] R. Patel, P. Frasca, J. W. Durham, R. Carli, and F. Bullo, "Dynamic partitioning and coverage control with asynchronous one-to-base-station communication," *IEEE transactions on control of network systems*, vol. 3, no. 1, pp. 24–33, 2015.

[16] E. Restrepo, A. Lorìa, I. Sarras, and J. Marzat, "Consensus of open multi-agent systems over dynamic undirected graphs with preserved connectivity and collision avoidance," in *61st Conference on Decision and Control*, IEEE, 2022, pp. 4609–4614.

[17] E. Restrepo and P. R. Giordano, "A distributed strategy for generalized biconnectivity maintenance in open multi-robot systems," in *63rd Conference on Decision and Control*, 2024.

[18] Z. Xu, Y. Lv, and Y. Bao, "Adaptive collaborative formation control of unmanned vehicles in open environments," in *International Conference on Unmanned Systems (ICUS)*, IEEE, 2024, pp. 1023–1028.

[19] V. S. Varma, I.-C. Morărescu, and D. Nešić, "Open multi-agent systems with discrete states and stochastic interactions," *IEEE Control Systems Letters*, vol. 2, no. 3, pp. 375–380, 2018.

[20] M. Xue, Y. Tang, W. Ren, and F. Qian, "Stability of multi-dimensional switched systems with an application to open multi-agent systems," *Automatica*, vol. 146, p. 110 644, 2022.

[21] Z. Liao and Y. Wang, "Passivity analysis of nonlinear impulsive multi-dimensional switched systems," in *2021 American Control Conference (ACC)*, IEEE, 2021, pp. 846–851.

[22] A. Mironchenko, "Live systems of varying dimension: Modeling and stability," in *62nd Conference on Decision and Control*, IEEE, 2023, pp. 3956–3961.

[23] J. M. Hendrickx and S. Martin, "Open multi-agent systems: Gossiping with deterministic arrivals and departures," in *2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, IEEE, 2016, pp. 1094–1101.

[24] J. M. Hendrickx and S. Martin, "Open multi-agent systems: Gossiping with random arrivals and departures," in *56th Conference on Decision and Control*, IEEE, 2017, pp. 763–768.

[25] M. Avella-Medina, F. Parise, M. T. Schaub, and S. Segarra, "Centrality measures for graphons: Accounting for uncertainty in networks," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 1, pp. 520–537, 2018.

[26] S. Gao and P. E. Caines, "Graphon control of large-scale networks of linear systems," *IEEE Transactions on Automatic Control*, vol. 65, no. 10, pp. 4090–4105, 2019.

[27] B. Zhou, J. H. Park, Y. Yang, R. Hao, and Y. Jiao, "Prescribed-time consensus of time-varying open multi-agent systems with delays on time scales," *Information Sciences*, vol. 680, p. 120 957, 2024.

[28] Z. A. Z. S. Dashti, C. Seatzu, and M. Franceschelli, "Dynamic consensus on the median value in open multi-agent systems," in *58th Conference on Decision and Control*, IEEE, 2019, pp. 3691–3697.

[29] C. M. de Galland and J. M. Hendrickx, "Lower bound performances for average consensus in open multi-agent systems," in *58th Conference on Decision and Control*, IEEE, 2019, pp. 7429–7434.

[30] C. M. de Galland and J. M. Hendrickx, "Fundamental performance limitations for average consensus in open multi-agent systems," *IEEE Transactions on Automatic Control*, vol. 68, no. 2, pp. 646–659, 2022.

[31] Z. A. Z. S. Dashti, G. Oliva, C. Seatzu, A. Gasparri, and M. Franceschelli, "Distributed mode computation in open multi-agent systems," *IEEE Control Systems Letters*, vol. 6, pp. 3481–3486, 2022.

[32] M. Franceschelli and P. Frasca, "Proportional dynamic consensus in open multi-agent systems," in *57th Conference on Decision and Control*, IEEE, 2018, pp. 900–905.

[33] M. Abdelrahim, J. M. Hendrickx, and W. Heemels, "Max-consensus in open multi-agent systems with gossip interactions," in *IEEE 56th Conference on Decision and Control*, 2017, pp. 4753–4758.

[34] G. Oliva, M. Franceschelli, A. Gasparri, and A. Scala, "A sum-of-states preservation framework for open multi-agent systems with nonlinear heterogeneous coupling," *IEEE Transactions on Automatic Control*, 2023.

[35] C. M. de Galland, R. Vizuete, J. M. Hendrickx, P. Frasca, and E. Panteley, "Random coordinate descent algorithm for open multi-agent systems with complete topology and homogeneous agents," in *60th Conference on Decision and Control*, IEEE, 2021, pp. 1701–1708.

[36] H. W. Abrahamson and E. Wei, "Primetime: A finite-time consensus protocol for open networks," in *62nd Conference on Decision and Control*, IEEE, 2023, pp. 5014–5019.

[37] S. Jiang and Y. Zhang, "Coordination on open double-integrator multi-agent systems," in *2023 42nd Chinese Control Conference (CCC)*, IEEE, 2023, pp. 5687–5692.

[38] Y.-G. Hsieh, F. Iutzeler, J. Malick, and P. Mertikopoulos, "Multi-agent online optimization with delays: Asynchronicity, adaptivity, and optimism," *Journal of Machine Learning Research*, vol. 23, no. 78, pp. 1–49, 2022.

[39] R. Sawamura, N. Hayashi, and M. Inuiguchi, "A distributed primal-dual push-sum algorithm on open multiagent networks," *IEEE Transactions on Automatic Control*, 2024.

[40] Y. Liu, M. Ye, L. Ding, L. Xie, and S. Xu, "Distributed Online Resource Allocation in Open Networks," *IEEE Transactions on Automatic Control*, 2024.

[41] R. Vizuete, C. M. de Galland, J. M. Hendrickx, P. Frasca, and E. Panteley, "Resource allocation in open multi-agent systems: An online optimization analysis," in *61st Conference on Decision and Control*, IEEE, 2022, pp. 5185–5191.

[42] C. M. D. Galland, R. Vizuete, J. M. Hendrickx, E. Panteley, and P. Frasca, "Random Coordinate Descent for Resource Allocation in Open multiagent Systems," *IEEE Transactions on Automatic Control*, 2024.

[43] I. Bistritz and N. Bambos, "Gamekeeper: Online learning for admission control of networked open multiagent systems," *IEEE Transactions on Automatic Control*, 2024.

[44] T. Nakamura, N. Hayashi, and M. Inuiguchi, "Cooperative learning for adversarial multi-armed bandit on open multi-agent systems," *IEEE Control Systems Letters*, vol. 7, pp. 1712–1717, 2023.

[45] E. K. Ryu and S. Boyd, "Primer on monotone operator methods," *Applied Comput. Math*, vol. 15, no. 1, pp. 3–43, 2016.

[46] H. H. Bauschke and P. L. Combettes, *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*, 2nd ed. Cham, Switzerland: Springer, 2017.

[47] N. Bastianello, D. Deplano, M. Franceschelli, and K. H. Johansson, "Robust online learning over networks," *IEEE Transactions on Automatic Control*, vol. 70, no. 2, pp. 933–946, 2025.

[48] L. Elsner, I. Koltracht, and M. Neumann, "Convergence of sequential and asynchronous nonlinear paracontractions," *Numerische mathematik*, vol. 62, no. 1, pp. 305–319, 1992.

[49] A. A. Raja and S. Grammatico, "Payoff distribution in robust coalitional games on time-varying networks," *IEEE Transactions on Control of Network Systems*, vol. 9, no. 1, pp. 511–520, 2021.

[50] A. Rai and S. Mou, "Distributed algorithms for edge-agreements: More than consensus," in *62nd Conference on Decision and Control*, IEEE, 2023, pp. 4417–4422.

[51] X. Li, L. Xie, and N. Li, "A survey on distributed online optimization and online games," *Annual Reviews in Control*, vol. 56, p. 100 904, 2023.

[52] N. Bastianello, R. Carli, L. Schenato, and M. Todescato, "Asynchronous distributed optimization over lossy networks via relaxed ADMM: Stability and linear convergence," *IEEE Transactions on Automatic Control*, vol. 66, no. 6, pp. 2620–2635, 2021.

[53] P. Giselsson and S. Boyd, "Linear convergence and metric selection for douglas-rachford splitting and admm," *IEEE Transactions on Automatic Control*, vol. 62, no. 2, pp. 532–544, 2016.

[54] E. Makridis, A. Grammenos, G. Oliva, E. Kalyvianaki, C. N. Hadjicostis, and T. Charalambous, "Average consensus over directed networks in open multi-agent systems with acknowledgement feedback," in *63rd Conference on Decision and Control*, IEEE, 2024, pp. 3051–3056.

[55] C. N. Hadjicostis and A. D. Dominguez-Garcia, "Distributed average consensus in open multi-agent systems," in *63rd Conference on Decision and Control*, IEEE, 2024, pp. 3037–3042.

[56] M. Zhu and S. Martinez, "Discrete-time dynamic average consensus," *Automatica*, vol. 46, no. 2, pp. 322–329, 2010.

[57] D. Deplano, N. Bastianello, M. Franceschelli, and K. H. Johansson, "A unified approach to solve the dynamic consensus on the average, maximum, and median values with linear convergence," in *62nd Conference on Decision and Control*, IEEE, 2023, pp. 6442–6448.

[58] D. Deplano, M. Franceschelli, and A. Giua, "Dynamic min and max consensus and size estimation of anonymous multiagent networks," *IEEE Transactions on Automatic Control*, vol. 68, no. 1, pp. 202–213, 2023.

[59] N. Parikh and S. Boyd, "Proximal algorithms," *Foundations and Trends® in Optimization*, vol. 1, no. 3, pp. 127–239, 2014.

[60] P. Lancaster and H. K. Farahat, "Norms on direct sums and tensor products," *mathematics of computation*, vol. 26, no. 118, pp. 401–414, 1972.

[61] D. Serre, *Matrices: Theory and Applications (2nd Edition)*. ASpringer New York, NY, 2012.

Appendix: Proofs of Lemmas 1-2 and Theorem 3

A. Proof of Lemma 1

Let $\hat{x}_k = \mathsf{F}_{k+1}(\hat{x}_k)$ be a fixed point of the standard iteration. Since the updates of Open ADMM are the result of the application of the Peaceman-Rachford operator to the dual of the distributed version of the problem in (9), then the output variable at a fixed point is equal to $(\mathbb{1}_{n_k} \otimes y_k^\star)$ where $y_k^\star \in \mathcal{Y}_k^\star$ is a solution to problem in (9) (cfr. [46, Therorem 26.11]), i.e.,

$$(\mathbb{1}_{n_k} \otimes y_k^\star) = \text{prox}_{f_k}^{1/\rho\eta_k}(D_k A_k^\top \hat{x}_k). \tag{21}$$

Thus, the fixed point $\hat{x}_k$ of the standard iteration satisfies

$$(I + P_k)\hat{x}_k = 2\rho P_k A_k (\mathbb{1}_{n_k} \otimes y_k^\star),$$

thus completing the first part of the proof. Now, we need to prove that the following is bounded from above,

$$d_{\text{SH}}(\hat{\mathcal{X}}_k, \hat{\mathcal{X}}_{k-1}) = \sup_{z\in\mathbb{R}^{\mathcal{I}_k}} d(\text{proj}(z, \hat{\mathcal{X}}_k), \text{proj}(z, \hat{\mathcal{X}}_{k-1})).$$

Let us denote by $\hat{z}_k$ the projection of $z \in \mathbb{R}^{\mathcal{I}_k}$ onto $\hat{\mathcal{X}}_k \subseteq \mathbb{R}^{\mathcal{I}_k}$, namely $\hat{z}_k = \text{proj}(z, \hat{\mathcal{X}}_k)$, then by [59, Section 6.2.2] it holds

$$\hat{z}_k = z - (I + P_k)^\dagger((I + P_k)z - 2\rho P_k A_k(\mathbb{1}_{n_k} \otimes y_k^\star)).$$

Since $I + P_k = I_{\xi_k/2} \otimes (J \otimes I_p)$ where $J = (\Pi + I_2)$ is a matrix with all ones with pseudoinverse $J^\dagger = \frac{1}{4}J$, we can decompose $\hat{z}_k$ into vectors $\hat{z}_k^{ij} \in \mathbb{R}^{2p}$ where $i \in \mathcal{V}_k$, $j \in \mathcal{N}_k^i$ such that $j > i$, given by

$$\hat{z}_k^{ij} = z^{ij} - \frac{1}{2}(J \otimes I_p)(z^{ij} - \rho(\mathbb{1}_2 \otimes y_k^\star)).$$

Similarly, the components in $\mathcal{I}_k$ of $\text{proj}(z, \hat{\mathcal{X}}_{k-1})$ are given by

$$\hat{z}_{k-1}^{ij} = z^{ij} - \frac{1}{2}(J \otimes I_p)(z^{ij} - \rho(\mathbb{1}_2 \otimes y_{k-1}^\star)).$$

Let $\mathcal{R}_k^i = \mathcal{N}_k^i \cap \mathcal{N}_{k-1}^i$. Thus, $d(\text{proj}(z, \hat{\mathcal{X}}_k), \text{proj}(z, \hat{\mathcal{X}}_{k-1})) =$

$$= \sqrt{\sum_{i\in\mathcal{V}_k^{\text{R}}} \sum_{i<j\in\mathcal{R}_k^i} \left\| \hat{z}_k^{ij} - \hat{z}_{k-1}^{ij} \right\|^2}$$

$$= \sqrt{\sum_{i\in\mathcal{V}_k^{\text{R}}} \sum_{i<j\in\mathcal{R}_k^i} \frac{\rho^2}{4} \left\| (J\otimes I_p)(\mathbb{1}_2\otimes(y_k^\star - y_{k-1}^\star)) \right\|^2}$$

$$\overset{(i)}{\leq} \sqrt{\sum_{i\in\mathcal{V}_k^{\text{R}}} \sum_{i<j\in\mathcal{R}_k^i} \frac{\rho^2}{4} \|(J\otimes I_p)\|^2 \left\| \mathbb{1}_2\otimes(y_k^\star - y_{k-1}^\star) \right\|^2}$$

$$\overset{(ii)}{\leq} \sqrt{\sum_{i\in\mathcal{V}_k^{\text{R}}} \sum_{i<j\in\mathcal{R}_k^i} \frac{\rho^2\sigma^2}{2} \|J\otimes I_p\|^2} \overset{(iii)}{=} \sqrt{\sum_{i\in\mathcal{V}_k^{\text{R}}} \sum_{i<j\in\mathcal{R}_k^i} \frac{\rho^2\sigma^2}{2} \|J\|^2 \|I_p\|^2}$$

$$\overset{(iv)}{=} \sqrt{\sum_{i\in\mathcal{V}_k^{\text{R}}} \sum_{i<j\in\mathcal{R}_k^i} \frac{\rho^2\sigma^2}{2} \|J\|^2} \overset{(v)}{\leq} \sqrt{\sum_{i\in\mathcal{V}_k^{\text{R}}} \sum_{i<j\in\mathcal{R}_k^i} \frac{\rho^2\sigma^2}{2} \|J\|_1 \|J\|_\infty}$$

$$\overset{(vi)}{=} \sqrt{\sum_{i\in\mathcal{V}_k^{\text{R}}} \sum_{i<j\in\mathcal{R}_k^i} \frac{\rho^2\sigma^2}{2} \cdot 2\cdot 2} = \rho\sigma \sqrt{\sum_{i\in\mathcal{V}_k^{\text{R}}} \sum_{i<j\in\mathcal{R}_k^i} 2} \overset{(vii)}{\leq} \rho\sigma\sqrt{|\mathcal{R}_k|}$$

where (i) follows by sub-multiplicativity of the norm; (ii) holds by Assumption 2(iii) and the properties of the Kronecker product; (iii) follows by [60, Theorem 8]; (iv) holds because the 2-norm of an identity matrix is equal to 1; (v) follows by the Riesz-Thorin Theorem [61, Theorem 4.3.1]; (vi) follows by the fact that $\|M\|_1$ and $\|M\|_\infty$ are, respectively, the row- and column- sum of the absolute values of the matrix M, and from the fact that J has exactly 2 ones in each row and each column; (vii) holds because the number of remaining channels at time k is equal to $|\mathcal{R}_k|$. Therefore, the TSI is bounded with $B = \rho\sigma$, completing the proof.

B. Proof of Lemma 2

Open ADMM requires that state components x_k^{ij} of all arriving agents $i \in \mathcal{V}_k^{\text{A}}$ and new state components of the remaining agents $i \in \mathcal{V}_k^{\text{R}}$, $j \in \mathcal{A}_k^i$ are initialized to the minimizer $y_k^{i,\star}$ of the local cost f_k^i, scaled by the penalty parameter $\rho > 0$, namely,

$$x_k^{ij} = \rho y_k^{i,\star}, \text{ where } y_k^{i,\star} \in \mathcal{Y}_k^{i,\star} = \left\{ y : f_k^i(y) = \min_{z\in\mathbb{R}^p} f_k^i(z) \right\}.$$

Let $x_k^{\text{A}} \in \mathbb{R}^{\mathcal{A}_k}$ be the vector stacking the components of all arriving labels, and denote $\hat{x}_k^{\text{A}} = \text{proj}(x_k^{\text{A}}, \hat{\mathcal{X}}_k)$. Further, let us decompose $x_k^{\text{A}}, \hat{x}_k^{\text{A}}$ into vectors $x_k^{ij}, \hat{x}_k^{ij} \in \mathbb{R}^p$ where $i \in \mathcal{V}_k$, $j \in \mathcal{A}_k^i$. Then, by Lemma 1 it holds that:

$$d(x_k^{\text{A}}, \hat{\mathcal{X}}_k) = d(x_k^{\text{A}}, \text{proj}(x_k^{\text{A}}, \hat{\mathcal{X}}_k)) = d(x_k^{\text{A}}, \hat{x}_k^{\text{A}})$$

$$= \sqrt{\sum_{i\in\mathcal{V}_k} \sum_{i<j\in\mathcal{A}_k^i} \left\| \begin{bmatrix} x_k^{ij} \\ x_k^{ji} \end{bmatrix} - \begin{bmatrix} \hat{x}_k^{ij} \\ \hat{x}_k^{ji} \end{bmatrix} \right\|^2} = \sqrt{\sum_{i\in\mathcal{V}_k} \sum_{i<j\in\mathcal{A}_k^i} \left\| \rho \begin{bmatrix} y_k^{i,\star} \\ y_k^{j,\star} \end{bmatrix} - \begin{bmatrix} \hat{x}_k^{ij} \\ \hat{x}_k^{ji} \end{bmatrix} \right\|^2}$$

$$\overset{(i)}{\leq} \sqrt{\sum_{i\in\mathcal{V}_k} \sum_{i<j\in\mathcal{A}_k^i} \left\| \frac{\rho}{2}(J\otimes I_p) \left(\begin{bmatrix} y_k^{i,\star} \\ y_k^{j,\star} \end{bmatrix} - (\mathbb{1}_2 \otimes y_{k-1}^\star) \right) \right\|^2}$$

$$\overset{(ii)}{\leq} \sqrt{\frac{\rho^2}{4} \|(J\otimes I_p)\|^2 \sum_{i\in\mathcal{V}_k} \sum_{i<j\in\mathcal{A}_k^i} \left\| \begin{bmatrix} y_k^{i,\star} - y_k^\star \\ y_k^{j,\star} - y_k^\star \end{bmatrix} \right\|^2}$$

$$\overset{(iii)}{\leq} \sqrt{\rho^2 \sum_{i\in\mathcal{V}_k} \sum_{i<j\in\mathcal{A}_k^i} 2\omega^2} = \rho\omega \sqrt{\sum_{i\in\mathcal{V}_k} \sum_{i<j\in\mathcal{A}_k^i} 2} \overset{(iv)}{\leq} \rho\omega\sqrt{|\mathcal{A}_k|},$$

where (i) hold by Lemma 1; (ii) holds by triangle inequality; (iii) holds as explained in steps $(iii) - (vi)$ at the end of the proof of Lemma 1 and by Assumption 2(iv); (iv) holds because the number of arriving channels at time k is equal to $|\mathcal{A}_k|$. Therefore, the arrival process is bounded with $H = \frac{\rho\omega}{\sqrt{p}}$, completing the proof.

C. Proof of Theorem 3

Denoting by x_k^i the vector stacking the edge variables x_k^{ij} for $j \in \mathcal{N}_k^i$ of node $i \in \mathcal{V}_k$, we first compute an upper bound to the distance between the local estimate y_k^i and the set of optimal solution $\mathcal{Y}_k^\star$:

$$
\begin{aligned}
d(y_k^i,\mathcal{Y}_k^\star) &= \inf_{y\in\mathcal{Y}_k^\star}\left\|y_k^i - y\right\| \overset{(i)}{\le} \left\|y_k^i - y_k^\star\right\| \\
&\overset{(ii)}{\le} \left\| \mathrm{prox}_{f_k^i}^{1/\rho\eta_k^i}\left(\frac{1}{\rho\eta_k^i}A_{i,k}^\top x_k^i\right) - \mathrm{prox}_{f_k^i}^{1/\rho\eta_k^i}\left(\frac{1}{\rho\eta_k^i}A_{i,k}^\top \hat{x}_k^i\right)\right\| \\
&\overset{(iii)}{\le} \left\|\frac{1}{\rho\eta_k^i}A_{i,k}^\top (x_k^i - \hat{x}_k^i)\right\| \overset{(iv)}{\le} \left\|\frac{1}{\rho\eta_k^i}A_{i,k}^\top\right\| \left\|x_k^i - \hat{x}_k^i\right\| \\
&\overset{(v)}{=} \frac{1}{\rho\sqrt{\eta_k^i}} d(x_k^i, \hat{\mathcal{X}}_k),
\end{aligned} \tag{22}
$$

where (i) holds since $y_k^\star \in \mathcal{Y}_k^\star$ (ii) holds by eqs. (12) and (21), where $A_{i,k}^\top$ is an opportune slicing of $A_k^\top$; (iii) follows by the non-expansiveness of the proximal; (iv) holds by choosing $\hat{x}_k = \mathrm{arginf}_{y\in\hat{\mathcal{X}}_k}\|x_k - y\|$; (v) holds because matrix $\frac{1}{\eta_k^i}A_{i,k}^\top$ is column-stochastic and its rows sum up to $1/\eta_k^i$.

We now compute an upper bound to the distance between the estimation vector $\widetilde{y}_k$ of the whole network as in eq. (16) from the consensus state $\widetilde{\mathcal{C}}_k^\star$ on the solutions as in eq. (17):

$$
\begin{aligned}
d(\widetilde{y}_k, \widetilde{\mathcal{C}}_k^\star) &= \sqrt{\sum_{i\in\mathcal{V}_k}\sum_{j\in\mathcal{N}_k^i} d^2(y_k^i, y_k^\star)} = \sqrt{\sum_{i\in\mathcal{V}_k} \eta_k^i d^2(y_k^i, y_k^\star)} \\
&\overset{(i)}{\le} \frac{1}{\rho}\sqrt{\sum_{i\in\mathcal{V}_k} d^2(x_k^i, \hat{\mathcal{X}}_k^\star)} = \frac{1}{\rho} d(x_k, \hat{\mathcal{X}}_k^\star),
\end{aligned}
$$

where (i) holds by $d(y_k^i, \mathcal{Y}_k^\star) \le (\rho^2\eta_k^i)^{-1/2} d(x_k^i, \hat{\mathcal{X}}_k)$ as per eq. (22). This means that the linear convergence of $\widetilde{y}_k$ to a neighborhood of $\widetilde{\mathcal{C}}_k^\star$ is implied by that of x_k to a neighborhood of $\hat{\mathcal{X}}_k$, which follows from Theorem 2 and Lemmas 1-2, up to a convergence radius computed as follows

$$
\limsup_{k\to\infty} \frac{d(\widetilde{y}_k, \widetilde{\mathcal{C}}_k^\star)}{\sqrt{p\xi_k}} \le \limsup_{k\to\infty} \frac{d(x_k, \hat{\mathcal{X}}_k)}{\rho\sqrt{p\xi_k}} = \frac{1}{\rho}\frac{B+H}{1-\frac{\gamma}{\beta}}
$$

Using $B = \rho\sigma$ in (14), $H = \rho\omega$ in (15), yields the thesis.

Diego Deplano (S'17–M'21) is Assistant Professor (RTD-A) at the Department of Electrical and Electronic Engineering, University of Cagliari, Italy. He received the B.S., M.S., and PhD degrees in Electronic Engineering "cum laude" from the University of Cagliari, Italy, respectively in '15, '17, and '21. He was awarded the best PhD Thesis defended in the area of Systems and Control Engineering at an Italian University by SIDRA. He spent visiting periods at the Nanyang Technological University (NTU), Singapore, at the Centre National de la Recherche Scientifique (CNRS), Grenoble, France, and at the University of Toronto (UofT), Toronto, Canada. He has lectured seminars at KTH in Stockholm, Sweden, and at Peking University in Beijing, China. He is a member of the Conference Editorial Board (CEB) for the IEEE Control Systems Society (CSS) and serves as Associate Editor for the IEEE Conference on Decision and Control (CDC) and the IEEE American Control Conference (ACC) since 2025. He also serves as an Associate Editor for the IEEE Conference on Automation Science and Engineering (CASE) since 2024. His research interests include open multi-agent systems, distributed optimization and learning, dynamic consensus problems, monotone theory, contraction theory.

Nicola Bastianello (M'18) is a post-doc at the School of Electrical Engineering and Computer Science, KTH Royal Institute of Technology, Sweden. From 2021 to 2022 he was a post-doc at the Department of Information Engineering (DEI), University of Padova, Italy. He received the Ph.D. in Information Engineering at the University of Padova, Italy in 2021. During the Ph.D. he was a visiting student at the Department of Electrical, Computer, and Energy Engineering (ECEE), University of Colorado Boulder, Colorado, USA. He received the master degree in Automation Engineering (2018) and the bachelor degree in Information Engineering (2015) from the University of Padova, Italy. He currently serves in the IEEE CSS and EUCA Conference Editorial Boards. His research lies at the intersection of optimization and learning, with a focus on multi-agent systems.

Mauro Franceschelli (M'11–SM'22) is Associate Professor at the Department of Electrical and Electronic Engineering, University of Cagliari, Italy. He received the Laurea degree in Electronic Engineering "cum laude" in 2007 and the PhD degree in 2011 from the University of Cagliari. He spent visiting periods at the Georgia Institute of Technology (GaTech), and the University of California at Santa Barbara (UCSB), USA. In 2013 he received a fellowship from the National Natural Science Foundation of China (NSFC), at Xidian University, Xi'an, China. In 2015 he was awarded a position of Assistant Professor (RTD-A) funded by the Italian Ministry of Education, University and Research (MIUR) under the 2014 call "Scientific Independence of Young Researchers" (SIR). He is currently an Associate Editor for the IEEE Transactions on Automatic Control and IFAC Nonlinear Analysis: Hybrid Systems (NAHS). He is member of the Conference Editorial Board (CEB) for the IEEE Control Systems Society (CSS) since 2019. He serves as Associate Editor for the IEEE Conference on Automation Science and Engineering (CASE) since 2015. He was an Associate Editor for the IEEE Transactions on Automation Science and Engineering (T-ASE) 2021-2023 and Guest Associate Editor for the IEEE Open Journal of Control Systems (OJ-CSYS) in 2024. He is IEEE Senior Member and Officer (vice-Chair) for the IEEE Italy Section Chapter of the IEEE Control Systems Society since 2023. His research interests include consensus problems, gossip algorithms, multi-agent systems, multi-robot systems, distributed optimization and electric demand side management.

Karl H. Johansson is Swedish Research Council Distinguished Professor in Electrical Engineering and Computer Science at KTH Royal Institute of Technology in Sweden and Founding Director of Digital Futures. He earned his MSc degree in Electrical Engineering and PhD in Automatic Control from Lund University. He has held visiting positions at UC Berkeley, Caltech, NTU and other prestigious institutions. His research interests focus on networked control systems and cyber-physical systems with applications in transportation, energy, and automation networks. For his scientific contributions, he has received numerous best paper awards and various distinctions from IEEE, IFAC, and other organizations. He has been awarded Distinguished Professor by the Swedish Research Council, Wallenberg Scholar by the Knut and Alice Wallenberg Foundation, Future Research Leader by the Swedish Foundation for Strategic Research. He has also received the triennial IFAC Young Author Prize and IEEE CSS Distinguished Lecturer. He is the recipient of the 2024 IEEE CSS Hendrik W. Bode Lecture Prize. His extensive service to the academic community includes being President of the European Control Association, IEEE CSS Vice President Diversity, Outreach & Development, and Member of IEEE CSS Board of Governors and IFAC Council. He has served on the editorial boards of Automatica, IEEE TAC, IEEE TCNS and many other journals. He has also been a member of the Swedish Scientific Council for Natural Sciences and Engineering Sciences. He is Fellow of both the IEEE and the Royal Swedish Academy of Engineering Sciences.