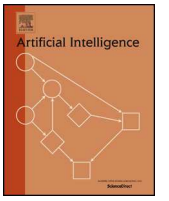
Artificial Intelligence 331 (2024) 104094

Contents lists available at ScienceDirect

Artificial Intelligence

journal homepage: www.elsevier.com/locate/artint



Decentralized fused-learner architectures for Bayesian reinforcement learning ☆

Augustin A. Saucan [a,1], Subhro Das [b], Moe Z. Win [c,*]

[a] Wireless Information and Network Sciences Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139, USA
[b] MIT-IBM Watson AI Lab, IBM Research, Cambridge, MA 02139, USA
[c] Laboratory for Information and Decision Systems (LIDS), Massachusetts Institute of Technology, Cambridge, MA 02139, USA

ARTICLE INFO

Keywords:
Bayesian reinforcement learning
Decentralized training
Information fusion
Actor critic algorithms
Kullback–Leibler divergence

ABSTRACT

Decentralized training is a robust solution for learning over an extensive network of distributed agents. Many existing solutions involve the averaging of locally inferred parameters which constrain the architecture to independent agents with identical learning algorithms. Here, we propose decentralized fused-learner architectures for Bayesian reinforcement learning, named fused Bayesian-learner architectures (FBLAs), that are capable of learning an optimal policy by fusing potentially heterogeneous Bayesian policy gradient learners, i.e., agents that employ different learning architectures to estimate the gradient of a control policy. The novelty of FBLAs relies on fusing the full posterior distributions of the local policy gradients. The inclusion of higher-order information, i.e., probabilistic uncertainty, is employed to robustly fuse the locally-trained parameters. FBLAs find the barycenter of all local posterior densities by minimizing the total Kullback–Leibler divergence from the barycenter distribution to the local posterior densities. The proposed FBLAs are demonstrated on a sensor-selection problem for Bernoulli tracking, where multiple sensors observe a dynamic target and only a subset of sensors is allowed to be active at any time.

1. Introduction

Cooperative multi-agent learning systems [1,2] are comprised of a multitude of individual agents that operate in an environment toward a shared goal. Through sensing, communication, and cooperation, multi-agent systems are capable of addressing many of today's challenging tasks such as network navigation and localization [3–6], multi-object tracking [7–9], resource management [10–12], simultaneous mapping and localization [13], autonomous driving [14], network packet delivery [15], traffic-light management [16], and e-commerce logistics [17]. Accurate and scalable solutions are especially critical in massive multi-agent systems operating in challenging environments with limited observability and communication capabilities. For example, sensor management for centralized [18–20] and distributed [21–24] target tracking comprises sensor selection, scheduling, and frequency/timeslot allocation. Efficient sensor management is necessary for ensuring accurate object tracking in robotics, safe operation of agents in factory of the future, as well as data collection in wireless networks [25–28]. Planning of agent paths and their interaction

☆ This paper is part of the Special Issue: "Risk-aware Autonomous Systems: Theory and Practice".

* Corresponding author.
E-mail addresses: augustin.saucan@telecom-sudparis.eu (A.A. Saucan), subhro.das@ibm.com (S. Das), moewin@mit.edu (M.Z. Win).
[1] A. A. Saucan is now with the SAMOVAR Laboratory, Télécom SudParis, Institut Polytechnique de Paris, 91120 Palaiseau, France.

https://doi.org/10.1016/j.artint.2024.104094
Received 15 May 2022; Received in revised form 30 August 2023; Accepted 31 January 2024

Available online 13 February 2024

is highly relevant for heterogeneous networks of agents with different sensing modalities and where the fusion of information across the entire network is necessary to unambiguously localize and track objects. Many robust multi-agent systems are decentralized [29], that is, there is no centralized fusion node that collects all observations and dispatches control policies to its constituent agents. Accordingly, decentralized inference algorithms [23,30,31] rely on a communication graph in order to achieve network consensus through neighbor-to-neighbor communication.

Multi-simulator training enabled today's success of deep reinforcement learning (RL) [32] through the stabilization of the learning process and by achieving higher learning throughput [33]. These achievements were obtained by distributing computation across several agents wherein training data is locally simulated. Centralized training of decentralized policies [34–36] has emerged as a solution to reduce the complexity of multi-agent RL. These solutions assume a centralized communication graph during training; while during the execution phase, each agent operates based solely on its local action-observation history or based on a limited communication channel with neighboring agents.

Several decentralized training algorithms for RL [37–43] were proposed in recent years by employing various distributed estimation methods such as gossiping, consensus, diffusion, and consensus+innovations. In [37], each agent implements the actor-critic (AC) method of [44,45] with a parametric representation for both the actor (i.e., policy) and critic (i.e., state-action value or Q function) where network consensus is achieved for both sets of parameters in an asynchronous manner. Similarly in [42], asynchronous gossip is employed to reach consensus for both actor and critic with non-linear function approximation. A consensus+innovation strategy is employed in [38] to learn the optimal Q function in a tabular case, i.e., finite state and action spaces without parametrization. Relying on linearly parametrized state-value functions and a primal-dual argument, the works of [39,41] achieve fully decentralized RL solutions. By incorporating updates from neighboring agents, a gossip-based architecture for distributed temporal difference RL was proposed in [40]. Of further note in [40] is the ability of each agent to employ a different set of features to represent the state-value function. A decentralized primal-dual formulation is employed in [43] coupled with a non-linear function approximation for the state-value function.

In this work, we propose two decentralized fused Bayesian-learner architectures (FBLAs) for RL, that we name FBLA with precision matrix gossip (FBLA-PMG) and FBLA with precision matrix gossip and consensus (FBLA-PMGC), for which the full posterior distribution of individual learning agents is fused together in a communication-efficient manner. The agents locally simulate the environment by using a parametric policy and learn a posterior density estimate of the policy gradient using potentially different Bayesian methods, e.g., Bayesian policy gradient (BPG) and Bayesian actor-critic (BAC). Learner fusion is achieved by minimizing a weighted average Kullback–Leibler divergence (KLD) from each of the local posteriors. The Gaussian nature of the posteriors resulting from Bayesian RL is harnessed to yield a closed-form solution for the KLD minimization problem. For both FBLAs architectures, the first and second order moments of the local posteriors are shared among neighboring agents through gossiping. Consensus for both mean and covariance are not necessary for either FBLAs, however, FBLA-PMGC performs an additional precision matrix consensus (PMC) step for precision (and implicitly covariance) consensus across all agents. Both FBLAs architectures are shown to maintain the local moment estimates to be within an ϵ-ball of each other.

In summary, our contributions are twofold:

- the development of gossip-based architectures for decentralized training of Bayesian RL agents using an average KLD fusion criterion,
- the derivation of ϵ-ball guarantees that bound the differences of the first and second order moments of the locally fused policies across the different agents.

To the best of the authors knowledge, the FBLAs are the first Bayesian RL architectures to be proposed for decentralized training over a network of agents that implement Bayesian RL methods (e.g., BPG or BAC). In comparison to the state-of-the-art works of [42,46,47], the FBLAs novelty is given by: (i) extending the single-agent Bayesian RL methods of [46] and [47] to a network of multi-agent learners, and (ii) devising a gossip-based architecture that fuses the entire distributions of the local policy gradient estimators as opposed to just the first-order statistics (i.e., mean estimates) as done in [42].

Several advantages arise of the proposed FBLA. First, the FBLAs fuse the full posterior information of the agents as opposed to just the first-order (i.e., mean value) statistics. Second, the FBLAs can merge results from heterogeneous agents, i.e., BPG or BAC with different dictionaries for the local critics can be fused together. This feature can lead to a faster exploration of the policy space and a better adaptability to the computational and memory capabilities of each agent. Thirdly, the dimension of communication messages in the FBLAs are $O(d^2)$, where d is the number of policy parameters, since only actor-related quantities (the mean vector and covariance matrix of the policy gradient) are exchanged among neighboring agents. This is in contrast to fusing critic-related quantities, which would involve a much larger communication overhead due to the dependence of the posterior Gaussian process (GP) critic on the number of training samples.

In closing, we demonstrate the FBLAs through numerical experimentation in the case of a sensor selection problem for Bernoulli tracking. More specifically, a dynamical Bernoulli random finite set (RFS) is observed through an extensive network of non-linear sensors and the objective is to find a sensor activation policy that maximizes the inferred information on the Bernoulli RFS while imposing a constraint on the number of sensors that can be active at any time. The setting for this application is borrowed from [48]. However, this work differs significantly from that of [48] based on two main aspects: (i) no multi-agent learning is considered in [48] and (ii) the control policy of [48] differs from the one considered in this work (see Sec. 4.1 for details). Here, multiple learning agents locally implement various BPG or BAC algorithms and fuse their policy gradient posteriors through the FBLAs. We

showcase the improved training results of the FBLAs when contrasted with a fusion method that averages the posterior moments of the local posteriors.

The rest of the paper is organized as follows. Background is provided in Sec. 2. The proposed fused architectures are presented in Sec. 3, while numerical results are given in Sec. 4.

2. Background

Notation. Random variables are displayed in sans serif, upright fonts; their realizations in serif, italic fonts. Vectors and matrices are denoted by bold lowercase and uppercase letters, respectively. Random sets and their realizations are denoted by upright sans serif and calligraphic font, respectively. Set cardinality is denoted as $\#\mathcal{X}$. For any non-negative integers $a < b$, we denote the set $\mathbb{N}_a^b \triangleq \{a, a+1, \dots, b\}$. The expectation operator is denoted via $\mathbb{E}\{\cdot\}$ while the covariance operator is denoted with $\mathbb{C}\mathrm{ov}\{\cdot,\cdot\}$.

A random vector $\mathbf{x}$ that obeys a Gaussian distribution with mean vector $\boldsymbol{m}$ and covariance matrix $\boldsymbol{P}$ is denoted via $\mathbf{x} \sim f_{\mathrm{G}}(\boldsymbol{m}, \boldsymbol{P})$. Whenever $\boldsymbol{P}$ is positive definite, the multidimensional Gaussian probability density function is denoted via $f_{\mathrm{G}}(\cdot; \boldsymbol{m}, \boldsymbol{P})$. The Kronecker product is denoted with $\otimes$. For a n-tuple of positive definite matrices $\left(\boldsymbol{P}_i\right)_{i=1}^n$, a max operator $\tilde{\boldsymbol{P}} = \max\left\{\left(\boldsymbol{P}_i\right)_{i=1}^n\right\}$ is defined as $\tilde{\boldsymbol{P}} = \boldsymbol{P}_{i_*}$ where $i_* = \operatorname{argmin}\left(\rho\left(\boldsymbol{P}_i^{-1}\right) : i \in \mathbb{N}_1^n\right)$ and $\rho(\cdot)$ is the spectral radius operator.

A stochastic process is a collection of random variables $\{\mathsf{f}(x)\}_{x\in\mathcal{X}}$, where the covariates $\boldsymbol{x}$ take values in some (potentially uncountable) set $\mathcal{X}$, e.g., $\mathcal{X} \subset \mathbb{R}^n$. A GP is a collection of random variables, any finite number of which jointly obey a Gaussian distribution [49, def. 2.1]. A GP [49] is denoted via $\mathsf{f} \sim f_{\mathrm{GP}}(\mu, \kappa)$, where μ is the mean function on $\mathcal{X}$ and κ is a covariance function, i.e., a symmetric function of positive type on $\mathcal{X}\times\mathcal{X}$ [50, Th. 1.11] (see also [49, Ch. 4]). Moreover, for a GP $\mathsf{f} \sim f_{\mathrm{GP}}(\mu, \kappa)$ and for any finite set $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_n\} \subset \mathcal{X}$, the vector of random variables $[\mathsf{f}(\boldsymbol{x}_1), \mathsf{f}(\boldsymbol{x}_2), \dots, \mathsf{f}(\boldsymbol{x}_n)]^\top$ is Gaussian distributed with mean vector $\boldsymbol{\mu} = [\mu(\boldsymbol{x}_1), \mu(\boldsymbol{x}_2), \dots, \mu(\boldsymbol{x}_n)]^\top$ and covariance matrix $\boldsymbol{K}$ with entries $[\boldsymbol{K}]_{ij} = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$ for $1 \leqslant i, j \leqslant n$. The positive semidefinite nature of $\boldsymbol{K}$ follows from κ being a covariance function (see [49, p. 80]).

Reinforcement learning. RL [51–54] refers to a class of learning problems where one or several agents aim to optimize a measure of long-term performance by interacting with a stochastic environment. In general, such interactions are modeled via a Markov decision process (MDP) represented by the tuple $\left(\mathcal{X}, \mathcal{U}, f, h, f_0\right)$, where $\mathcal{X}$ and $\mathcal{U}$ are the state and action (or control) spaces respectively, $f(\cdot|\boldsymbol{x}, \boldsymbol{u})$ is the probability distribution of transitioning from state $\boldsymbol{x}$ when taking action $\boldsymbol{u}$, and $f_0(\cdot)$ is the initial state distribution. The reward of taking action $\boldsymbol{u}$ in state $\boldsymbol{x}$ is represented by the random variable $\mathsf{r}(\boldsymbol{x}, \boldsymbol{u})$, and has distribution $h(\cdot|\boldsymbol{x}, \boldsymbol{u})$. Furthermore, we assume the existence of a stationary policy $\pi(\cdot|\boldsymbol{x}; \boldsymbol{\theta})$ that dictates the probability distribution of actions taken in state $\boldsymbol{x}$ and which is smoothly parameterized by a vector $\boldsymbol{\theta} \in \Theta \subset \mathbb{R}^d$.

Denoting the joint state-action tuple by $\boldsymbol{y} = [\boldsymbol{x}, \boldsymbol{u}]^\top$ and the corresponding space by $\mathcal{Y} = \mathcal{X}\times\mathcal{U}$, we construct a Markov chain with transition probability density function $f^\pi(\boldsymbol{y}_t|\boldsymbol{y}_{t-1}) = \pi(\boldsymbol{u}_t|\boldsymbol{x}_t; \boldsymbol{\theta}) f(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{u}_{t-1})$. The agent maximizes the expected cumulative reward $\eta(\boldsymbol{\theta}) = \mathbb{E}\left\{\sum_{t=0}^\infty \gamma^t \mathsf{r}(\mathbf{x}_t, \mathbf{u}_t)|\pi\right\}$ with respect to the policy parameter $\boldsymbol{\theta}$. The discount parameter $\gamma \in [0, 1)$ controls the relevance of future rewards in the decision process at the current state. The state-action function, also called Q-function, is defined as $Q^\pi(\boldsymbol{x}, \boldsymbol{u}) = \mathbb{E}\left\{\sum_{t=0}^\infty \gamma^t \mathsf{r}(\mathbf{x}_t, \mathbf{u}_t)|\mathbf{x}_0 = \boldsymbol{x}, \mathbf{u}_0 = \boldsymbol{u}; \pi\right\}$. Furthermore, by defining the discounted weighting function for state-action pairs as

$$d^\pi(\boldsymbol{y}) = \sum_{k=0}^\infty \gamma^k \int_{\mathcal{Y}^k} f_0(\boldsymbol{y}_0) f^\pi(\boldsymbol{y}|\boldsymbol{y}_{k-1}) \prod_{i=1}^{k-1} f^\pi(\boldsymbol{y}_i|\boldsymbol{y}_{i-1}) \mathrm{d}(\boldsymbol{y}_0 \cdots \boldsymbol{y}_{k-1}) \tag{1}$$

and assuming the class of policies $\{\pi(\cdot|\boldsymbol{x}; \boldsymbol{\theta}) : \boldsymbol{\theta} \in \Theta\}$ to be sufficiently smooth (see [47]), then the policy-gradient theorem [44,45] relates the gradient of the cumulative reward to the Q function via

$$\boldsymbol{g}(\boldsymbol{\theta}) \triangleq \nabla_{\boldsymbol{\theta}} \eta(\boldsymbol{\theta}) = \int_{\mathcal{Y}} \nabla_{\boldsymbol{\theta}} \log \pi(\boldsymbol{u}|\boldsymbol{x}; \boldsymbol{\theta}) Q^\pi(\boldsymbol{y}) d^\pi(\boldsymbol{y}) \mathrm{d}\boldsymbol{y}\,. \tag{2}$$

Gossip algorithms. Gossip algorithms [30,31] serve to compute averages across multiple nodes interconnected in a peer-to-peer graph topology by only using neighbor-to-neighbor communications. Let the communication graph be denoted via the directed graph $\mathcal{G}_k(\mathcal{V}, \mathcal{E}_k)$, where $\mathcal{V} = \{v_i : i \in \mathbb{N}_1^N\}$ is the set of agents. The edge set $\mathcal{E}_k$ encodes the communication links between agents at time k, i.e., $e_{ij} \in \mathcal{E}_k$ if agent v_i can send information to agent v_j at time step k. The graph diameter is denoted with $\mathrm{diam}(\mathcal{G}_k)$. Furthermore, let $\boldsymbol{x}_i \in \mathbb{R}^d$ denote the local vector of the i-th agent and $\boldsymbol{X} = [\boldsymbol{x}_1, \dots, \boldsymbol{x}_N]^\top$ the row-wise concatenation of these vectors. Gossip algorithms iteratively perform updates of the form $\boldsymbol{X}^{(k+1)} = \boldsymbol{\Omega}^{(k)} \boldsymbol{X}^{(k)}$ for $k = 0, 1, \dots$, with the initialization $\boldsymbol{X}^{(0)} = \boldsymbol{X}$. The mixing matrix $\boldsymbol{\Omega}^{(k)}$ is determined by the edge set $\mathcal{E}_k$ and is allowed to be time varying. Several constructions of the mixing matrix are possible, however, all must respect the graph topology in the sense that $[\boldsymbol{\Omega}^{(k)}]_{ij} \geqslant 0\ \forall i, j \in \mathcal{V}$ and $[\boldsymbol{\Omega}^{(k)}]_{ij} = 0$ if $e_{ij} \notin \mathcal{E}_k$. Under the assumption of symmetric and doubly-stochastic mixing matrices $\boldsymbol{\Omega}^{(k)}\ \forall k$, that is, $[\boldsymbol{\Omega}^{(k)}]_{ij} = [\boldsymbol{\Omega}^{(k)}]_{ji} \in [0, 1]\ \forall i, j$, $\sum_{i=1}^N [\boldsymbol{\Omega}^{(k)}]_{ij} = 1\ \forall j$, and $\sum_{j=1}^N [\boldsymbol{\Omega}^{(k)}]_{ij} = 1\ \forall i$, gossip iterations converge to the average of the initial agent values, i.e., $\lim_{k\to\infty} \boldsymbol{x}_i^{(k)} = \frac{1}{N}\sum_{j=1}^N \boldsymbol{x}_j^{(0)}$ for each agent $v_i \in \mathcal{V}$ (see [31] for more details on gossip algorithms).

3. The fused Bayesian-learner architecture

In this section the proposed fused Bayesian-learner architecture is presented. The architecture achieves a decentralized evaluation of the policy gradient in a fully Bayesian manner, i.e., the posterior distributions of local gradients are optimally fused and the new

fused posterior is obtained via gossip. Based on a given set of policy parameters, in Sec. 3.1 we present one example of a local Bayesian learner that estimates a posterior distribution over the space of Q-functions and subsequently over the space of policy gradients given the local training data. Sec. 3.2 presents the fusion mechanism of local gradient posteriors.

3.1. Local Bayesian learner

For completeness, we introduce the BAC method of [46] as one possible learning algorithm implemented by an individual learner. According to its own policy, each agent simulates interactions with the MDP $\left(\mathcal{X}, \mathcal{U}, f, h, f_0\right)$. Even though an agent only has access to its own local training data, the MDP characteristics are considered identical across all agents. Agent $\upsilon_i \in \mathcal{V}$ imposes a GP prior on the space of Q functions, i.e., $\mathsf{Q}_i \sim f_{\mathrm{GP}}(0, \kappa_i)$, with a mean function that is identically zero and a covariance kernel κ_i. The covariance kernels of the agents and hence the prior GP characteristics of their Q processes are allowed to be different in order to accommodate the local agent capabilities. Given an instance of the local policy parameters $\boldsymbol{\theta}_i \in \mathbb{R}^d$, agent $\upsilon_i \in \mathcal{V}$ interacts with the MDP $\left(\mathcal{X}, \mathcal{U}, f, h, f_0\right)$ according to $\pi(\boldsymbol{u}_t|\boldsymbol{x}_t; \boldsymbol{\theta}_i)$ and collects data that is used to update the GP prior on Q_i. Data is generated through the generative model [46]

$$\mathsf{r}_i(\boldsymbol{y}_t) = \mathsf{Q}_i(\boldsymbol{y}_t) - \gamma \mathsf{Q}_i(\boldsymbol{y}_{t+1}) + \mathsf{n}_i(\boldsymbol{y}_t, \boldsymbol{y}_{t+1}) \tag{3}$$

where the covariates $\boldsymbol{y}_t = (\boldsymbol{x}_t, \boldsymbol{u}_t)$ for $t \in \mathbb{N}_0^{t_i}$ represent a sampled trajectory of the Markov chain $\mathbf{y}_t = [\mathbf{x}_t, \mathbf{u}_t]^\top$ with transitions $\mathbf{x}_{t+1}|\boldsymbol{y}_t \sim f(\cdot|\boldsymbol{x}_t, \boldsymbol{u}_t)$ and $\mathbf{u}_{t+1}|\boldsymbol{x}_{t+1} \sim \pi(\cdot|\boldsymbol{x}_{t+1}; \boldsymbol{\theta}_i)$. The learning noise n_i accounts for the discrepancies between the instantaneous rewards r_i and the temporal difference $\mathsf{Q}_i(\boldsymbol{y}_t) - \gamma \mathsf{Q}_i(\boldsymbol{y}_{t+1})$. Note that even if the local reward variables have the same distributions, i.e., $\mathsf{r}_i(\boldsymbol{y}) \sim h(\cdot|\boldsymbol{y})$ $\forall i$ and $\forall \boldsymbol{y} \in \mathcal{Y}$, the learning noise terms $\{\mathsf{n}_i\}_{i=1}^N$ have potentially different characteristics in accordance with their respective priors on the Q function.

Introducing the vector notation

$$\mathbf{r}_i = [\mathsf{r}_i(\boldsymbol{y}_0), \mathsf{r}_i(\boldsymbol{y}_1), \cdots, \mathsf{r}_i(\boldsymbol{y}_{t_i-1})]^\top$$
$$\mathbf{Q}_i = [\mathsf{Q}_i(\boldsymbol{y}_0), \mathsf{Q}_i(\boldsymbol{y}_1), \cdots, \mathsf{Q}_i(\boldsymbol{y}_{t_i})]^\top$$
$$\mathbf{n}_i = [\mathsf{n}_i(\boldsymbol{y}_0, \boldsymbol{y}_1), \mathsf{n}_i(\boldsymbol{y}_1, \boldsymbol{y}_2), \cdots, \mathsf{n}_i(\boldsymbol{y}_{t_i-1}, \boldsymbol{y}_{t_i})]^\top$$

leads (3) to be written in the vectorial form

$$\mathbf{r}_i = \boldsymbol{H}_i \mathbf{Q}_i + \mathbf{n}_i \tag{4}$$

where we introduced the matrix

$$\boldsymbol{H}_i = \left[\boldsymbol{I}_{t_i}, \boldsymbol{0}_{t_i}\right] - \gamma \left[\boldsymbol{0}_{t_i}, \boldsymbol{I}_{t_i}\right] \tag{5}$$

and $\boldsymbol{I}_{t_i}$ is the identity matrix of size t_i while $\boldsymbol{0}_{t_i} = [0, \cdots, 0]^\top$ is the zero vector of dimension t_i. The learning noise is assumed Gaussian with zero mean and with a covariance matrix that obeys $\boldsymbol{\Sigma}_i = \mathbb{C}\mathrm{ov}\{\mathbf{n}_i, \mathbf{n}_i\} = \sigma_i^2 \boldsymbol{H}_i \boldsymbol{H}_i^\top$ under certain assumptions on the distribution of the discounted returns [55]. Denoting the locally observed trajectory data by $\mathcal{D}_i = \left((\boldsymbol{y}_t^{(i)}, r_t^{(i)}, \boldsymbol{y}_{t+1}^{(i)})\right)_{t=0}^{t_i}$ and the reward vector by $\boldsymbol{r}_i = [r_0^{(i)}, r_1^{(i)}, \cdots, r_{t_i-1}^{(i)}]^\top$, the posterior distribution of the Q function is the GP $\mathsf{Q}_i|\mathcal{D}_i \sim f_{\mathrm{GP}}(\hat{Q}_i, \hat{S}_i)$, with mean function and covariance kernel given by [47]

$$\hat{Q}_i(\boldsymbol{y}) = \mathbb{E}\{\mathsf{Q}_i(\boldsymbol{y})|\mathcal{D}_i\} = \boldsymbol{k}_i(\boldsymbol{y})^\top \boldsymbol{\alpha}_i \tag{6a}$$
$$\begin{aligned}\hat{S}_i(\boldsymbol{y}, \boldsymbol{y}') &= \mathbb{C}\mathrm{ov}\{\mathsf{Q}_i(\boldsymbol{y}), \mathsf{Q}_i(\boldsymbol{y}')|\mathcal{D}_i\} \\ &= \kappa_i(\boldsymbol{y}, \boldsymbol{y}') - \boldsymbol{k}_i(\boldsymbol{y})^\top \boldsymbol{C}_i \boldsymbol{k}_i(\boldsymbol{y}')\end{aligned} \tag{6b}$$

for any $\boldsymbol{y}, \boldsymbol{y}' \in \mathcal{Y}$ and where [47]

$$\boldsymbol{k}_i(\boldsymbol{y}) = [\kappa_i(\boldsymbol{y}_0^{(i)}, \boldsymbol{y}), \kappa_i(\boldsymbol{y}_1^{(i)}, \boldsymbol{y}), \ldots, \kappa_i(\boldsymbol{y}_{t_i}^{(i)}, \boldsymbol{y})]^\top \tag{7a}$$
$$\boldsymbol{K}_i = [\boldsymbol{k}_i(\boldsymbol{y}_0^{(i)}), \boldsymbol{k}_i(\boldsymbol{y}_1^{(i)}), \ldots, \boldsymbol{k}_i(\boldsymbol{y}_{t_i}^{(i)})] \tag{7b}$$
$$\boldsymbol{\alpha}_i = \boldsymbol{H}_i^\top \left(\boldsymbol{H}_i \boldsymbol{K}_i \boldsymbol{H}_i^\top + \boldsymbol{\Sigma}_i\right)^{-1} \boldsymbol{r}_i \tag{7c}$$
$$\boldsymbol{C}_i = \boldsymbol{H}_i^\top \left(\boldsymbol{H}_i \boldsymbol{K}_i \boldsymbol{H}_i^\top + \boldsymbol{\Sigma}_i\right)^{-1} \boldsymbol{H}_i\,. \tag{7d}$$

Let $\boldsymbol{s}_i(\boldsymbol{y})$ denote the score vector $\boldsymbol{s}_i(\boldsymbol{y}) = \nabla_{\boldsymbol{\theta}} \log \pi(\boldsymbol{u}|\boldsymbol{x}; \theta_i)$ and the corresponding Fisher information matrix defined as $\boldsymbol{F}_i = \int \boldsymbol{s}_i(\boldsymbol{y}) \boldsymbol{s}_i(\boldsymbol{y})^\top d_i^\pi(\boldsymbol{y}) \mathrm{d}\boldsymbol{y}$, where $d_i^\pi(\boldsymbol{y})$ is the discounted weighting function for the i-th agent (see (1) and [47, Sec.2] for more details). Furthermore, we assume that the GP kernel can be decomposed as [47]

$$\kappa_i(\boldsymbol{y}, \boldsymbol{y}') = \kappa_{i,\mathrm{x}}(\boldsymbol{x}, \boldsymbol{x}') + \kappa_{i,\mathrm{F}}(\boldsymbol{y}, \boldsymbol{y}') \tag{8a}$$
$$\kappa_{i,\mathrm{F}}(\boldsymbol{y}, \boldsymbol{y}') = \boldsymbol{s}_i(\boldsymbol{y})^\top \boldsymbol{F}_i^{-1} \boldsymbol{s}_i(\boldsymbol{y}')\,. \tag{8b}$$

Note that the form of the state-dependent kernel $\kappa_{i,\mathrm{X}}$ is specific to agent $v_i \in \mathcal{V}$, while all agents have the same form for the Fisher kernel $\kappa_{i,\mathrm{F}}$, i.e., the form in (8b). The following proposition gives the distribution of the local policy gradient when employing the posterior GP for the Q-function in the policy-gradient equation of (2).

Proposition 1. *A linear integral operator transforms the GP posterior $\mathrm{Q}_i|D_i$ into the Gaussian-distributed policy gradient $\hat{\mathbf{g}}_i|D_i;\boldsymbol{\theta}_i^{(k)} \sim f_{\mathrm{G}}(\hat{\boldsymbol{g}}_i, \hat{\boldsymbol{G}}_i)$. Under the additional decomposition of (8), the posterior gradient moments are given by [47]*

$$\hat{\boldsymbol{g}}_i = \boldsymbol{S}_i \boldsymbol{\alpha}_i \tag{9a}$$

$$\hat{\boldsymbol{G}}_i = \boldsymbol{F}_i - \boldsymbol{S}_i \boldsymbol{C}_i \boldsymbol{S}_i^{\top} \tag{9b}$$

where $\boldsymbol{S}_i = \left[\boldsymbol{s}_i(\boldsymbol{y}_0^{(i)}), \boldsymbol{s}_i(\boldsymbol{y}_1^{(i)}), \ldots, \boldsymbol{s}_i(\boldsymbol{y}_{t_i}^{(i)})\right]$.

A proof of Proposition 1 is found in Sections 7.2 and Appendix D of [47].

3.2. Fusion criterion and architecture

The objective of this section is to propose a method for merging the different local posteriors of the learning agents. In general, the agents implement potentially heterogeneous Bayesian learning algorithms for obtaining posterior density estimates $\hat{f}_i$ of the policy gradient vector $\hat{\mathbf{g}}_i|D_i;\boldsymbol{\theta}_i^{(k)}$. For example, both variants of BPG algorithms from [47] and the BAC algorithm from [46] can be used together in our proposed distributed fusion architecture. In general, the posterior densities for the policy gradient obtained by each agent are different due to different local conditions. Such conditions are given by: (i) the type of algorithm used, (ii) the specific implementation of the environment when a simulator is employed, and (iii) the prior parameters used by the learning algorithm.

Even when each agent employs the same type of algorithm (e.g., BAC of Sec. 3.1), the conditions at (iii) imply that a careful fusion of the entire information contained in the local posteriors is necessary as opposed to simply averaging their mean values. For example, the local kernel functions $\kappa_{i,\mathrm{X}}$ and consequently the characteristics of the learning noises n_i are different for each agent $v_i \in \mathcal{V}$. Mercer's theorem [49, Thm. 4.2] links a specific kernel $\kappa_{i,\mathrm{X}}$ to an inner product on a high (potentially infinite) dimensional Hilbert space $\mathcal{H}_i$, called feature space.[2] Thus, each agent is allowed to have a different feature space for a diversified representation of the Q function. Other agent-specific learning parameters are the number of training episodes and of data samples per episode, as well as the threshold value and dictionary size used for on-line sparsification.

A frequently-employed mechanism for merging probability densities is given by the KLD centroid of the set of local posteriors $\{\hat{f}_i\}_{i=1}^N$ (see [57–59]). More specifically, denoting via $D_{\mathrm{KL}}\{f,g\} \triangleq \int f(\boldsymbol{x}) \log\left(\frac{f(\boldsymbol{x})}{g(\boldsymbol{x})}\right) \mathrm{d}\boldsymbol{x}$ the KLD between two densities f and g, and letting $w_i \in (0,1)$ represent a confidence weight associated with agent v_i, where $\sum_{i=1}^N w_i = 1$, we propose to merge the local posteriors $\{\hat{f}_i\}_{i=1}^N$ via the following variational program

$$\begin{aligned} f_{\mathrm{F}} \triangleq \underset{f}{\operatorname{argmin}} \quad & \sum_{i=1}^{N} w_i \, D_{\mathrm{KL}}\{f, \hat{f}_i\} \\ \text{subject to} \quad & f(\boldsymbol{g}) \geqslant 0 \ \forall \boldsymbol{g} \in \mathbb{R}^d \\ & \int f(\boldsymbol{g}) \, \mathrm{d}\boldsymbol{g} = 1 \,. \end{aligned} \tag{10}$$

Proposition 2. *The solution to the KLD minimization of (10) is given by the fused distribution f_{F} shown to be*

$$f_{\mathrm{F}}(\boldsymbol{g}) = Z^{-1} \prod_{i=1}^{N} \left[\hat{f}_i(\boldsymbol{g})\right]^{w_i} \tag{11}$$

where $Z \triangleq \int \prod_{i=1}^N \left[\hat{f}_i(\boldsymbol{g})\right]^{w_i} \mathrm{d}\boldsymbol{g} \in (0,1]$ is a normalization constant.

A proof of Proposition 2 is found in [57]. In general, the density $f_{\mathrm{F}}(\cdot)$ does not admit an analytic expression due to the normalization constant Z. The weights $\{w_i\}_{i=1}^N$ encode the confidence given to each agent during the learning process, with equal confidence given by $w_i = 1/N \ \forall \ i$.

Multiple criteria for fusing probability densities exist in the literature, see [58] for a comprehensive study. The fusion criterion in (10) employs the exclusive form of the KLD $D_{\mathrm{KL}}\{f, \hat{f}_i\}$ (or I-projection [60, Ch. 8.5]), which favors densities f that provide a good match to the most probable mode of the target density. This is in contrast to the inclusive form of the KLD, which favors densities f that match the main mass of the target density but also assign significant values to the entire support of the target density, at the expense of higher variance. The reader is referred to [60, Ch. 8.5] for a detailed discussion on their different properties. The symmetrized form of the KLD is also a possible fusion criterion. Note that when the local densities are Gaussian (as in the case

[2] In other words, there exists a mapping $\phi_i : \mathcal{X} \to \mathcal{H}_i$ such that $\kappa_{i,\mathrm{X}}(\boldsymbol{x}, \boldsymbol{x}') = \langle \phi_i(\boldsymbol{x}), \phi_i(\boldsymbol{x}') \rangle_{\mathcal{H}_i}$ for all $v_i \in \mathcal{V}$ (see [56, Ch.2] for more details).

of Bayesian RL), the fusion criterion in (10) leads to another Gaussian density, which is not the case for criteria using either the inclusive or the symmetrized forms of the KLD. This result is formalized in the following proposition.

Proposition 3. *If the local probability densities are Gaussian, i.e.,* $\hat{f}_i(\boldsymbol{g}) = f_{\mathrm{G}}(\boldsymbol{g}; \hat{\boldsymbol{m}}_i, \hat{\boldsymbol{P}}_i)\ \forall i$*, the fused density is also Gaussian* $f_{\mathrm{F}}(\boldsymbol{g}) = f_{\mathrm{G}}(\boldsymbol{g}; \boldsymbol{m}_{\mathrm{F}}, \boldsymbol{P}_{\mathrm{F}})$ *with parameters*

$$\boldsymbol{P}_{\mathrm{F}}^{-1} = \sum_{i=1}^{N} w_i \hat{\boldsymbol{P}}_i^{-1}, \tag{12a}$$

$$\boldsymbol{m}_{\mathrm{F}} = \boldsymbol{P}_{\mathrm{F}} \sum_{i=1}^{N} w_i \hat{\boldsymbol{P}}_i^{-1} \hat{\boldsymbol{m}}_i\,. \tag{12b}$$

A proof of Proposition 3 is found in [61, Sec. 3.1].

The fusion rule of (12) is referred to as covariance intersection (CI) in the robotics [62–65] and target-tracking [66] communities. Additionally, in [67] an information geometric characterization is given for the fused Gaussian density of (12), in the case of $N = 2$ and with optimal weights, as the unique intersection of the exponential geodesic curve joining the two densities and its dual hyperplane. Considering the densities $\{\hat{f}_i\}_{i=1}^{N}$ as the marginal distributions of a set of local gradient estimators $\{\hat{\mathbf{g}}_i\}_{i=1}^{N}$, the CI rule is shown in [68] to provide a consistent estimator when the cross-correlations between the local estimators are unknown. In the terminology of [68], consistency is attained when the covariance of the fused estimator is always greater than or equal to the covariance of the optimally fused estimator that has access to the correlation information.

Algorithm 1 FBLA-PMG and FBLA-PMGC (executed synchronously at each agent $v_i \in \mathcal{V}$).

1: **Initialization** $\boldsymbol{\theta}_i^{(0)}$, $\boldsymbol{\Theta}_i^{(0)}$, t_i, κ_i, σ_i^2, and learning rate β.
2: **for** $k = 0, 1, \cdots, K$ **do**
3: Sample trajectory data $D_i \triangleq \left(\boldsymbol{x}_t^{(i)}, \boldsymbol{u}_t^{(i)}, r_t^{(i)}, \boldsymbol{x}_{t+1}^{(i)}\right)_{t=0}^{t_i}$
4: Estimate posterior $\mathrm{Q}_i | D_i \sim f_{\mathrm{GP}}(\hat{Q}_i, \hat{S}_i)$ via (6) or other BPG method
5: Estimate local gradient posterior $\hat{\mathbf{g}}_i | D_i; \boldsymbol{\theta}_i^{(k)} \sim f_{\mathrm{G}}\left(\hat{\boldsymbol{g}}_i^{(k)}, \hat{\boldsymbol{G}}_i^{(k)}\right)$ (Proposition 1)
6: Local precision matrix initialization $\boldsymbol{\Lambda}_i^{(0)} \leftarrow w_i N \left[\hat{\boldsymbol{G}}_i^{(k)}\right]^{-1}$
7: Set weights $\omega_{ij}^{(k)} \leftarrow \left(1 + \#\mathcal{N}_{i,k}^{\mathrm{in}}\right)^{-1}$ for $j \in \{i\} \cup \mathcal{N}_{i,k}^{\mathrm{in}}$, while $\omega_{ij}^{(k)} \leftarrow 0$ for $j \notin \{i\} \cup \mathcal{N}_{i,k}^{\mathrm{in}}$
8: **for** $l = 0, 1, \cdots, L-1$ **do**
9: Broadcast $\boldsymbol{\Lambda}_i^{(l)}$ to the out-neighbors $\mathcal{N}_{i,k}^{\mathrm{out}}$
10: Receive $\left\{\boldsymbol{\Lambda}_j^{(l)}\right\}_{j \in \mathcal{N}_{i,k}^{\mathrm{in}}}$ from all in-neighbors $\mathcal{N}_{i,k}^{\mathrm{in}}$
11: Precision matrix gossip $\boldsymbol{\Lambda}_i^{(l+1)} \leftarrow \omega_{ii}^{(k)} \boldsymbol{\Lambda}_i^{(l)} + \sum_{j \in \mathcal{N}_{i,k}^{\mathrm{in}}} \omega_{ij}^{(k)} \boldsymbol{\Lambda}_j^{(l)}$
12: **end for**
13: For FBLA-PMG: Set $\tilde{\boldsymbol{\Lambda}}_i \leftarrow \boldsymbol{\Lambda}_i^{(L)}\ \forall$ agents $v_i \in \mathcal{V}$.
14: For FBLA-PMGC: max-consensus for precision matrices $\left\{\boldsymbol{\Lambda}_i^{(L)}\right\}_{i=1}^{N}$ and obtain $\left\{\tilde{\boldsymbol{\Lambda}}_i\right\}_{i=1}^{N}$, with $\tilde{\boldsymbol{\Lambda}}_i = \tilde{\boldsymbol{\Lambda}}_j\ \forall$ agents $v_i, v_j \in \mathcal{V}$ and $\tilde{\boldsymbol{\Lambda}}_i \leftarrow \max\left(\left\{\boldsymbol{\Lambda}_j^{(L)}\right\}_{j=1}^{N}\right)$.
15: Policy gradient update $\tilde{\boldsymbol{\theta}}_i \leftarrow \boldsymbol{\theta}_i^{(k)} + \beta \tilde{\boldsymbol{\Lambda}}_i^{-1} \boldsymbol{\Lambda}_i^{(0)} \hat{\boldsymbol{g}}_i^{(k)}$
16: Broadcast $\tilde{\boldsymbol{\theta}}_i$ to the out-neighbors $\mathcal{N}_{i,k}^{\mathrm{out}}$
17: Receive $\left\{\tilde{\boldsymbol{\theta}}_j\right\}_{j \in \mathcal{N}_{i,k}^{\mathrm{in}}}$ from all in-neighbors $\mathcal{N}_{i,k}^{\mathrm{in}}$
18: Policy gradient gossip $\boldsymbol{\theta}_i^{(k+1)} \leftarrow \omega_{ii}^{(k)} \tilde{\boldsymbol{\theta}}_i + \sum_{j \in \mathcal{N}_{i,k}^{\mathrm{in}}} \omega_{ij}^{(k)} \tilde{\boldsymbol{\theta}}_j$
19: Fused policy covariance $\boldsymbol{\Theta}_i^{(k+1)} \leftarrow \tilde{\boldsymbol{\Lambda}}_i^{-1}$
20: **end for**
21: **Return** fused policy mean $\boldsymbol{\theta}_i^{(K)}$ and covariance $\boldsymbol{\Theta}_i^{(K)}$.

The fusion of posterior densities of policy gradients, as given in Proposition 3, can be achieved in a decentralized manner through various methods. All agents $v_i \in \mathcal{V}$ operate synchronously and iterate between: collection of local data by interacting with the environment, local estimation of the policy gradient posterior, and inter-agent communications. In addition, different architectures arise since the rule of Proposition 3 involves the fusion of both mean vectors and covariance matrices. In Algorithm 1, we propose two FBLA variants: FBLA-PMG and FBLA-PMGC.

In both FBLA variants, gossip is first performed across the agents to fuse their local precision matrices (12a) at lines 9-11, followed by local updates of the policy gradient at line 15. Finally, a second gossip operation is carried out for the updated policy vectors at lines 18. In FBLA with PMC, an additional max-consensus is carried out for the precision matrices at line 14. The max-consensus operation additionally ensures that all precision matrices are identical $\tilde{\boldsymbol{\Lambda}}_i = \tilde{\boldsymbol{\Lambda}}_j$ for all agents $v_i, v_j \in \mathcal{V}$. Note that the maximum operator across a set of matrices yields the matrix that has the smallest spectral radius for its inverse. Other choices for this operator are possible. However, the convention adopted here ensures that the resulting matrix is a valid precision matrix for which the result in Lemma 3.2 (ii) holds.

3.3. Theoretical guarantees

In the following, we provide ϵ-ball guarantees for the distances between the different policy parameters estimated by the agents. The following assumptions will be necessary for these guarantees.

A.1) The graph sequence $\mathcal{G}_k$ is strongly connected at all times k and induces a sequence of row-stochastic mixing matrices $\mathcal{M} = \left(\mathbf{\Omega}^{(k)} : \ k = 0, 1, \dots, K\right)$, where each matrix is ergodic and any finite product of matrices from $\mathcal{M}$ (including repetitions) is again ergodic. Furthermore, the joint spectral radius of a collection $\mathcal{M}$ of matrices will be denoted with $\rho(\mathcal{M})$ (see [69] for more details).

A.2) The local gradient covariance matrices are positive definite and bounded away from zero at all times in the following manner $\varphi_i = \sup\left\{ \left\| \left[\hat{\boldsymbol{G}}_i^{(k)}\right]^{-1} \right\|_{\mathrm{F}} : \ k = 1, 2, \dots \right\} < \infty \forall i \in \mathbb{N}_1^N$.

A.3) The local gradient vectors are bounded in norm at all times as $\vartheta_i = \sup\left\{ \left\| \left[\hat{\boldsymbol{G}}_i^{(k)}\right]^{-1} \hat{\boldsymbol{g}}_i^{(k)} \right\|_2 : \ k = 1, 2, \dots \right\} < \infty \forall i \in \mathbb{N}_1^N$.

A.4) The local gradient covariance matrices have bounded spectral radii at all times as $\boldsymbol{\Phi}_i = \sup\left\{ \rho\left(\hat{\boldsymbol{G}}_i^{(k)}\right) : \ k = 1, 2, \dots \right\} < \infty \forall i \in \mathbb{N}_1^N$.

The preceding assumptions are in line with those assumed in the decentralized estimation literature. More specifically, assumptions on the communication graph, such as Assumption A.1), are common to many gossip-based architectures, e.g., [31,69,70], whereas the condition of bounded local vectors of Assumption A.3) is typical for ϵ-ball guarantees, e.g., [42]. Here, similar conditions for the local covariance matrices are also required. This takes the form of Assumption A.4), while Assumption A.2) additionally imposes the non-singularity of these matrices, as required by the fusion criterion.

The following lemmas provide various bounds on the distances between the local estimates of the architectures in Algorithm 1. The proofs are found in the appendices.

Lemma 3.1. *Under assumptions A.1)-A.2), at any time step k, and after L gossip steps the precision matrices $\mathbf{\Lambda}_i^{(L)}$, for all agents $v_i \in \mathcal{V}$, are within a constant distance from their average value, i.e.,*

$$\left\| \mathbf{\Lambda}_i^{(L)} - \bar{\mathbf{\Lambda}} \right\|_{\mathrm{F}} \leqslant CN\varphi q^L \tag{13}$$

where $\bar{\mathbf{\Lambda}} \triangleq \frac{1}{N}\sum_{i=1}^N \mathbf{\Lambda}_i^{(L)}$, $\varphi = \max\left(\{\sqrt{w_i}\varphi_i\}_{i=1}^N\right)$, while $q > \rho(\mathcal{M})$, and C is a constant.

Lemma 3.2. *The following provide bounds for the spectral radii of the matrices $\left\{\left[\mathbf{\Lambda}_i^{(L)}\right]^{-1}\right\}_i$ and those of the matrices $\left\{\tilde{\mathbf{\Lambda}}_i^{-1}\right\}_i$.*

(i) *Assuming A.4) and after L iterations, the spectral radius of the covariance matrix $\left[\mathbf{\Lambda}_i^{(L)}\right]^{-1}$ is upper bounded by the maximum of the radii of the local precision matrices, i.e., $\rho\left(\left[\mathbf{\Lambda}_i^{(L)}\right]^{-1}\right) \leqslant N^{-1}\varrho$ where $\varrho \triangleq \max\left(\{w_i^{-1}\boldsymbol{\Phi}_i\}_{i=1}^N\right)$ $\forall k$ and for any node $v_i \in \mathcal{V}$.*

(ii) *For FBLA with PMC, assuming A.4) and after diam$(\mathcal{G}_k)$ iterations for max consensus for matrices, the maximum of the spectral radius of $\tilde{\mathbf{\Lambda}}_i^{-1}$ is also bounded as $\rho\left(\tilde{\mathbf{\Lambda}}_i^{-1}\right) \leqslant N^{-1}\varrho$ for any node $v_i \in \mathcal{V}$.*

Theorem 3.3. ***(ϵ-ball for FBLA-PMG and FBLA-PMGC)*** *Under assumptions A.1)-A.4), at any time step k, and for the FBLA-PMG, the fused mean vector and covariance matrix of each agent $v_i \in \mathcal{V}$ are contained within an ϵ-ball from their respective average values across all agents, i.e.,*

$$\left\| \boldsymbol{\theta}_i^{(k)} - \bar{\boldsymbol{\theta}}^{(k)} \right\|_2 \leqslant \beta C \frac{\varrho\vartheta}{1-q} \tag{14}$$

$$\left\| \mathbf{\Theta}_i^{(k)} - \bar{\mathbf{\Theta}}^{(k)} \right\|_2 \leqslant 2\varphi\varrho^2 C N^{-1} q^L \tag{15}$$

where $\vartheta = \max\left(\{\sqrt{w_i}\vartheta_i\}_{i=1}^N\right)$ is a constant, the average mean vector is defined as $\bar{\boldsymbol{\theta}}^{(k)} \triangleq \frac{1}{N}\sum_{i=1}^N \boldsymbol{\theta}_i^{(k)}$, and the average covariance matrix is defined as $\bar{\mathbf{\Theta}}^{(k)} \triangleq \frac{1}{N}\sum_{i=1}^N \mathbf{\Theta}_i^{(k)}$.

The ϵ-ball result for the mean vectors in (14) applies to the FBLA-PMGC as well, whereas covariance matrices are identical across all agents by construction.

4. Numerical experimentation

We consider a sensor activation problem similar to that presented in [48]. A grid of $S = 49$ equidistantly-spaced sensors is employed to track a single appearing and disappearing object. The upper panel of Fig. 1 showcases the sensor grid and one instance of the object trajectory. Note the times of birth and death of the object as shown in the lower panel of Fig. 1. The object is modeled via a Bernoulli random finite set [71]. The object evolves in a 2D space with a state vector given by $\boldsymbol{x} = [p_x, p_y, \dot{p}_x, \dot{p}_y]^\top$, where

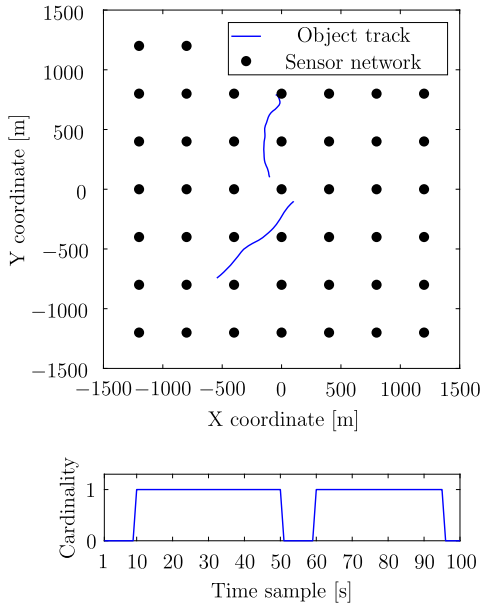


Fig. 1. Sensor grid and one example of the object trajectory (upper panel). Cardinality of the object set, showcasing the times of appearance, disappearance, and reappearance of the object (lower panel).

(p_x, p_y) and $(\dot{p}_x, \dot{p}_y)$ represent its coordinates and velocities along the two axes. The object dynamics are governed by a white-noise acceleration model [72, Ch. 6.3.2].

The sensors are providing noisy distance measurements between their positions and the object position. However, the sensors are further affected by miss-detections and false-alarm measurements–also called clutter. A sensor $s \in \mathbb{N}_1^S$ detects the object with probability $P_s^{\mathrm{D}} \in (0, 1)$ and generates an object-originated measurement. Independently of the object, at each time step and for each sensor s, a number of clutter measurements are generated according to a Poisson point process. The number of clutter measurements (or points) is Poisson distributed with rate λ_s^{FA}. Conditionally on the number of clutter points, their measurement values are uniformly distributed in the measurement (i.e., range) space of the sensor. Given a set of sensor measurements, a corresponding filtering algorithm is employed to estimate a posterior probability of existence and the posterior spatial density for the object as in [73]. In the scenario considered here, we consider a limited number (e.g., 10) of high-quality sensors with parameters $P_s^{\mathrm{D}} = 0.9$ and measurement noise standard deviation of 10 m; whereas the others have a probability of detection of $P_s^{\mathrm{D}} = 0.5$ and measurement noise standard deviation of 150 m. The false-alarm rate is identical across all sensors with a value $\lambda_s^{\mathrm{FA}} = 1$ per time sample.

4.1. Sensor activation POMDP

This section describes the problem of determining an optimal sensor activation policy, i.e., a rule for determining which sensors to activate based on the current estimates of the object state. This problem is modeled as a partially observed Markov decision process (POMDP) (see [53] for more details) since the true object state is not directly observed by the sensors. Thus, an information-space (or belief-space) reformulation is employed, in order to transform the POMDP into an MDP on an appropriately defined information state vector $\boldsymbol{i}$. Formally, the notation for the state vector $\boldsymbol{x}$ is replaced with $\boldsymbol{i}$ throughout this section.

The information-space MDP that incorporates the available information at time sample t for the decision problem is composed of:

- $\mathcal{I} \subset \mathbb{R}^{2n+1}$ is the set of information state vectors $\boldsymbol{i}_t$ which are constructed as $\boldsymbol{i}_t = \left[r_{t|t-1}, \boldsymbol{m}_{t|t-1}^{\top}, \mathrm{diag}\left(\boldsymbol{P}_{t|t-1}\right)^{\top}\right]^{\top}$ from the predicted probability of existence $r_{t|t-1}$, the predicted mean $\boldsymbol{m}_{t|t-1} \in \mathbb{R}^n$ and diagonal of the predicted covariance $\boldsymbol{P}_{t|t-1} \in \mathbb{R}^{n \times n}$ as obtained by the Bernoulli filter [73],
- $\mathcal{U} \subseteq \mathbb{R}^S$ is the set of admissible actions $\boldsymbol{u}_t = [u_{1,t}, u_{2,t}, \ldots, u_{S,t}]^{\top}$, i.e., vectors of unnormalized log-probabilities $u_{s,t} \in \mathbb{R}\ \forall s$,
- $f(\cdot|\boldsymbol{i}_{t-1}, \boldsymbol{u}_{t-1})$ is the transition kernel of the information state $\mathbf{i}_t$ given the previous state and action,
- $h(\cdot|\boldsymbol{i}_t, \boldsymbol{u}_t)$ is the distribution of the instantaneous reward, here the Cauchy-Schwartz information gain for Bernoulli tacking (see [9]),
- $\pi(\cdot|\boldsymbol{i}_t; \boldsymbol{\theta})$ is the conditional probability density of the action $\mathbf{u}_t$ given the current information state $\boldsymbol{i}_t$ and is parameterized via $\boldsymbol{\theta} \in \mathbb{R}^d$,
- $\gamma \in [0, 1)$ is the reward discount factor (here $\gamma = 0.99$).

Note that the information-space formulation for Bernoulli tracking, i.e., the construction of the information state vector $\boldsymbol{i}_t$, is borrowed from [48]. However in [48], no multi-agent learning is considered and the control policy is given by a specific multi-Beta

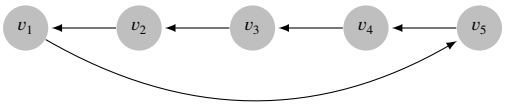


Fig. 2. Communication graph for the $N = 5$ learning agents.

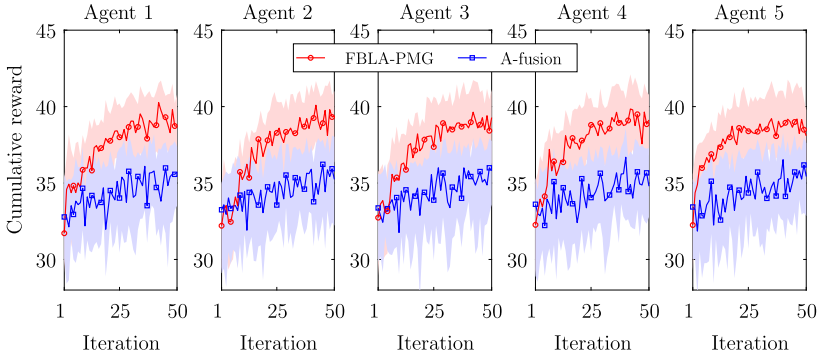


Fig. 3. Cumulative reward as a function of learning iteration across all agents. Both median and confidence interval (first and third quartiles) are reported over 60 independent simulations for both the proposed FBLA-PMG and the A-fusion architecture.

distribution form which sensor activation probabilities are sampled. Furthermore in [48], imposing a constraint on the average number of sensors that can be active at any given time involves an additional optimization step, which is avoided in the policy formulation presented next.

In contrast to [48], here, the control policy is given by the normal distribution

$$\pi(\boldsymbol{u}|\boldsymbol{i};\boldsymbol{\theta}) = \prod_{s=1}^{S} f_{\mathrm{G}}(u_s; \varphi_s(\boldsymbol{i};\theta_s), \sigma_\pi^2) \tag{16}$$

where $\boldsymbol{\theta} \triangleq [\theta_1, \theta_2, \ldots, \theta_S]^\top \in \mathbb{R}^S$ is the vector of policy parameters (for the case here $d = S$). Equivalently stated, given $\boldsymbol{i}$ and $\boldsymbol{\theta}$, the action (or control) of each sensor s is a Gaussian random variable $\mathrm{u}_s \in \mathbb{R}$ with mean $\varphi_s(\boldsymbol{i};\theta_s)$ and variance $\sigma_\pi^2 = 1$. An instance u_s of u_s represents an unnormalized log-probability and determines the probability p_s of choosing (i.e., activating) sensor $s \in \mathbb{N}_1^S$ at a given time. More specifically, the vector of sensor activation probabilities $\boldsymbol{p} = [p_1, p_2, \ldots, p_S]^\top$ is obtained from the vector of unnormalized log-probabilities $\boldsymbol{u} = [u_1, u_2, \ldots, u_S]^\top$ via the softmax function, i.e., $p_s = \frac{\exp(u_s)}{\sum_{j=1}^{S} \exp(u_j)}$ $\forall s$. Note that while $u_s \in \mathbb{R}$ $\forall s$, the activation probabilities lie in the unit simplex, i.e., $p_s \in (0, 1]$ and $\sum_{j=1}^{S} p_j = 1$. For a given control vector $\boldsymbol{u}$ (and implicitly $\boldsymbol{p}$), the collection of active sensors is determined by sampling M times (here $M = 3$) without replacement from a categorical distribution over the indices $1, 2, \cdots, S$ and with probabilities given by $\boldsymbol{p}$.

Furthermore, the mean unnormalized log-probability for sensor s is

$$\varphi_s(\boldsymbol{i};\theta_s) = \frac{\theta_s \phi_s(\boldsymbol{i})}{1 + \phi_s(\boldsymbol{i})} \tag{17}$$

where $\phi_s(\boldsymbol{i})$ quantifies the ability of the sensor to observe a Bernoulli object characterized by the information vector $\boldsymbol{i}$ (for an exact expression see eq. (7) from [48]).

4.2. Multi-agent learning setup

A number of $N = 5$ learning agents are employed to train the sensor-activation policy of (16). The agents are arranged in a ring communication network as depicted in Fig. 2. The first three agents use an online-version of the BAC learning algorithm of [47], with the following kernel covariance functions κ_x: exponential for agent v_1, rational quadratic of order v_2 for agent 2, and Matérn for agent v_3 (see [49, Ch. 4.2.2]). Agents v_4 and v_5 employ the vector-model BPG algorithm of [47]. The learning noise standard deviations are set to $\sigma_1 = 1$ and $\sigma_i = 20$ for $i \in \mathbb{N}_2^5$. Per learning iteration, all agents simulate the Bernoulli target tracking problem in 10 independent episodes of 100 time samples each. Subsequently, several fusion architectures are employed to merge the local policy gradient posterior estimates of these heterogeneous learning agents.

We compare the proposed FBLA-PMG, FBLA-PMGC, and an average-fusion (A-fusion) architecture in which the mean and covariance matrices of the policy gradients are averaged via gossiping. The same number $L = 5$ of gossip iterations is employed for all algorithms. The cumulative reward achieved at each agent is showcased in Fig. 3 for the FBLA-PMG and the A-fusion architecture. The second quartile (median) and inter-quartile (first-third quartile) interval is reported over 60 independent simulations. Note the faster learning rate with a smaller inter-quartile range of the FBLA-PMG as compared to the A-fusion architecture. Similar results

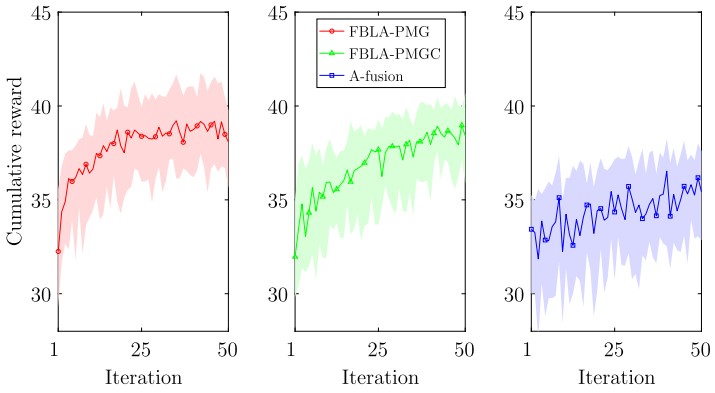


Fig. 4. Comparison of cumulative reward achieved at agent 5 for the three architectures: FBLA-PMG, FBLA-PMGC, and A-fusion. Both median and confidence interval (first and third quartiles) are reported over 60 independent simulations.

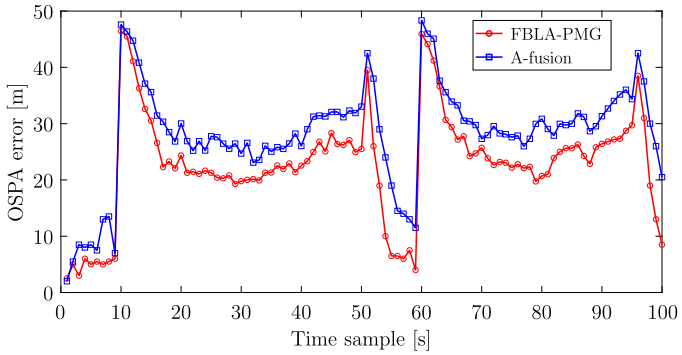


Fig. 5. Mean OSPA error achieved by Bernoulli filtering with the sensor-selection parameters learned via FBLA-PMG and the reference A-fusion architecture. Error curves are averaged over 100 independent simulations.

were observed for the FBLA-PMGC. This can be more easily seen in Fig. 4, where the same cumulative reward is reported for all three architectures: FBLA-PMG, FBLA-PMGC, and the A-fusion architecture. The addition of the PMC operation only seems to slightly reduce the variability of FBLA learning, i.e., improved stability, but the overall effect is marginal on the cumulative reward. However, the improvement over the A-fusion architecture is significant. This is attributed to the higher-order information that is employed in the fusion process of the FBLA. As it can be seen from Proposition 3, the estimated covariance of the local policy gradient vector acts as a weight on the mean vector estimate of that same policy gradient. This ensures a robust fusion procedure in which more noisy policy parameters are weighted accordingly.

We assess the tracking performance achieved when using the learned policy parameters of the FBLA-PMG and the A-fusion architecture in Fig. 5. More specifically, the multi-sensor Bernoulli filter of [73] is employed in conjunction with the sensor activation policy of Sec. 4.1. The parameters $\boldsymbol{\theta}$ of the policy are set to either the values learned via FBLA-PMG or the A-fusion architecture. The multi-sensor Bernoulli filter produces an estimated object set–the set is either empty or contains a single vector estimate of the object state vector. In order to quantify errors in estimating a set, we employ the optimum subpattern assignment (OSPA) distance [74] which incorporates errors in both the cardinality (i.e., estimated number of objects) and the elements in the set (i.e., estimated state of the objects). The resulting OSPA errors for the two fusion architectures are showcased in Fig. 5 as a function of the time sample. The OSPA error curves are averaged over 100 independent simulations. Observe the lower OSPA error obtained when using the policy parameters learned by the proposed FBLA-PMG. Similar error results are obtainable when using the policy parameters learned via the FBLA-PMGC.

The theoretical ϵ-ball results of Sec. 3.3 are assessed numerically in Fig. 6. We assess both the deviation of mean vectors and of covariance matrices from their respective average values computed across the $N = 5$ agents. The vector L^2 norm and matrix L^2 norm are used to measure the respective distances. From Fig. 6 we observe the stability (i.e., no increasing trend) of the learned policy mean vectors across all agents and for both variants of FBLA. Furthermore, in general a decreasing trend is observed for the learned covariance matrices. The exception is agent v_5 with FBLA without PMC learning, in which case the fluctuations are much smaller than those recorded at the other agents. These numerical results support the bounding results of Sec. 3.3. Comparing the two variants of FBLA, the PMC operation is shown to numerically reduce the distances between the learned parameters across the different agents.

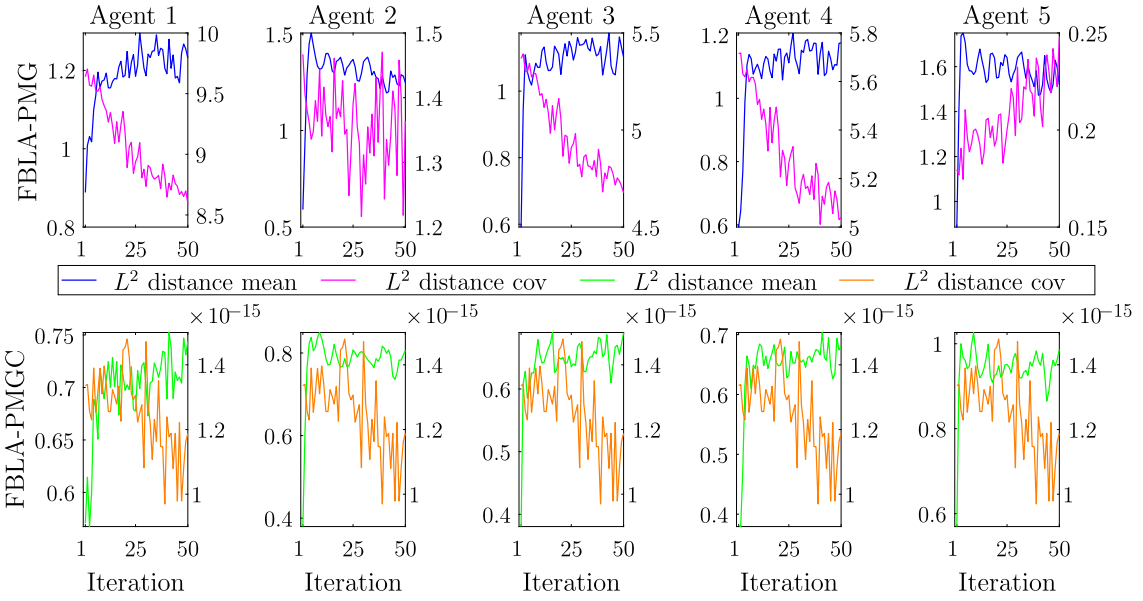


Fig. 6. ϵ-ball results for FBLA-PMG (upper row) and FBLA-PMGC (lower row). The panel showcase the distances in L^2 norm from the policy parameters at some agent and their corresponding average value for each learning iteration. Each panel has a double y-axis plot with the corresponding L^2-norm distance for the mean vector on the left axis and L^2-norm distance for the covariance matrix on the right axis.

5. Conclusion

In this work we introduced two decentralized fusion architectures of BPG learners. The proposed architectures robustly fuse the entire posterior information contained in the local posterior estimates and thus incorporate uncertainty on the locally-learned parameters. The fusion criterion finds the barycenter of all local posterior densities by minimizing a weighted KLD. This leads to a fusion procedure that is capable of fusing heterogeneous learners, i.e., learners that implement the same policy but may employ different procedures to obtain an estimate of the policy gradient. Examples of such learners are the BPG and BAC with various critic structures. Furthermore, ϵ-ball theoretical guarantees are given for the decentralized fusion architectures, that bound the differences of the first and second order moments of the locally fused policies across the different agents.

Declaration of competing interest

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

Data availability

No data was used for the research described in the article.

Acknowledgement

The fundamental research described in this paper was supported, in part, by the MIT-IBM Watson AI Lab Project Grant Number RPA53, by the National Science Foundation under Grant CNS-2148251, and by federal agency and industry partners in the RINGS program.

Appendix A. Proof of Lemma 3.1

By use of the vectorization operator vect$(\cdot)$, we define the column vectors $\boldsymbol{t}_i^{(l)} = \text{vect}(\boldsymbol{\Lambda}_i^{(l)})$ for any inner-iteration index $l \in \mathbb{N}_0^L$ and agent $v_i \in \mathcal{V}$. The corresponding concatenated matrices are defined as $\boldsymbol{T}^{(l)} = [\boldsymbol{t}_1^{(l)}, \boldsymbol{t}_2^{(l)}, \cdots, \boldsymbol{t}_N^{(l)}]^\top \ \forall l \in \mathbb{N}_0^L$.

Also let $\boldsymbol{\Omega}^{(k)}$ be the mixing matrix formed of the gossip weights $[\boldsymbol{\Omega}^{(k)}]_{ij} = \omega_{ij}$. After L gossiping steps, the precision matrices become $\boldsymbol{T}^{(L)} = \left[\boldsymbol{\Omega}^{(k)}\right]^L \boldsymbol{T}^{(0)}$. The mixing matrix $\boldsymbol{\Omega}^{(k)}$ is row stochastic (Ass. A.1)), thus its largest eigenvalue is one with corresponding eigenvector in the span of $\{\mathbf{1}_N\}$. Similar to [42,69], let the matrix $\boldsymbol{Q} \in \mathbb{R}^{(N-1)\times N}$ define an orthogonal projection onto the space orthogonal to the span of $\{\mathbf{1}_N\}$. Each mixing matrix $\boldsymbol{\Omega}^{(k)}$ has a corresponding and unique matrix $\boldsymbol{\Psi}^{(k)} \in \mathbb{R}^{(N-1)\times(N-1)}$ such that $\boldsymbol{Q}\boldsymbol{\Omega}^{(k)} = \boldsymbol{\Psi}^{(k)}\boldsymbol{Q}$. Let $\mathcal{M}'$ be the set of all matrices $\boldsymbol{\Psi}^{(k)}$. Furthermore, as noted in [69], the projection matrix $\boldsymbol{Q}$ has the following

properties: $\boldsymbol{Q}\mathbf{1}_N = \mathbf{0}_{N-1}$ and $\|\boldsymbol{Q}\boldsymbol{x}\|_2 = \|\boldsymbol{x}\|_2$ when $\boldsymbol{x}^\top \mathbf{1}_N = 0$. Let $\bar{\boldsymbol{t}}^{(L)} \triangleq \frac{1}{N}\sum_{i=1}^N \boldsymbol{t}_i^{(L)}$ and $\bar{\boldsymbol{T}}^{(L)} \triangleq \big(\bar{\boldsymbol{t}}^{(L)}\big)^\top \otimes \mathbf{1}_N$. Note that the projection $\boldsymbol{Q}$ eliminates one multiplicity of the unit eigenvalue from the spectrum of the mixing matrix $\boldsymbol{\Omega}^{(k)}$. Since $\big(\boldsymbol{T}^{(L)} - \bar{\boldsymbol{T}}^{(L)}\big)^\top \mathbf{1}_N = \mathbf{0}_{d^2}$, we have

$$\begin{aligned}\Big\|\boldsymbol{T}^{(L)} - \bar{\boldsymbol{T}}^{(L)}\Big\|_2 &= \Big\|\boldsymbol{Q}\boldsymbol{T}^{(L)} - \boldsymbol{Q}\bar{\boldsymbol{T}}^{(L)}\Big\|_2 \\ &= \Big\|\big[\boldsymbol{\Psi}^{(k)}\big]^L \boldsymbol{T}^{(0)}\Big\|_2 \\ &\leqslant C q^L \Big\|\boldsymbol{T}^{(0)}\Big\|_2\end{aligned}$$

where $q > \rho(\mathcal{M}')$ and $C \in \Theta(1)$ potentially depends on q but not on L [69]. The ergodicity assumption (Ass. A.1)) [70] ensures that $q < 1$. From the facts that $\Big\|\big[\hat{\boldsymbol{G}}_i^{(k)}\big]^{-1}\Big\|_{\mathrm{F}} \leqslant \varphi_i$ and $\Big\|\boldsymbol{T}^{(0)}\Big\|_2^2 \leqslant \varphi^2 N^2$ with $\varphi = \max\big(\{\sqrt{w_i}\varphi_i\}_{i=1}^N\big)$, we obtain the bounds

$$\Big\|\boldsymbol{t}_i^{(L)} - \bar{\boldsymbol{t}}^{(L)}\Big\|_2 \leqslant \Big\|\boldsymbol{T}^{(L)} - \bar{\boldsymbol{T}}^{(L)}\Big\|_2 \leqslant CN\varphi q^L$$

for all agents $v_i \in \mathcal{V}$. The first inequality follows since the L^2 norm of any row of a matrix is upper bounded by the induced matrix norm. This proves the claim in (13).

Appendix B. Proof of Lemma 3.2

Claim (i). Let the time index k be fixed. Note that Assumption A.2) guarantees the positive definiteness of the matrices $\big\{\hat{\boldsymbol{G}}_i^{(k)}\big\}_{i=1}^N$. It follows that the matrices $\big\{\boldsymbol{\Lambda}_i^{(l)}\big\}_{i=1}^N$ are also positive definite for all iterations $l = 0, 1, \cdots, L$. Subsequently, the matrices $\big\{\boldsymbol{\Theta}_i^{(k+1)}\big\}_{i=1}^N$ are also positive definite. The largest eigenvalue of the symmetric and real matrix $\boldsymbol{\Lambda}_i^{-(l)} \triangleq \big[\boldsymbol{\Lambda}_i^{(l)}\big]^{-1}$, with $l \in \mathbb{N}_1^L$, according to the Courant–Fischer theorem [75, Ch. III.1], is given by

$$\begin{aligned}\rho(\boldsymbol{\Lambda}_i^{-(l)}) &= \max_{\boldsymbol{x}\neq\mathbf{0}} \frac{\boldsymbol{x}^\top\big[\boldsymbol{\Lambda}_i^{(l)}\big]^{-1}\boldsymbol{x}}{\boldsymbol{x}^\top\boldsymbol{x}} \\ &\leqslant \max_{\boldsymbol{x}\neq\mathbf{0}} \sum_{j=1}^N \omega_{ij} \frac{\boldsymbol{x}^\top\big[\boldsymbol{\Lambda}_j^{(l-1)}\big]^{-1}\boldsymbol{x}}{\boldsymbol{x}^\top\boldsymbol{x}} \\ &\leqslant \sum_{j=1}^N \omega_{ij}\rho\big(\boldsymbol{\Lambda}_j^{-(l-1)}\big) \\ &\leqslant \max\big\{\rho\big(\boldsymbol{\Lambda}_j^{-(l-1)}\big) :\ j = 1, 2, \ldots, N\big\}\end{aligned}$$

where the first inequality follows from the arithmetic-mean-harmonic-mean inequality [76] applied to the symmetric matrices $\big\{\boldsymbol{\Lambda}_j^{-(l-1)}\big\}_j$.

Proceeding in a decreasing order from $l = L$ to $l = 1$, we obtain $\rho\big(\boldsymbol{\Theta}_i^{(k+1)}\big) \leqslant \max\big\{\rho\big(\boldsymbol{\Lambda}_j^{-(0)}\big) :\ j = \mathbb{N}_1^N\big\}$ for all $i \in \mathbb{N}_1^N$. From assumption A.4) and since $\rho\big(\boldsymbol{\Lambda}_j^{-(0)}\big) = N^{-1}w_j^{-1}\rho\big(\hat{\boldsymbol{G}}_j^{(k)}\big)$, we obtain the bound

$$\rho(\boldsymbol{\Theta}_i^{(k+1)}) \leqslant N^{-1}\max\big(\{w_j^{-1}\boldsymbol{\Phi}_j :\ j \in \mathbb{N}_1^N\}\big)$$

for all $i \in \mathbb{N}_1^N$. Since k was arbitrarily fixed, the result holds for any time.

Claim (ii). The claim directly follows from the previous result.

Appendix C. Proof of Theorem 3.3

Let us denote the vectors $\boldsymbol{v}_i^{(k)} \triangleq \big[\boldsymbol{\Lambda}_i^{(L)}\big]^{-1}\boldsymbol{\Lambda}_i^{(0)}\hat{\boldsymbol{g}}_i^{(k)}\ \forall i \in \mathbb{N}_1^N$ and note that $\|\boldsymbol{v}_i^{(k)}\|_2 \leqslant w_i\varrho\vartheta_i$ from Assumption A.3) and the result in Lemma 3.2 (i). Subsequently, we introduce the matrix $\boldsymbol{\Upsilon}^{(k)} = [\boldsymbol{v}_1^{(k)}, \boldsymbol{v}_2^{(k)}, \cdots, \boldsymbol{v}_N^{(k)}]^\top$ and note that $\|\boldsymbol{\Upsilon}^{(k)}\|_2 \leqslant \vartheta\varrho$, where $\vartheta = \max(\{\sqrt{w_i}\vartheta_i\}_{i=1}^N)$. Furthermore, let $\boldsymbol{V}^{(k)} = [\boldsymbol{\theta}_1^{(k)}, \boldsymbol{\theta}_2^{(k)}, \cdots, \boldsymbol{\theta}_N^{(k)}]^\top$.

Combining lines 15 and 18 from Algorithm 1, the policy gradient update equation for all agents becomes

$$\begin{aligned}\boldsymbol{V}^{(k+1)} &= \boldsymbol{\Omega}^{(k)}\Big[\boldsymbol{V}^{(k)} + \beta\boldsymbol{\Upsilon}^{(k)}\Big] \\ &= \Big[\prod_{t=0}^k \boldsymbol{\Omega}^{(t)}\Big]\boldsymbol{V}^{(0)} + \beta\sum_{s=0}^k\Big[\prod_{t=s}^k \boldsymbol{\Omega}^{(t)}\Big]\boldsymbol{\Upsilon}^{(s)}.\end{aligned}$$

Let $\bar{\boldsymbol{\theta}}^{(k)} \triangleq \frac{1}{N}\sum_{i=1}^N \boldsymbol{\theta}_i^{(k)}$ and $\bar{\boldsymbol{V}}^{(k)} \triangleq \big(\bar{\boldsymbol{\theta}}^{(k)}\big)^\top \otimes \mathbf{1}_N$. Assuming the initial parameter vectors to be identical across all nodes, i.e., $\boldsymbol{\theta}_i^{(0)} = \boldsymbol{\theta}_j^{(0)}$ $\forall\, v_i, v_j \in \mathcal{V}$, proceeding in a similar manner to the proof in App. A, we obtain

$$\begin{aligned}\left\|\boldsymbol{V}^{(k+1)}-\bar{\boldsymbol{V}}^{(k+1)}\right\|_2 &= \left\|\boldsymbol{Q}\boldsymbol{V}^{(k+1)}-\boldsymbol{Q}\bar{V}^{(k+1)}\right\|_2 \\ &\leqslant \beta C\sum_{s=0}^{k} q^{k+1-s}\|\boldsymbol{\Upsilon}^{(s)}\|_2 \\ &\leqslant \beta C\frac{\vartheta\rho}{1-q}\,. \end{aligned} \tag{C.1}$$

Furthermore, considering that $\boldsymbol{\Theta}_i^{(k)}$ is a covariance matrix, the following ϵ-ball result holds for the local covariance matrix estimates as

$$\begin{aligned}\left\|\boldsymbol{\Theta}_i^{(k)}-\bar{\boldsymbol{\Theta}}^{(k)}\right\|_2 &\leqslant \frac{1}{N}\sum_{j=1}^{N}\left\|\boldsymbol{\Theta}_i^{(k)}-\boldsymbol{\Theta}_j^{(k)}\right\|_2 \\ &\leqslant \frac{1}{N}\sum_{j=1}^{N}\rho\left(\boldsymbol{\Theta}_i^{(k)}\right)\rho\left(\boldsymbol{\Theta}_j^{(k)}\right)\left\|\boldsymbol{\Lambda}_j^{(L)}-\boldsymbol{\Lambda}_i^{(L)}\right\|_2 \\ &\leqslant 2\varphi\rho^2 C N^{-1}q^L \end{aligned} \tag{C.2}$$

where the triangle inequality was employed first, followed by the fact that the matrices $\left\{\boldsymbol{\Theta}_i^{(k)}\right\}_{i=1}^{N}$ are symmetric and positive definite, and from the application of Lemmas 3.1 and 3.2.

References

[1] G. Weiss (Ed.), Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence, MIT Press, Cambridge, MA, USA, 1999.
[2] L. Buşoniu, R. Babuška, B. De Schutter, A comprehensive survey of multiagent reinforcement learning, IEEE Trans. Syst. Man Cybern., Part C 38 (2) (2008) 156–172, https://doi.org/10.1109/TSMCC.2007.913919.
[3] M.Z. Win, A. Conti, S. Mazuelas, Y. Shen, W.M. Gifford, D. Dardari, M. Chiani, Network localization and navigation via cooperation, IEEE Commun. Mag. 49 (5) (2011) 56–62, https://doi.org/10.1109/MCOM.2011.5762798.
[4] M.Z. Win, Y. Shen, W. Dai, A theoretical foundation of network localization and navigation, Proc. IEEE 106 (7) (2018) 1136–1165, special issue on Foundations and Trends in Localization Technologies, https://doi.org/10.1109/JPROC.2018.2844553.
[5] R. Niu, A. Vempaty, P.K. Varshney, Received-signal-strength-based localization in wireless sensor networks, Proc. IEEE 106 (7) (2018) 1166–1182, https://doi.org/10.1109/JPROC.2018.2828858.
[6] M.Z. Win, W. Dai, Y. Shen, G. Chrisikos, H.V. Poor, Network operation strategies for efficient localization and navigation, Proc. IEEE 106 (7) (2018) 1224–1254, special issue on Foundations and Trends in Localization Technologies, https://doi.org/10.1109/JPROC.2018.2835314.
[7] Y. Bar-Shalom, P.K. Willett, X. Tian, Tracking and Data Fusion: A Handbook of Algorithms, YBS Publishing, Storrs, CT, 2011.
[8] R. Mahler, Advances in Statistical Mutisource-Multitarget Information Fusion, Artech House, Norwood, MA, 2014.
[9] A.A. Saucan, M.Z. Win, Information-seeking sensor selection for ocean-of-things, IEEE Internet Things J. 7 (10) (2020) 10072–10088, special issue on Internet of Things for Smart Ocean, https://doi.org/10.1109/JIOT.2020.2992509.
[10] S. Safavi, U.A. Khan, S. Kar, J.M.F. Moura, Distributed localization: a linear theory, Proc. IEEE 106 (7) (2018) 1204–1223, https://doi.org/10.1109/JPROC.2018.2823638.
[11] U.A. Khan, S. Kar, J.M.F. Moura, Distributed sensor localization in random environments using minimal number of anchor nodes, IEEE Trans. Signal Process. 57 (5) (2009) 2000–2016, https://doi.org/10.1109/TSP.2009.2014812.
[12] W. Dai, Y. Shen, M.Z. Win, Distributed power allocation for cooperative wireless network localization, IEEE J. Sel. Areas Commun. 33 (1) (2015) 28–40, https://doi.org/10.1109/JSAC.2014.2369631.
[13] J.S. Mullane, B.-N. Vo, M.D. Adams, B.-T. Vo, Random Finite Sets for Robot Mapping & SLAM, Springer-Verlag, Berlin, 2011, https://doi.org/10.1007/978-3-642-21390-8.
[14] Y. Cao, W. Yu, W. Ren, G. Chen, An overview of recent progress in the study of distributed multi-agent coordination, IEEE Trans. Industrial Informatics 9 (1) (2013) 427–438, https://doi.org/10.1109/TII.2012.2219061.
[15] D. Ye, M. Zhang, Y. Yang, A multi-agent framework for packet routing in wireless sensor networks, Sensors 15 (5) (2015) 10026–10047, https://doi.org/10.3390/s150510026.
[16] T. Chu, J. Wang, L. Codecà, Z. Li, Multi-agent deep reinforcement learning for large-scale traffic signal control, IEEE Trans. Intell. Transp. Syst. 21 (3) (2020) 1086–1095, https://doi.org/10.1109/TITS.2019.2901791.
[17] W. Ying, S. Dayong, Multi-agent framework for third party logistics in E-commerce, Expert Syst. Appl. 29 (2) (2005) 431–436, https://doi.org/10.1016/j.eswa.2005.04.039.
[18] A.A. Saucan, T. Chonavel, C. Sintes, J.M.L. Caillec, CPHD-DOA tracking of multiple extended sonar targets in impulsive environments, IEEE Trans. Signal Process. 64 (5) (2016) 1147–1160, https://doi.org/10.1109/TSP.2015.2504349.
[19] S. Nannuru, S. Blouin, M. Coates, M. Rabbat, Multisensor CPHD filter, IEEE Trans. Aerosp. Electron. Syst. 52 (4) (2016) 1834–1854, https://doi.org/10.1109/TAES.2016.150265.
[20] A.A. Saucan, C. Sintes, T. Chonavel, J.-M. Le Caillec, Model-based adaptive 3D sonar reconstruction in reverberating environments, IEEE Trans. Image Process. 24 (10) (2015) 2928–2940, https://doi.org/10.1109/TIP.2015.2432676.
[21] S. Das, J.M.F. Moura, Distributed Kalman filtering with dynamic observations consensus, IEEE Trans. Signal Process. 63 (17) (2015) 4458–4473, https://doi.org/10.1109/TSP.2015.2424205.
[22] P. Sharma, A. Saucan, D.J. Bucci, P.K. Varshney, Decentralized Gaussian filters for cooperative self-localization and multi-target tracking, IEEE Trans. Signal Process. 67 (22) (2019) 5896–5911, https://doi.org/10.1109/TSP.2019.2946017.
[23] S. Das, J.M.F. Moura, Consensus+innovations distributed Kalman filter with optimized gains, IEEE Trans. Signal Process. 65 (2) (2017) 467–481, https://doi.org/10.1109/TSP.2016.2617827.
[24] A.A. Saucan, P.K. Varshney, Distributed cross-entropy δ-GLMB filter for multi-sensor multi-target tracking, in: Proc. Int. Conf. on Inform. Fusion, 2018, pp. 1559–1566, https://doi.org/10.23919/ICIF.2018.8455604.
[25] T. Wang, B. Teague, M.Z. Win, Distributed situation-aware scheduling algorithm for network navigation, in: IEEE Int. Conf. Ubiquitous Wirel. Broadband ICUWB, 2017, pp. 1–5, https://doi.org/10.1109/ICUWB.2017.8250974.

[26] B. Teague, Z. Liu, F. Meyer, A. Conti, M.Z. Win, Network localization and navigation with scalable inference and efficient operation, IEEE Trans. Mobile Comput. 21 (6) (2022) 2072–2087, https://doi.org/10.1109/TMC.2020.3035511.
[27] A. Giorgetti, M. Lucchi, M. Chiani, M.Z. Win, Throughput per pass for data aggregation from a wireless sensor network via a UAV, IEEE Trans. Aerosp. Electron. Syst. 47 (4) (2011) 2610–2626, https://doi.org/10.1109/TAES.2011.6034654.
[28] M. Lucchi, A. Giorgetti, M.Z. Win, M. Chiani, Using a UAV to collect data from low-power wireless sensors, in: Proc. XIX National Congress AIDAA (Associazione Italiana di Aeronautica e Astronautica), Forli, Italy, 2007.
[29] K. Zhang, Z. Yang, H. Liu, T. Zhang, T. Başar, Fully decentralized multi-agent reinforcement learning with networked agents, in: J. Dy, A. Krause (Eds.), Proc. Int. Conf. on Mach. Learn. (ICML), 10–15 Jul, in: Proc. Mach Learn. Res., vol. 80, PMLR, Stockholm, Sweden, 2018, pp. 5872–5881.
[30] A.G. Dimakis, S. Kar, J.M.F. Moura, M.G. Rabbat, A. Scaglione, Gossip algorithms for distributed signal processing, Proc. IEEE 98 (11) (2010) 1847–1864, https://doi.org/10.1109/JPROC.2010.2052531.
[31] A. Nedić, A. Olshevsky, M.G. Rabbat, Network topology and communication-computation tradeoffs in decentralized optimization, Proc. IEEE 106 (5) (2018) 953–976, https://doi.org/10.1109/JPROC.2018.2817461.
[32] V. Mnih, K. Kavukcuoglu, D. Silver, A.A. Rusu, J. Veness, M.G. Bellemare, A. Graves, M. Riedmiller, A.K. Fidjeland, G. Ostrovski, et al., Human-level control through deep reinforcement learning, Nature 518 (7540) (2015) 529–533, https://doi.org/10.1038/nature14236.
[33] V. Mnih, A.P. Badia, M. Mirza, A. Graves, T. Harley, T.P. Lillicrap, D. Silver, K. Kavukcuoglu, Asynchronous methods for deep reinforcement learning, in: M.F. Balcan, K.Q. Weinberger (Eds.), Proc. Int. Conf. on Mach. Learn. (ICML), 20–22 Jun, in: Proc. Mach Learn. Res., vol. 48, PMLR, New York, NY, USA, 2016, pp. 1928–1937.
[34] J.N. Foerster, Y.M. Assael, N. de Freitas, S. Whiteson, Learning to communicate with deep multi-agent reinforcement learning, in: D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, R. Garnett (Eds.), Adv. Neural Inf. Process. Syst. (NeurIPS), vol. 29, Curran Associates, Inc., 2016.
[35] J.N. Foerster, G. Farquhar, T. Afouras, N. Nardelli, S. Whiteson, Counterfactual multi-agent policy gradients, in: AAAI Conference on Artificial Intelligence, vol. 32, Association for the Advancement of Artificial Intelligence (AAAI), New Orleans, Louisiana, USA, 2018, https://doi.org/10.1609/aaai.v32i1.11794.
[36] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, I. Mordatch, Multi-agent actor-critic for mixed cooperative-competitive environments, in: I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Adv. Neural Inf. Process. Syst. (NeurIPS), vol. 30, Curran Associates, Inc., 2017, pp. 6382–6393.
[37] P. Pennesi, I.C. Paschalidis, A distributed actor-critic algorithm and applications to mobile sensor network coordination problems, IEEE Trans. Autom. Control 55 (2) (2010) 492–497, https://doi.org/10.1109/TAC.2009.2037462.
[38] S. Kar, J.M.F. Moura, H.V. Poor, QD-learning: a collaborative distributed strategy for multi-agent reinforcement learning through consensus + innovations, IEEE Trans. Signal Process. 61 (7) (2013) 1848–1862, https://doi.org/10.1109/TSP.2013.2241057.
[39] S.V. Macua, J. Chen, S. Zazo, A.H. Sayed, Distributed policy evaluation under multiple behavior strategies, IEEE Trans. Autom. Control 60 (5) (2015) 1260–1274, https://doi.org/10.1109/TAC.2014.2368731.
[40] A. Mathkar, V.S. Borkar, Distributed reinforcement learning via gossip, IEEE Trans. Autom. Control 62 (3) (2017) 1465–1470, https://doi.org/10.1109/TAC.2016.2585302.
[41] H.-T. Wai, Z. Yang, Z. Wang, M. Hong, Multi-agent reinforcement learning via double averaging primal-dual optimization, in: S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, R. Garnett (Eds.), Adv. Neural Inf. Process. Syst. (NeurIPS), vol. 31, Curran Associates, Inc., 2018.
[42] M. Assran, J. Romoff, N. Ballas, J. Pineau, M. Rabbat, Gossip-based actor-learner architectures for deep reinforcement learning, in: H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, R. Garnett (Eds.), Adv. Neural Inf. Process. Syst. (NeurIPS), vol. 32, Curran Associates, Inc., 2019.
[43] C. Qu, S. Mannor, H. Xu, Y. Qi, L. Song, J. Xiong, Value propagation for decentralized networked deep multi-agent reinforcement learning, in: H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, R. Garnett (Eds.), Adv. Neural Inf. Process. Syst. (NeurIPS), vol. 32, Curran Associates, Inc., 2019.
[44] V.R. Konda, J.N. Tsitsiklis, Actor-critic algorithms, in: S. Solla, T. Leen, K. Müller (Eds.), Adv. Neural Inf. Process. Syst. (NeurIPS), vol. 12, MIT Press, 1999.
[45] R.S. Sutton, D.A. McAllester, S.P. Singh, M. Yishay, Policy gradient methods for reinforcement learning with function approximation, in: S. Solla, T. Leen, K. Müller (Eds.), Adv. Neural Inf. Process. Syst. (NeurIPS), vol. 12, MIT Press, 1999.
[46] M. Ghavamzadeh, Y. Engel, Bayesian actor-critic algorithms, in: Proc. Int. Conf. on Mach. Learn. (ICML), Association for Computing Machinery, New York, NY, USA, 2007, pp. 297–304, https://doi.org/10.1145/1273496.1273534.
[47] M. Ghavamzadeh, Y. Engel, M. Valko, Bayesian policy gradient and actor-critic algorithms, J. Mach. Learn. Res. 17 (66) (2016) 1–53.
[48] A.A. Saucan, S. Das, M.Z. Win, On multisensor activation policies for Bernoulli tracking, in: IEEE Proc. Military Commun. Conf., San Diego, CA, 2021, pp. 795–801, https://doi.org/10.1109/MILCOM52596.2021.9652984.
[49] C.E. Rasmussen, C. Williams, Gaussian Processes for Machine Learning, The MIT Press, 2005, https://doi.org/10.7551/mitpress/3206.001.0001.
[50] J.-F. Le Gall, Brownian Motion, Martingales, and Stochastic Calculus, Springer, New York, NY, USA, 2016, https://doi.org/10.1007/978-3-319-31089-3.
[51] D.P. Bertsekas, Dynamic Programming: Deterministic and Stochastic Models, Prentice Hall, Upper Saddle River, NJ, 1987.
[52] D.P. Bertsekas, J.N. Tsisiklis, Neuro-Dynamic Programming, Athena Scientific, Belmont, MA, 1996.
[53] D.P. Bertsekas, Dynamic Programming and Optimal Control, Volume I, 4th edition, Athena Scientific, Belmont, MA, 2017.
[54] D.P. Bertsekas, Rollout, Policy Iteration, and Distributed Reinforcement Learning, Athena Scientific, Belmont, MA, 2020.
[55] Y. Engel, S. Mannor, R. Meir, Reinforcement learning with Gaussian processes, in: Int. Conf. on Mach. Learn. (ICML), Bonn, Germany, 2005, pp. 201–208, https://doi.org/10.1145/1102351.1102377.
[56] B. Schölkopf, A.J. Smola, Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond, MIT Press, 2002, https://doi.org/10.7551/mitpress/4175.001.000.
[57] K. Dedecius, P.M. Djurić, Sequential estimation and diffusion of information over networks: a Bayesian approach with exponential family of distributions, IEEE Trans. Signal Process. 65 (7) (2017) 1795–1809, https://doi.org/10.1109/TSP.2016.2641380.
[58] G. Koliander, Y. El-Laham, P.M. Djurić, F. Hlawatsch, Fusion of probability density functions, Proc. IEEE 110 (4) (2022) 404–453, https://doi.org/10.1109/JPROC.2022.3154399.
[59] A.A. Saucan, V. Elvira, P.K. Varshney, M.Z. Win, Information fusion via importance sampling, IEEE Trans. Signal Inf. Process. Netw. (2023) 1–14, https://doi.org/10.1109/TSIPN.2023.3299512.
[60] D. Koller, N. Friedman, Probabilistic Graphical Models: Principles and Techniques, MIT Press, Cambridge, MA, 2009.
[61] E.B. Sudderth, A.T. Ihler, M. Isard, W.T. Freeman, A.S. Willsky, Nonparametric belief propagation, in: IEEE Conf. Comp. Vision Pattern Recog., 2003, https://doi.org/10.1109/CVPR.2003.1211409.
[62] J.K. Uhlmann, General data fusion for estimates with unknown cross covariances, in: I. Kadar, V. Libby (Eds.), Signal Process., Sensor Fusion, Target Recog. V, vol. 2755, Int. Soc. Optics Photonics, SPIE, 1996, pp. 536–547, https://doi.org/10.1117/12.243195.
[63] J.K. Uhlmann, S.J. Julier, M. Csorba, Nondivergent simultaneous map-building and localization using covariance intersection, in: S.A. Speigle (Ed.), Navigation and Control Technologies for Unmanned Systems II, vol. 3087, Int. Soc. Optics Photonics, SPIE, 1997, pp. 2–11, https://doi.org/10.1117/12.277216.
[64] S.J. Julier, J.K. Uhlmann, A non-divergent estimation algorithm in the presence of unknown correlations, in: Proc. Americ. Control Conf., vol. 4, 1997, pp. 2369–2373, https://doi.org/10.1109/ACC.1997.609105.
[65] S. Julier, J.K. Uhlmann, General decentralized data fusion with covariance intersection (CI), in: D. Hall, J. Llinas (Eds.), Handbook of Multisensor Data Fusion, CRC Press, Boca Raton, FL, 2001, pp. 269–294, Ch. 12, https://doi.org/10.1201/9781420038545-15.
[66] K. Chang, C. Chong, S. Mori, Analytical and computational evaluation of scalable distributed fusion algorithms, IEEE Trans. Aerosp. Electron. Syst. 46 (4) (2010) 2022–2034, https://doi.org/10.1109/TAES.2010.5595611.

[67] F. Nielsen, An information-geometric characterization of Chernoff information, IEEE Signal Process. Lett. 20 (3) (2013) 269–272, https://doi.org/10.1109/LSP.2013.2243726.
[68] S.J. Julier, J.K. Uhlmann, A new extension of the Kalman filter to nonlinear systems, in: Proc. AeroSense, Orlando, FL, 1997, pp. 182–193.
[69] V.D. Blondel, J.M. Hendrickx, A. Olshevsky, J.N. Tsitsiklis, Convergence in multiagent coordination, consensus, and flocking, in: Proc. IEEE Conf. on Dec. Control, 2005, pp. 2996–3000, https://doi.org/10.1109/CDC.2005.1582620.
[70] J. Wolfowitz, Products of indecomposable, aperiodic, stochastic matrices, Proc. Am. Math. Soc. 14 (5) (1963) 733–737, https://doi.org/10.1090/s0002-9939-1963-0154756-3.
[71] A.A. Saucan, M.J. Coates, M. Rabbat, A multisensor multi-Bernoulli filter, IEEE Trans. Signal Process. 65 (20) (2017) 5495–5509, https://doi.org/10.1109/TSP.2017.2723348.
[72] Y. Bar-Shalom, X.-R. Li, T. Kirubarajan, Estimation with Applications to Tracking and Navigation, John Wiley & Sons, Hoboken, NJ, 2001, https://doi.org/10.1002/0471221279.
[73] B.-T. Vo, C.M. See, N. Ma, W.T. Ng, Multi-sensor joint detection and tracking with the Bernoulli filter, IEEE Trans. Aerosp. Electron. Syst. 48 (2) (2012) 1385–1402, https://doi.org/10.1109/TAES.2012.6178069.
[74] D. Schuhmacher, B.-T. Vo, B.-N. Vo, A consistent metric for performance evaluation of multi-object filters, IEEE Trans. Signal Process. 56 (8) (2008) 3447–3457, https://doi.org/10.1109/TSP.2008.920469.
[75] R. Bhatia, Matrix Analysis, Graduate Texts in Mathematics, vol. 169, Springer-Verlag, New York, 1997, https://doi.org/10.1007/978-1-4612-0653-8.
[76] B. Mond, J.E. Pečarić, A mixed arithmetic-mean-harmonic-mean matrix inequality, Linear Algebra Appl. 237–238 (1996) 449–454, Linear Algebra and Statistics: In Celebration of C. R. Rao's 75th Birthday (September 10, 1995), https://doi.org/10.1016/0024-3795(95)00269-3.