# Distributed State Estimation With Deep Neural Networks for Uncertain Nonlinear Systems Under Event-Triggered Communication

Federico M. Zegers, *Member, IEEE*, Runhan Sun, Girish Chowdhary, *Senior Member, IEEE*, and Warren E. Dixon, *Fellow, IEEE*

***Abstract*—This work explores the distributed state estimation problem for an uncertain, nonlinear, and continuous-time system. Given a sensor network, each agent is assigned a deep neural network (DNN) that is used to approximate the system's dynamics. Each agent updates the weights of their DNN through a multiple timescale approach, i.e., the outer layer weights are updated online with a Lyapunov-based gradient descent update law, and the inner layer weights are updated concurrently using a supervised learning strategy. To promote the efficient use of network resources, the distributed observer uses event-triggered communication. A nonsmooth Lyapunov analysis demonstrates that the distributed event-triggered observer achieves uniformly ultimately bounded state reconstruction. A simulation example of a five-agent sensor network estimating the state of a two-link robotic manipulator tracking a desired trajectory is provided to validate the result and showcase the performance improvements afforded by DNNs.**

***Index Terms*—Multi-agent systems, Lyapunov methods, nonlinear control systems, wireless sensor networks, deep learning, state estimation.**

## I. Introduction

A wireless sensor network (WSN) is a multi-agent system of wirelessly linked autonomous sensors that are scattered over an area to monitor desired phenomena [1]. By sharing partially observable state measurements of a system with their neighbors and leveraging a consensus algorithm, WSNs are capable of estimating the state of a system in a distributed fashion [2]. Such distributed state estimation does not require a centralized data-fusion module. Additionally, it is preferable to centralized state estimation methods because it can better accommodate each agent's limited computing capacity, eliminate single points of failure, and promote scalability.

In [3], the authors developed a distributed filter for a known linear time-varying stochastic system that attains stable state estimation using adaptive weights under periodic communication. Event-triggered control is a popular technique employed to coordinate multi-agent systems while more efficiently utilizing network resources via intermittent communication [4], [5]. Given that agents of a WSN may be powered by limited energy sources, [6] and [7] presented distributed state estimation strategies for linear systems that use event-triggered communication to economize resources. Similarly, [8] developed a distributed observer with stochastic event-triggered communication for a known linear time-varying system.

While the works in [2], [3], [6], [7], and [8] provide valuable contributions to the literature on distributed state estimation, these results primarily focus on systems with known linear models. Moreover, although results on distributed state estimation for nonlinear systems exist, e.g., [9], [10], [11], the distributed state estimation problem for uncertain nonlinear systems remains relatively open. In [9], the authors solved the state estimation problem for a known nonlinear discrete-time system by exploiting the Taylor series. In particular, the known nonlinear state estimation error dynamics are expressed as the sum of a linear function and remainder comprised of higher order terms, where the remainder is assumed to be bounded by a function of the norm of the state estimation error squared. The distributed extended Kalman filter is then used to yield a locally uniformly ultimately bounded (UUB) state estimation error. In [10], a set-membership filter is utilized to solve the distributed state estimation problem for a known nonlinear discrete-time system. Like [9], the system's nonlinear function is written as the sum of a known linear function and Lagrange remainder, where the remainder is accurately estimated by the proposed algorithm. The result in [11] developed a distributed state observer for a nonlinear discrete-time system, where the agents of the WSN use event-triggered communication over a randomly time-varying topology that is modeled with Gilbert–Elliott channels. Under a robust control approach, the result demonstrates that the filtered error covariance is bounded and minimized at each time instant.

The enhanced computing power of modern processors and abundance of data encourage the development of a distributed observer capable of applying machine learning techniques to improve state reconstruction. However, the protocols that train the weights of a deep neural network (DNN) do not usually have an accompanying stability analysis, which hinders their implementation for online estimation and control. Recently, the works in [12] and [13] contributed a model reference adaptive control technique that utilizes a DNN as an adaptive element while demonstrating that the estimation error is UUB via a Lyapunov stability analysis. These works are among the first to employ DNNs for real-time adaptive control while providing a formal stability assurance. This guarantee is made possible by updating the outer layer weights of the DNN with state feedback, while the inner layer weights are modified using batch updates—harnessing the power of data-driven learning. Based on this observation, the authors in [14] developed an adaptive controller for an uncertain nonlinear dynamical system capable of asymptotically tracking a desired trajectory while using a DNN to approximate the uncertain dynamics. In [14], the outer layer weights

Manuscript received 20 June 2022; revised 29 August 2022; accepted 2 October 2022. Date of publication 25 October 2022; date of current version 26 April 2023. This work was supported in part by A Task Order contract with the Air Force Research Laboratory, Munitions Directorate at Eglin AFB, Office of Naval Research under Grant N00014-21-1-2481, in part by NEEC under Award N00174-18-1-0003, in part by AFOSR under Award FA9550-18-1-0109 and Award FA9550-19-1-0169. Recommended by Senior Editor Tetsuya Iwasaki and Guest Editors George J. Pappas, Anuradha M. Annaswamy, Manfred Morari, Claire J. Tomlin, Rene Vidal, and Melanie N. Zeilinger. *(Corresponding Author: Federico M. Zegers.)*

Federico M. Zegers, Runhan Sun, and Warren E. Dixon are with the Department of Mechanical and Aerospace Engineering, University of Florida, Gainesville, FL 32603 USA (e-mail: fredzeg@ufl.edu; runhansun@ufl.edu; wdixon@ufl.edu).

Girish Chowdhary is with the Department of Agricultural and Biological Engineering, University of Illinois Urbana-Champaign, Champaign, IL 61801 USA (e-mail: girishc@illinois.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TAC.2022.3217022.

Digital Object Identifier 10.1109/TAC.2022.3217022

are trained in real time with a Lyapunov-based update law, while the inner layer weights are updated using a supervised learning algorithm, i.e., the Levenberg–Marquardt algorithm. The results in [12], [13], and [14] solve the desired trajectory tracking problem for a single-agent system and demonstrate that multiple timescale learning, i.e., training the weights of a DNN with a combination of online state feedback and concurrent supervised learning, can produce improved performance when compared to traditional adaptive techniques.

Motivated by [12], [13], and [14], a DNN-based event-triggered distributed state observer for an uncertain nonlinear continuous-time system is developed. Unlike [9] and [10], the system's nonlinear function is uncertain and is not expressed as a linear function with remainder. Instead, the nonlinear dynamics are approximated via DNNs—an advancement over [11]. Furthermore, the observer is shown to yield UUB state estimation while being robust to bounded disturbances through a nonsmooth Lyapunov stability analysis. Similar to [12], [13], and [14], we develop a multiple timescale learning strategy to train the DNN weights. In particular, the outer layer weights of each DNN are adjusted online with a Lyapunov-based update law and output feedback to ensure stability. The inner layer weights are updated concurrently using collected data and supervised learning. However, unlike [12], [13], and [14], which consider a single-agent system, we contemplate a multi-agent system, where each agent is assigned their own DNN that is managed independently. Furthermore, the update laws for the outer layer weights in [12], [13], and [14] utilize state feedback while the ones proposed in this article use output feedback. The development is further complicated by the use of event-triggered control, facilitating intermittent and asynchronous communication. The theoretical results are validated through a simulation example. The state of a two-link robotic manipulator tracking a desired trajectory is estimated by a five-agent sensor network in two simulations. In the first simulation, each observer updates their outer layer weights with output feedback while the inner layer weights are held constant. The result confirms Theorem 1, i.e., the state estimation error of each agent is UUB. The second simulation is identical to the first, except the inner layer weights are updated using the Levenberg–Marquardt algorithm and collected data, where the state estimation error of each agent is UUB and approximately 77% more accurate relative to the first simulation.

## II. Preliminaries

### A. Notation

Let $\mathbb{R}$ and $\mathbb{Z}$ denote the set of real numbers and integers, respectively. For $x \in \mathbb{R}$, let $\mathbb{R}_{\geq x} \triangleq [x, \infty)$, $\mathbb{R}_{>x} \triangleq (x, \infty)$, $\mathbb{Z}_{\geq x} \triangleq \mathbb{R}_{\geq x} \cap \mathbb{Z}$, and $\mathbb{Z}_{>x} \triangleq \mathbb{R}_{>x} \cap \mathbb{Z}$. The $p \times q$ zero matrix and the $p \times 1$ zero column vector are denoted by $0_{p\times q}$ and $0_p$, respectively. The $p \times p$ identity matrix and the $p \times 1$ column vector of ones are denoted by $I_p$ and $1_p$, respectively. The Euclidean norm of a vector $r \in \mathbb{R}^p$ is denoted by $\|r\| \triangleq \sqrt{r^\top r}$. Given $M \in \mathbb{Z}_{>1}$, let $[M] \triangleq \{1, 2, \ldots, M\}$. The Kronecker product of $A \in \mathbb{R}^{p\times q}$ and $B \in \mathbb{R}^{u\times v}$ is denoted by $A \otimes B \in \mathbb{R}^{pu\times qv}$. The block diagonal matrix whose diagonal blocks consist of $G_1, G_2, \ldots, G_n$ is denoted by $\mathrm{diag}(G_1, G_2, \ldots, G_n)$. Let $\mathrm{tr}(A)$ denote the trace of a square matrix $A \in \mathbb{R}^{p\times p}$. Let $\mathrm{vec}(\cdot)$ denote the vectorization transformation that converts a matrix into a column vector. Given functions $f : \mathbb{R}^q \to \mathbb{R}^r$ and $g : \mathbb{R}^p \to \mathbb{R}^q$, the symbol $\circ$ denotes function composition, i.e., $f \circ g : \mathbb{R}^p \to \mathbb{R}^r$ and $(f \circ g)(x) = f(g(x))$. Given $s \in \mathbb{Z}_{>1}$ and $\{w_i\}_{i\in[s]} \subset \mathbb{R}^d$, let $(w_i)_{i\in[s]} \triangleq [w_1^\top, w_2^\top, \ldots, w_s^\top]^\top \in \mathbb{R}^{sd}$.

### B. Graphs

Let $\mathcal{G} \triangleq (\mathcal{V}, \mathcal{E})$ be a static and undirected graph with node set $\mathcal{V} \triangleq [N]$, $N \in \mathbb{Z}_{>0}$, and edge set $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. The edge $(i, j) \in \mathcal{E}$ if and only if node $i$ can send information to node $j$. Since $\mathcal{G}$ is undirected, $(i, j) \in \mathcal{E}$ if and only if $(j, i) \in \mathcal{E}$. Let $\mathcal{A} \triangleq [a_{ij}] \in \mathbb{R}^{N\times N}$ denote the adjacency matrix of $\mathcal{G}$, where $a_{ij} > 0$ if and only if $(j, i) \in \mathcal{E}$. Otherwise, $a_{ij} = 0$. An undirected graph is connected if and only if there exists a sequence of edges in $\mathcal{E}$ between any two distinct nodes. The neighbor set of node $i$ is $\mathcal{N}_i \triangleq \{j \in \mathcal{V} : (j, i) \in \mathcal{E}\}$, where $1 \leq |\mathcal{N}_i| < N - 1$. Moreover, no self-loops are considered, and, therefore, $a_{ii} \triangleq 0$ for all $i \in \mathcal{V}$. The degree matrix of $\mathcal{G}$ is denoted by the diagonal matrix $\Delta \triangleq [\Delta_{ij}] \in \mathbb{R}^{N\times N}$, where $\Delta_{ij} \triangleq 0$ for all $i \neq j$, and $\Delta_{ii} \triangleq \sum_{j\in\mathcal{V}} a_{ij}$. The Laplacian matrix of $\mathcal{G}$ is denoted by $\mathcal{L} \triangleq \Delta - \mathcal{A} \in \mathbb{R}^{N\times N}$.

## III. System Dynamics and Network Topology

Consider a system with model

$$\dot{x}_0 = f(x_0) + d, \tag{1}$$

where $x_0 \in \mathbb{R}^n$ represents the state variable, the nonlinear and locally Lipschitz function $f : \mathbb{R}^n \to \mathbb{R}^n$ is unknown, and $d : [0, \infty) \to \mathbb{R}^n$ is a locally Lipschitz exogenous disturbance. Note that $d$ is bounded, that is, there exists a $d_{\max} \in \mathbb{R}_{>0}$ such that $\|d(t)\| \leq d_{\max}$ for all $t \geq 0$. Furthermore, consider a sensor network composed of $N$ agents, which are indexed by $\mathcal{V}$. For each $i \in \mathcal{V}$, agent $i$ can continuously measure the output of the system in (1), which is given by $y_i \triangleq C_i x_0 \in \mathbb{R}^m$. Note that $C_i \in \mathbb{R}^{m\times n}$ represents the known output matrix of agent $i$. The agents in the sensor network may have different sensing capabilities, i.e., each agent may be able to measure a different function of the system's state. Each agent is also capable of intermittently communicating with its neighbors, where the flow of information between the agents in the sensor network is modeled by the static, connected, and undirected communication graph $\mathcal{G}$.

## IV. Objective

The objective is to develop a distributed observer capable of reconstructing the state of the unknown nonlinear system in (1) by using output feedback. The observer employs event-triggered communication to promote the efficient use of network resources. Since the system model in (1) is unknown, we desire an observer that concurrently utilizes online and offline learning techniques to estimate the function $f$. To quantify the state reconstruction objective, let the state estimation error of agent $i \in \mathcal{V}$ be

$$e_{1,i} \triangleq \hat{x}_i - x_0 \in \mathbb{R}^n, \tag{2}$$

where $\hat{x}_i \in \mathbb{R}^n$ denotes the estimate of $x_0$ as computed by agent $i$. For any time $t \geq 0$, the state estimation error $e_{1,i}(t)$ is an unmeasurable signal that is used only in the analysis. Let $\{t_k^i\}_{k=0}^\infty$ be a strictly increasing sequence of event times for agent $i$, where $t_k^i$ denotes the $k$th instant agent $i$ samples its state estimate $\hat{x}_i$ and broadcasts it to all agents $j \in \mathcal{N}_i$. Note that the broadcast information is received by all neighbors simultaneously. To facilitate the use of event-triggered control, let the variable $\tilde{x}_i \in \mathbb{R}^n$ represent the sampled state estimate of agent $i$. More precisely, let

$$\tilde{x}_i(t) \triangleq \hat{x}_i(t_k^i), \quad t \in [t_k^i, t_{k+1}^i). \tag{3}$$

For each event time of agent $i$, that is, $t_k^i$, $\tilde{x}_i$ is the information that agent $i$ broadcasts to each agent $j \in \mathcal{N}_i$. Moreover, as will be shown in the proceeding section, $\{\tilde{x}_j\}_{j\in\mathcal{N}_i\cup\{i\}}$ represents the local information agent $i$ will use to update its estimate of $x_0$. Using this sampled state estimate, let the sampled state estimation error of agent $i$ be given by

$$e_{2,i} \triangleq \tilde{x}_i - \hat{x}_i \in \mathbb{R}^n. \tag{4}$$

Since $\tilde{x}_i(t)$ is constant for all $t \in [t_k^i, t_{k+1}^i)$, the sampled state estimation error in (4) measures the quality of the broadcast state estimate of agent $i$ and will be used to determine when $\tilde{x}_i$ should be updated. For each agent $i \in \mathcal{V}$, let $\hat{y}_i \triangleq C_i\hat{x}_i \in \mathbb{R}^m$ denote the estimated output of the system as computed by agent $i$, which exploits knowledge of the output matrix $C_i$. The output estimation error of agent $i$ is

$$e_{3,i} \triangleq \hat{y}_i - y_i = C_i e_{1,i} \in \mathbb{R}^m. \tag{5}$$

By construction, agent $i$ can measure $e_{2,i} = \tilde{x}_i - \hat{x}_i$ and $e_{3,i} = \hat{y}_i - y_i$, where $e_{3,i}$ is used to drive the state estimate of agent $i$ toward the state of the system.

*Definition 1: Let* $\nu > 0$. The sensor network is said to achieve $\nu$-approximate state reconstruction whenever $\|e_{1,i}\| \le \nu$ for all $i \in \mathcal{V}$.

Given a user-defined $\nu > 0$, the sensor network successfully estimates the state $x_0$ of the system in (1) when it attains $\nu$-approximate state reconstruction.

## V. Observer Development

The following assumption is a necessary condition allowing state reconstruction.

*Assumption 1:* The state of the system in (1) evolves within a compact set $\mathcal{D} \subset \mathbb{R}^n$, i.e., $x_0(t) \in \mathcal{D}$ for all $t \ge 0$.

Since $f$ is continuous and $x_0$ is contained within a compact set, we can invoke the Stone–Weierstrass theorem to express the nonlinear dynamics in (1) restricted to $\mathcal{D}$ as

$$f(x_0) = W_0^\top \sigma\left(\Phi(x_0)\right) + \varepsilon(x_0). \tag{6}$$

In (6), $W_0 \in \mathbb{R}^{L\times n}$ is the ideal outer layer weight matrix, $\sigma : \mathbb{R}^p \to \mathbb{R}^L$ is a user-selected bounded and continuous activation function,[1] $\Phi : \mathbb{R}^n \to \mathbb{R}^p$ encodes the ideal inner DNN, and $\varepsilon : \mathbb{R}^n \to \mathbb{R}^n$ is the bounded function reconstruction error [15, Th. 7.32]. Note that $W_0$, $\Phi$, and $\varepsilon$ are unknown. The ideal inner DNN can be expressed as

$$\Phi(x_0) = \left(W_\ell^\top \phi_\ell \circ W_{\ell-1}^\top \phi_{\ell-1} \circ \cdots \circ W_1^\top \phi_1\right)(x_0), \tag{7}$$

where $\ell \in \mathbb{Z}_{\ge 1}$ denotes the number of user-defined inner layers of the DNN, $W_q \in \mathbb{R}^{L_q \times n_{q+1}}$, for $q \in [\ell]$, denotes the ideal weight matrix for the $q$th inner layer, and $\phi_q : \mathbb{R}^{n_q} \to \mathbb{R}^{L_q}$ is a user-selected bounded and continuous activation function corresponding to the $q$th inner layer. Note that $n_1 = n$, $n_{\ell+1} = p$, and $W_q$ is unknown for each $q \in [\ell]$. Using (6), the system model in (1) can be expressed as

$$\dot{x}_0 = W_0^\top \sigma(\Phi(x_0)) + \varepsilon(x_0) + d. \tag{8}$$

Guided by the structure of (8), the distributed observer of agent $i \in \mathcal{V}$ is designed as

$$\begin{aligned}\dot{\hat{x}}_i &\triangleq \widehat{W}_i^\top \sigma(\widehat{\Phi}_i(\hat{x}_i)) + K_1\left(z_i - C_i^\top e_{3,i}\right),\\ z_i &\triangleq \sum_{j\in\mathcal{V}} a_{ij}\left(\tilde{x}_j - \tilde{x}_i\right).\end{aligned} \tag{9}$$

The variable $\widehat{W}_i \in \mathbb{R}^{L\times n}$ represents the estimated outer layer weight matrix of the system as computed by agent $i$, $\widehat{\Phi}_i : \mathbb{R}^n \to \mathbb{R}^p$ encodes the estimated inner DNN of agent $i$, and $K_1 \in \mathbb{R}^{n\times n}$ is the symmetric solution to the bilinear matrix inequality (BMI)

$$\mathbf{M} \triangleq \frac{1}{2}\mathbf{K}_1 C^\top C + \frac{1}{2} C^\top C \mathbf{K}_1 + \mathbf{K}_1 \mathbf{L} \ge k_1 I_{nN}. \tag{10}$$

Observe that $\mathbf{K}_1 \triangleq I_N \otimes K_1 \in \mathbb{R}^{nN\times nN}$, $\mathbf{L} \triangleq \mathcal{L} \otimes I_n \in \mathbb{R}^{nN\times nN}$, $C \triangleq \text{diag}(C_1, C_2, \ldots, C_N) \in \mathbb{R}^{mN\times nN}$ denotes the output matrix of the sensor network, and $k_1 \in \mathbb{R}_{>0}$ is a user-defined parameter. The BMI in (10) encodes an observability condition that originates from the stability analysis (see Section VI). The estimated inner DNN of agent $i$, namely, $\widehat{\Phi}_i(\hat{x}_i)$, is modeled as a piecewise continuous function that is similar in structure to (7), i.e.,

$$\widehat{\Phi}_i\left(\hat{x}_i\right) \triangleq \left(\widehat{W}_{\ell,i}^\top \phi_\ell \circ \widehat{W}_{\ell-1,i}^\top \phi_{\ell-1} \circ \cdots \circ \widehat{W}_{1,i}^\top \phi_1\right)\left(\hat{x}_i\right), \tag{11}$$

where $\widehat{W}_{q,i} \in \mathbb{R}^{L_q\times n_{q+1}}$, for $q \in [\ell]$, represents the estimated weight of the $q$th layer of agent $i$'s inner DNN. While $\phi_q$ is continuous for each $q \in [\ell]$, the inner weights $\{\widehat{W}_{q,i} : q \in [\ell]\}$ are piecewise constant by design. Let $\{T_p^i\}_{p=1}^\infty$ be a strictly increasing sequence of inner DNN update times for agent $i$, where $T_p^i$ denotes the $p$th instant agent $i$ updates its inner DNN weights, that is, $\{\widehat{W}_{q,i} : q \in [\ell]\}$. While the sensor network attempts to estimate the state of the system, each agent can individually train its inner DNN weights using any suitable offline method.[2] Once, for example, agent $i$ completes the training of its inner DNN weights, agent $i$ can switch in the new weights for use in (9), which takes place at time $T_p^i$, for some $p \in \mathbb{Z}_{\ge 1}$. Observe that the weights $\{\widehat{W}_{q,i} : q \in [\ell]\}$ are held constant until the next time they are updated by agent $i$, e.g., $T_{p+1}^i$, which need not occur. Hence, for each $i \in \mathcal{V}$, the inner DNN weights of each agent are updated discretely and simultaneously.

The error between the ideal outer layer weight matrix and the estimated outer layer weight matrix of the system as computed by agent $i$ is

$$\widetilde{W}_i \triangleq W_0 - \widehat{W}_i \in \mathbb{R}^{L\times n}. \tag{12}$$

Observe that, for any $t \ge 0$, $\widetilde{W}_i(t)$ is not measurable because the ideal outer layer weight matrix $W_0$ is unknown. Let $\tilde{\omega}_i \triangleq \text{vec}(\widetilde{W}_i)$ and $\omega_i \triangleq \text{vec}(\widehat{W}_i)$. Based on the subsequent stability analysis, the outer layer weight update law of agent $i$, which is embedded within the continuous projection operator denoted by $\text{proj}(\cdot,\cdot)$ and defined in [17, Eq. 4], is designed as[3]

$$\dot{\omega}_i \triangleq \text{proj}\left(\mu_i, \omega_i\right), \ \mu_i \triangleq -\text{vec}(\Gamma_i \sigma(\widehat{\Phi}_i\left(\hat{x}_i\right)) e_{3,i}^\top C_i), \tag{13}$$

where $\Gamma_i \in \mathbb{R}^{L\times L}$ is a user-defined positive definite matrix used to adjust the learning rate of $\omega_i$. See Appendix A for an algorithm summarizing the implementation of the observer. The closed-loop dynamics of the ensemble are now derived. Substituting (2) and (4) into the definition of $z_i$ in (9) yields

$$z_i = \sum_{j\in\mathcal{V}} a_{ij}(e_{1,j} - e_{1,i}) + \sum_{j\in\mathcal{V}} a_{ij}(e_{2,j} - e_{2,i}). \tag{14}$$

Substituting (5), (8)–(12), and (14) into the time derivative of (2) while adding and subtracting $W_0^\top \sigma(\widehat{\Phi}_i(\hat{x}_i))$ yields

$$\begin{aligned}\dot{e}_{1,i} = &-\widetilde{W}_i^\top \sigma(\widehat{\Phi}_i\left(\hat{x}_i\right)) - K_1 C_i^\top C_i e_{1,i} + \chi_i\\ &+ K_1 \sum_{j\in\mathcal{V}} a_{ij}(e_{1,j} - e_{1,i}) + K_1 \sum_{j\in\mathcal{V}} a_{ij}(e_{2,j} - e_{2,i}),\end{aligned} \tag{15}$$

where $\chi_i \triangleq W_0^\top(\sigma(\widehat{\Phi}_i(\hat{x}_i)) - \sigma(\Phi(x_0))) - \varepsilon(x_0) - d \in \mathbb{R}^n$. Substituting (3), (5), (9), and (14) into the time derivative of (4) yields

$$\begin{aligned}\dot{e}_{2,i} = &-\widehat{W}_i^\top \sigma(\widehat{\Phi}_i\left(\hat{x}_i\right)) + K_1 C_i^\top C_i e_{1,i}\\ &- K_1 \sum_{j\in\mathcal{V}} a_{ij}(e_{1,j} - e_{1,i}) - K_1 \sum_{j\in\mathcal{V}} a_{ij}(e_{2,j} - e_{2,i}).\end{aligned} \tag{16}$$

[1] Examples of bounded and continuous activation functions are the sigmoid, hyperbolic tangent, and Gaussian functions. Activation functions with vector-valued inputs are applied elementwise.

[2] An example training strategy is discussed in Section VII and additional training strategies can be found in [12] and [16].

[3] The projection operator is used to ensure $\widehat{W}_i(t)$ remains within the set $\Omega \triangleq \{W \in \mathbb{R}^{L\times n} : \|W\| \le \bar{\omega}\}$ for all $t \ge 0$, where $\bar{\omega} \in \mathbb{R}_{>0}$ is a user-defined parameter.

To express the closed-loop ensemble dynamics and subsequent development in a compact form, let $e_1 \triangleq (e_{1,i})_{i\in\mathcal{V}} \in \mathbb{R}^{nN}$, $e_2 \triangleq (e_{2,i})_{i\in\mathcal{V}} \in \mathbb{R}^{nN}$, and $e_3 \triangleq (e_{3,i})_{i\in\mathcal{V}} \in \mathbb{R}^{mN}$, which represent the state estimation error, sampled state estimation error, and output estimation error of the ensemble, respectively. Furthermore, consider $\hat{x} \triangleq (\hat{x}_i)_{i\in\mathcal{V}} \in \mathbb{R}^{nN}$, $\chi \triangleq (\chi_i)_{i\in\mathcal{V}} \in \mathbb{R}^{nN}$, and $z \triangleq (z_i)_{i\in\mathcal{V}} \in \mathbb{R}^{nN}$. The block diagonal matrix composed from the DNN outer layer weight errors is $\widetilde{W} \triangleq \mathrm{diag}(\widetilde{W}_1, \widetilde{W}_2, \ldots, \widetilde{W}_N) \in \mathbb{R}^{LN\times nN}$. Similarly, the block diagonal matrix consisting of the DNN outer layer weight estimates is $\widehat{W} \triangleq \mathrm{diag}(\widehat{W}_1, \widehat{W}_2, \ldots, \widehat{W}_N) \in \mathbb{R}^{LN\times nN}$. The stacked vectorizations of the DNN outer layer weight errors $\{\widetilde{W}_i\}_{i\in\mathcal{V}}$ and DNN outer layer weight estimates $\{\widehat{W}_i\}_{i\in\mathcal{V}}$ are denoted by $\tilde{\omega} \triangleq (\tilde{\omega}_i)_{i\in\mathcal{V}} \in \mathbb{R}^{nLN}$ and $\omega \triangleq (\omega_i)_{i\in\mathcal{V}} \in \mathbb{R}^{nLN}$, respectively. Let $\Gamma \triangleq \mathrm{diag}(\Gamma_1, \Gamma_2, \ldots, \Gamma_N) \in \mathbb{R}^{LN\times LN}$ and $\sigma(\widehat{\Phi}(\hat{x})) \triangleq (\sigma(\widehat{\Phi}_i(\hat{x}_i)))_{i\in\mathcal{V}} \in \mathbb{R}^{LN}$. Using (15), (16), and the stacked expressions defined above, the closed-loop dynamics of $e_1$ and $e_2$ are

$$\begin{aligned}\dot{e}_1 = &-\widetilde{W}^\top \sigma(\widehat{\Phi}(\hat{x})) - \mathbf{K}_1 C^\top C e_1 - \mathbf{K}_1 \mathbf{L} e_1 \\ &- \mathbf{K}_1 \mathbf{L} e_2 + \chi, \end{aligned} \tag{17}$$

$$\begin{aligned}\dot{e}_2 = &-\widehat{W}^\top \sigma(\widehat{\Phi}(\hat{x})) + \mathbf{K}_1 C^\top C e_1 + \mathbf{K}_1 \mathbf{L} e_1 \\ &+ \mathbf{K}_1 \mathbf{L} e_2, \end{aligned} \tag{18}$$

respectively. Since $W_0$ is fixed, $\sigma$ is a bounded function, the function reconstruction error $\varepsilon$ is bounded, and the disturbance is bounded, there exists a constant $\chi_{\max} \in \mathbb{R}_{>0}$ such that $\|\chi(t)\| \leq \chi_{\max}$ for all $t \geq 0$. Using (5), (9), and (14), it also follows that

$$\dot{\hat{x}} = \widehat{W}^\top \sigma(\widehat{\Phi}(\hat{x})) - \mathbf{K}_1 C^\top C e_1 - \mathbf{K}_1 \mathbf{L} e_1 - \mathbf{K}_1 \mathbf{L} e_2. \tag{19}$$

Using (14), $z$ can be expressed as $z = -\mathbf{L}e_1 - \mathbf{L}e_2$, where Young's inequality yields

$$-\|e_1\|^2 \leq \|e_2\|^2 - \|z\|^2 / \left(2\|\mathbf{L}\|^2\right). \tag{20}$$

Note that (20) is a useful inequality employed in the development of the event trigger mechanisms for the sensor network.

Let $\beta \in \mathbb{R}_{\geq 0}$ be a timer variable that evolves according to $\dot{\beta} = 1$ with $\beta(0) = 0$. Let $\xi \triangleq \left[e_1^\top, e_2^\top, \hat{x}^\top, \tilde{\omega}^\top, \omega^\top, \beta\right]^\top \in \mathcal{X}$ and $\mathcal{X} \triangleq \mathbb{R}^{3nN} \times \mathbb{R}^{2nLN} \times \mathbb{R}$ represent the state vector and state space, respectively, of the sensor network. Using (5), (12), (13), and (17)–(19), the closed-loop dynamics of the sensor network are $\dot{\xi} = h(\xi)$, where

$$h(\xi) \triangleq \begin{bmatrix} -\widetilde{W}^\top \sigma(\widehat{\Phi}(\hat{x})) - \mathbf{K}_1 C^\top C e_1 + \mathbf{K}_1 z + \chi \\ -\widehat{W}^\top \sigma(\widehat{\Phi}(\hat{x})) + \mathbf{K}_1 C^\top C e_1 - \mathbf{K}_1 z \\ \widehat{W}^\top \sigma(\widehat{\Phi}(\hat{x})) - \mathbf{K}_1 C^\top C e_1 + \mathbf{K}_1 z \\ -(\mathrm{proj}(-\mathrm{vec}(\Gamma_i \sigma(\widehat{\Phi}_i(\hat{x}_i)) e_{1,i}^\top C_i^\top C_i), \omega_i))_{i\in\mathcal{V}}^\top \\ (\mathrm{proj}(-\mathrm{vec}(\Gamma_i \sigma(\widehat{\Phi}_i(\hat{x}_i)) e_{1,i}^\top C_i^\top C_i), \omega_i))_{i\in\mathcal{V}}^\top \\ 1 \end{bmatrix}. \tag{21}$$

Recall $z = -\mathbf{L}e_1 - \mathbf{L}e_2$, and the $i$th coordinate of $\chi$ can be written as a function of $\xi$ using the timer $\beta$ and (2). Whenever there is a communication event for agent $i$, the state $\xi$ jumps, where $e_{2,i}$ is reset to $0_n$ and all other variables in $\xi$ are mapped to themselves.

## VI. Stability Analysis

The following objects are presented to facilitate the analysis. Let $\widetilde{\chi} \triangleq \chi - (I_{nN} - C^\top C)\widetilde{W}^\top \sigma(\widehat{\Phi}(\hat{x}))$. Observe that $C$ is bounded by construction. Similarly, $\widetilde{W}$ is bounded since $W_0$ is fixed and the projection operator ensures that $\widehat{W}_i$ is bounded for each $i \in \mathcal{V}$. Moreover, $\sigma$ is bounded by design. Hence, there exists a $\widetilde{\chi}_{\max} \in \mathbb{R}_{>0}$ such that $\|\widetilde{\chi}(t)\| \leq \widetilde{\chi}_{\max}$ for all $t \geq 0$. Furthermore, there exists a constant $\widetilde{W}_{\max} \in \mathbb{R}_{>0}$ such that $\mathrm{tr}\,(\widetilde{W}^\top \Gamma^{-1} \widetilde{W}) \leq 2\widetilde{W}_{\max}$, which can be made arbitrarily small through the choice of $\Gamma$. Select $\kappa > 0$, $k_2 > 1/\kappa$, $\delta > 0$, $\rho \geq \widetilde{\chi}_{\max}$, and $\epsilon > 0$. It then follows that $k_1 \triangleq k_2 + \rho^2/\delta > 0$, $\alpha \triangleq k_2 - 1/\kappa > 0$, and $\bar{\delta} \triangleq \delta + \epsilon > 0$. Consider the event trigger function

$$\begin{aligned} \mathbf{T}_i(\xi) &\triangleq \mathtt{C}_2 \|z_i\|^2 - \mathtt{C}_1 \|e_{2,i}\|^2 + \frac{\epsilon}{N}, \\ \mathtt{C}_1 &\triangleq \frac{k_2}{2} + \frac{\kappa}{2}\|\mathbf{K}_1 \mathbf{L}\|^2,\ \mathtt{C}_2 \triangleq \frac{k_2}{4\|\mathbf{L}\|^2}. \end{aligned} \tag{22}$$

Since $k_2$ and $\kappa$ are positive, $\mathtt{C}_1 > 0$ and $\mathtt{C}_2 > 0$. The event times in $\{t_k^i\}_{k=0}^\infty$ that dictate when agent $i$ communicates its sampled state estimate $\tilde{x}_i$, as outlined in (3), are generated by the event trigger mechanism[4]

$$t_{k+1}^i \triangleq \inf\left\{t > t_k^i : \mathbf{T}_i(\xi) = 0\right\}. \tag{23}$$

*Remark 1:* The solution to the BMI in (10) and the constants $\mathtt{C}_1$ and $\mathtt{C}_2$ in the event trigger function in (22) depend on the graph Laplacian $\mathcal{L}$ and the output matrices $\{C_i\}_{i\in\mathcal{V}}$, which may be initially unknown to all agents. In such a scenario, distributed algorithms, such as that in [18], can leverage the static and connected communication network to compute $\mathcal{L}$. Moreover, the output matrices $\{C_i\}_{i\in\mathcal{V}}$ can be disseminated throughout the sensor network using gossip protocols, such as that in [19].

*Theorem 1:* For every $i \in \mathcal{V}$, the observer in (9) with outer layer weight update law in (13) render the sensor network state estimation error $e_1$ UUB in the sense that

$$\begin{aligned} \|e_1(t)\|^2 \leq &\left(\|e_1(0)\|^2 + \mathrm{tr}\left(\widetilde{W}^\top(0)\Gamma^{-1}\widetilde{W}(0)\right)\right)\exp\left(-\alpha t\right) \\ &+ 2(\widetilde{W}_{\max} + \bar{\delta}/\alpha) \end{aligned} \tag{24}$$

provided Assumption 1 is satisfied, there exists a matrix $K_1$ satisfying the BMI in (10), and agent $i$ broadcasts its state estimate $\tilde{x}_i$ as dictated by the event trigger mechanism in (23). Furthermore, $\nu$-approximate state reconstruction is achieved, where $\nu^2 \geq 2(\widetilde{W}_{\max} + \bar{\delta}/\alpha)$.

*Proof:* Let $V : \mathcal{X} \to \mathbb{R}_{\geq 0}$ be a Lyapunov function candidate, such that

$$V(\xi) \triangleq \frac{1}{2} e_1^\top e_1 + \frac{1}{2}\mathrm{tr}\,(\widetilde{W}^\top \Gamma^{-1} \widetilde{W}). \tag{25}$$

Observe that $V(\xi)$ can be bounded as

$$\frac{1}{2} e_1^\top e_1 \leq V(\xi) \leq \frac{1}{2} e_1^\top e_1 + \widetilde{W}_{\max}. \tag{26}$$

Suppose $\phi$ is a Filippov solution to the differential inclusion $\dot{\xi} \in K[h](\xi)$, i.e., $\phi = \xi$ along flows, where the mapping $K[\cdot]$ provides a calculus for computing Filippov's differential inclusion as defined in [20], and $h$ is the vector field provided in (21). The time derivative of $V(\xi)$ exists almost everywhere (a.e.) and

$$\dot{V}(\phi) \overset{a.e.}{\in} \dot{\tilde{V}}(\phi), \tag{27}$$

where $\dot{\tilde{V}}(\phi)$ is the generalized time derivative of $V(\xi)$ along the Filippov trajectories of $\dot{\phi} = h(\phi)$. By [21, Eq. 13],

$$\dot{\tilde{V}}(\phi) \triangleq \bigcap_{\eta \in \partial V(\phi)} \eta^\top \left[K[h](\phi)^\top,\ 1\right]^\top,$$

where $\partial V(\phi)$ denotes the Clarke generalized gradient of $V(\phi)$. Since $V(\phi)$ is continuously differentiable in $\phi$ during flows, $\partial V(\phi) =$

[4]The piecewise continuity of $e_{2,i}$, $\epsilon > 0$, and (23) can be used to show that, after each event time of agent $i$, there exists a well-defined time interval over which agent $i$ does not trigger.

$\{\nabla V(\phi)\}$, where $\nabla$ denotes the gradient operator. Using the calculus of $K[\cdot]$ and simplifying the substitution of (17) into the generalized time derivative of (25), one has

$$\begin{aligned}\dot{\tilde{V}}(\phi) \subseteq\ & -e_1^\top \widetilde{W}^\top K[\sigma(\widehat{\Phi}(\hat{x}))] + e_1^\top K[\chi] \\ & - \left\{e_1^\top \mathbf{K}_1 \mathbf{L} e_1\right\} - e_1^\top \mathbf{K}_1 \mathbf{L} K[e_2] \\ & - \left\{e_1^\top \mathbf{K}_1 C^\top C e_1\right\} - \operatorname{tr}(\widetilde{W}^\top \Gamma^{-1} K[\dot{\widehat{W}}]). \end{aligned} \tag{28}$$

Using the estimated outer layer weight update law in (13) for each $i \in \mathcal{V}$ and the expressions for $\widetilde{W}$, $\widehat{W}$, $e_3$, $C$, $\Gamma$, and $\sigma(\widehat{\Phi}(\hat{x}))$, it follows that

$$\begin{aligned}\operatorname{tr}(\widetilde{W}^\top \Gamma^{-1} \dot{\widehat{W}}) =\ & \sum_{i\in\mathcal{V}} \operatorname{vec}(\Gamma_i^{-1}\widetilde{W}_i)^\top \operatorname{proj}(\mu_i, \omega_i) \\ & \geq -e_3^\top C \widetilde{W}^\top \sigma(\widehat{\Phi}(\hat{x})). \end{aligned} \tag{29}$$

Substituting (5) for all $i \in \mathcal{V}$ into $e_3$ yields $e_3 = Ce_1$. Adding and subtracting $C^\top C$ while using $e_3 = Ce_1$ results in

$$\begin{aligned} e_1^\top \widetilde{W}^\top \sigma(\widehat{\Phi}(\hat{x})) =\ & e_3^\top C \widetilde{W}^\top \sigma(\widehat{\Phi}(\hat{x})) \\ & + e_1^\top (I_{nN} - C^\top C) \widetilde{W}^\top \sigma(\widehat{\Phi}(\hat{x})). \end{aligned} \tag{30}$$

Substituting (29) and (30) into (28) while using (27) yields

$$\begin{aligned}\dot{V}(\xi) \overset{a.e.}{\leq}\ & -e_1^\top \mathbf{K}_1 \mathbf{L} e_1 - e_1^\top \mathbf{K}_1 \mathbf{L} e_2 - e_1^\top \mathbf{K}_1 C^\top C e_1 \\ & + e_1^\top \widetilde{\chi}. \end{aligned} \tag{31}$$

Using Young's inequality, (31) can be upper bounded as

$$\begin{aligned}\dot{V}(\xi) \overset{a.e.}{\leq}\ & -e_1^\top \mathbf{M} e_1 + \frac{\kappa}{2}\|\mathbf{K}_1\mathbf{L}\|^2\|e_2\|^2 + \frac{1}{2\kappa}\|e_1\|^2 \\ & + \widetilde{\chi}_{\max}\|e_1\|. \end{aligned} \tag{32}$$

Using the BMI in (10) and $k_1 = k_2 + \rho^2/\delta$, (32) can be upper bounded as

$$\begin{aligned}\dot{V}(\xi) \overset{a.e.}{\leq}\ & -\left(k_2 - \frac{1}{2\kappa}\right)\|e_1\|^2 + \frac{\kappa}{2}\|\mathbf{K}_1\mathbf{L}\|^2\|e_2\|^2 \\ & + \widetilde{\chi}_{\max}\|e_1\| - \frac{\rho^2}{\delta}\|e_1\|^2. \end{aligned} \tag{33}$$

Since $\rho \geq \widetilde{\chi}_{\max}$, it follows that $\widetilde{\chi}_{\max}\|e_1\| - \rho^2\|e_1\|^2/\delta \leq \delta$. Using this inequality, (20), and the trigger function in (22), (33) can be upper bounded as

$$\dot{V}(\xi) \overset{a.e.}{\leq} -\frac{1}{2}\left(k_2 - \frac{1}{\kappa}\right)\|e_1\|^2 + \bar{\delta} - \sum_{i\in\mathcal{V}} \mathbf{T}_i(\xi). \tag{34}$$

For each $i \in \mathcal{V}$, the trigger function $\mathbf{T}_i(\xi) > 0$ during flows since agent $i$ provides state feedback according to the event trigger mechanism in (23). Hence, (34) can be upper bounded as

$$\dot{V}(\xi) \overset{a.e.}{\leq} -\frac{1}{2}\alpha\|e_1\|^2 + \bar{\delta}, \tag{35}$$

where $\alpha = k_2 - 1/\kappa$. Using (26), (35) can be bounded as

$$\dot{V}(\xi) \overset{a.e.}{\leq} -\alpha V(\xi) + \alpha \widetilde{W}_{\max} + \bar{\delta}. \tag{36}$$

Recall that $V(\xi)$ is a continuous function. Moreover, $\dot{V}(\xi)$ is continuous almost everywhere, where the set of discontinuities can be shown to have measure zero. Therefore, integrating (36) yields

$$\begin{aligned} V(\xi(t)) \leq\ & \left(\widetilde{W}_{\max} + \bar{\delta}/\alpha\right)(1 - \exp(-\alpha t)) \\ & + V(\xi(0))\exp(-\alpha t). \end{aligned} \tag{37}$$

Using (26) and (37), the result in (24) follows. The observer signals are now shown to be bounded. By Assumption 1, $x_0$ is bounded. Since $e_1$ is bounded given (24), the definition of $e_1$ implies that $e_{1,i}$ is bounded for each $i \in \mathcal{V}$. Hence, (2) implies $\hat{x}_i$ is bounded for each $i \in \mathcal{V}$. Since $\hat{x}_i$ is bounded, (3) implies $\tilde{x}_i$ is bounded. Since $\tilde{x}_i$ is bounded for each $i \in \mathcal{V}$, (9) implies $z_i$ is bounded. Since $\hat{x}_i$ is bounded and $C_i$ is fixed, $\hat{y}_i = C_i\hat{x}_i$ implies $\hat{y}_i$ is bounded. Furthermore, $y_i = C_i x_0$ implies $y_i$ is bounded. Since $\hat{y}_i$ and $y_i$ are bounded, (5) implies $e_{3,i}$ is bounded. Lastly, $\widehat{W}_i$ and $\sigma$ are bounded by construction. ■

*Theorem 2:* Let the inter-event times of agent $i$ be defined as the difference between consecutive broadcast times generated by the event trigger mechanism of agent $i \in \mathcal{V}$ in (23), i.e., $t_{k+1}^i - t_k^i$. For each $i \in \mathcal{V}$, there exists a $\delta_i > 0$ such that the inter-event times of agent $i$ are uniformly lower bounded by $\delta_i$. Specifically, $t_{k+1}^i - t_k^i \geq \delta_i$ for all $k \in \mathbb{Z}_{\geq 0}$ and $i \in \mathcal{V}$.

*Proof:* Recall that $\mathbf{T}_i(\xi) = \mathtt{C}_2\|z_i\|^2 - \mathtt{C}_1\|e_{2,i}\|^2 + \epsilon/N$, and let $\widetilde{\mathbf{T}}_i(\xi) \triangleq \epsilon/N - \mathtt{C}_1\|e_{2,i}\|^2$, where an event for agent $i$ occurs whenever $\mathbf{T}_i(\xi) = 0$. Since $\widetilde{\mathbf{T}}_i(\xi) \leq \mathbf{T}_i(\xi)$ for all $\xi$, the inter-event times produced by $\mathbf{T}_i(\xi)$ are lower bounded by those of $\widetilde{\mathbf{T}}_i(\xi)$. Therefore, it suffices to show the inter-event times of $\widetilde{\mathbf{T}}_i(\xi)$ are uniformly lower bounded away from 0. Let $\phi$ be a solution to the system $\dot{\xi} = h(\xi)$ using the event triggers $\{\widetilde{\mathbf{T}}_i(\xi)\}_{i\in\mathcal{V}}$. Furthermore, let $\mathtt{t}_k^i$ denote the $k$th event time of agent $i$ corresponding to $\widetilde{\mathbf{T}}_i(\xi)$, where $\widetilde{\mathbf{T}}_i(\phi(\mathtt{t}_k^i)) = 0$ for each $k \in \mathbb{Z}_{\geq 0}$. The moment after the $k$th event time of agent $i$, which we denote by $\mathtt{t}_k^{i+}$, the sampled state estimation error $e_{2,i}$ is reset to $0_n$, that is, $e_{2,i}(\phi(\mathtt{t}_k^{i+})) = 0_n$. Moreover, $\widetilde{\mathbf{T}}_i(\phi(\mathtt{t}_k^{i+})) = \mathfrak{h} \triangleq \epsilon/N$ for each $k \in \mathbb{Z}_{\geq 0}$, and

$$\frac{\mathtt{d}}{\mathtt{d}t}\widetilde{\mathbf{T}}_i(\xi) = -2\mathtt{C}_1\frac{\mathtt{d}}{\mathtt{d}t}\|e_{2,i}\|.$$

Substituting (9) into the time derivative of (4) yields $\dot{e}_{2,i} = -\widehat{W}_i^\top\sigma(\widehat{\Phi}_i(\hat{x}_i)) - K_1(z_i - C_i^\top e_{3,i})$. Recall that $z_i$ and $e_{3,i}$ are bounded by the proof of Theorem 1. Moreover, $\widehat{W}_i$ and $\sigma(\widehat{\Phi}_i(\hat{x}_i))$ are bounded by construction. Therefore, there exists $e_{2,i}^{\max} \in \mathbb{R}_{>0}$ such that $\|\dot{e}_{2,i}\| \leq e_{2,i}^{\max}$. Hence,

$$\frac{\mathtt{d}}{\mathtt{d}t}\|e_{2,i}\| = \frac{e_{2,i}^\top \dot{e}_{2,i}}{\|e_{2,i}\|} \leq \|\dot{e}_{2,i}\| \Rightarrow \frac{\mathtt{d}}{\mathtt{d}t}\|e_{2,i}\| \leq e_{2,i}^{\max}.$$

Letting $\mathtt{m}_i \triangleq 2\mathtt{C}_1 e_{2,i}^{\max}$, it follows that $\dot{\widetilde{\mathbf{T}}}_i(\xi) \geq -\mathtt{m}_i$. Notice that $\mathtt{t}_{k+1}^i - \mathtt{t}_k^i > 0$ since $\widetilde{\mathbf{T}}_i(\phi(\mathtt{t}_k^{i+})) = \mathfrak{h} > 0$, $\widetilde{\mathbf{T}}_i(\phi(\mathtt{t}_{k+1}^i)) = 0$, $\mathtt{t}_k^i < \mathtt{t}_k^{i+}$, and $\widetilde{\mathbf{T}}_i(\phi(t))$ is piecewise continuous. Let $\delta_i \triangleq \mathfrak{h}/\mathtt{m}_i$. Proceeding by contradiction, suppose $\mathtt{t}_{k+1}^i - \mathtt{t}_k^i < \delta_i$. The continuity of $\widetilde{\mathbf{T}}_i(\phi(s))$ over $s \in [\mathtt{t}_k^{i+}, \mathtt{t}_{k+1}^i]$ and $\mathtt{t}_k^i < \mathtt{t}_k^{i+}$ yield

$$\widetilde{\mathbf{T}}_i\left(\phi\left(\mathtt{t}_{k+1}^i\right)\right) = \mathfrak{h} + \int_{\mathtt{t}_k^{i+}}^{\mathtt{t}_{k+1}^i} \dot{\widetilde{\mathbf{T}}}_i(\phi(s))\mathtt{d}s > \mathfrak{h} - \mathtt{m}_i\left(\mathtt{t}_{k+1}^i - \mathtt{t}_k^i\right),$$

where $\mathtt{t}_{k+1}^i - \mathtt{t}_k^i < \delta_i$ implies that $\widetilde{\mathbf{T}}_i(\phi(\mathtt{t}_{k+1}^i)) > 0$. This is a contradiction. Therefore, $\mathtt{t}_{k+1}^i - \mathtt{t}_k^i \geq \delta_i$. ■

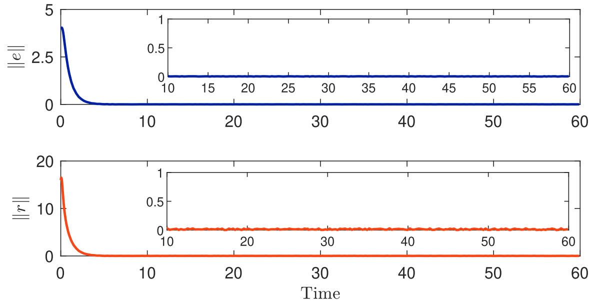


Fig. 1. Evolution of two-link robotic manipulator tracking errors for both simulations: $\|e\|$ versus time (top) and $\|r\|$ versus time (bottom). Since $e$ and $r$ converge to $0, q \to q_d$, and $\dot{q} \to \dot{q}_d$ asymptotically.

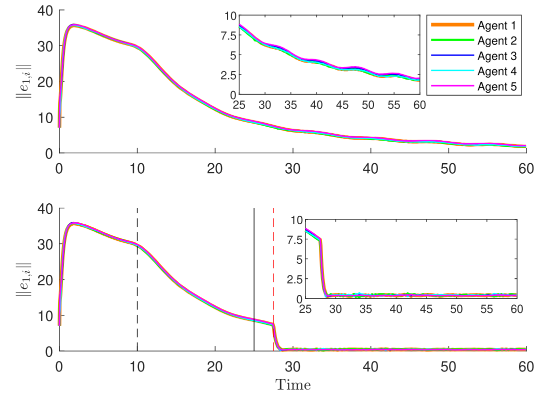


Fig. 2. Top plot illustrates the normed state estimation error $\|e_{1,i}\|$, $i \in \mathcal{V}$, versus time for the first simulation, where all initialized inner layer weights are held constant. However, for each agent, the outer layer weights are updated using (13) to ensure stability. The bottom plot shows the normed state estimation error of each agent versus time for the second simulation, where the inner layer weights are updated. For $i \in \mathcal{V}$ and at time $t = 10$, each agent begins to store their own values of $(\hat{x}_i, \dot{\hat{x}}_i)$ in memory. When $t = 25$, each agent stops collecting data and begins to train their inner layer weights using the Levenberg–Marquardt algorithm, their collected data, and (11). Training stops at $t = 27.5$ and each agent switches in their new inner layer weights. Consequently, each agent in the second simulation (bottom plot) improves their state estimation performance, where $\|e_{1,i}\| \leq 1$ for all $i \in \mathcal{V}$ and $t \in [30, 60]$.

## VII. Simulation Example

To examine the performance of the observer, two numerical simulations are provided for a two-link robotic manipulator with dynamics

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + F\dot{q} = \tau, \tag{38}$$

where $q = [q_1, q_2]^\top \in \mathbb{R}^2$ denotes the angular link position in radians, $\tau \in \mathbb{R}^2$ represents the torque input, $M(q) \in \mathbb{R}^{2\times 2}$ is the inertia matrix, $C(q,\dot{q}) \in \mathbb{R}^{2\times 2}$ is the centripetal–Coriolis matrix, and $F \in \mathbb{R}^{2\times 2}$ is a viscous friction matrix. Using [22, Sec. 4],

$$M(q) \triangleq \begin{bmatrix} p_1 + 2p_3\cos(q_2) & p_2 + p_3\cos(q_2) \\ p_2 + p_3\cos(q_2) & p_2 \end{bmatrix},$$

$$C(q,\dot{q}) \triangleq \begin{bmatrix} -p_3\sin(q_2)\dot{q}_2 & -p_3\sin(q_2)(\dot{q}_1 + \dot{q}_2) \\ p_3\sin(q_2)\dot{q}_1 & 0 \end{bmatrix},$$

$$F \triangleq \mathrm{diag}(F_1, F_2),$$

where $p_1 = 3.473$ kg·m$^2$, $p_2 = 0.193$ kg·m$^2$, $p_3 = 0.242$ kg·m$^2$, $F_1 = 5.3$ N·m·s, and $F_2 = 1.1$ N·m·s. Moreover, consider the desired trajectory $q_d \triangleq [\cos(0.5\beta), \sin(\beta)]^\top$, such that $\beta$ is the timer variable introduced in Section V.

Let $e \triangleq q_d - q \in \mathbb{R}^2$ and $r \triangleq \dot{e} + \alpha e \in \mathbb{R}^2$ with $\alpha \in \mathbb{R}_{>0}$. Leveraging exact model knowledge, the controller

$$\tau = M(q)(\ddot{q}_d + \alpha\dot{e}) + C(q,\dot{q})(\dot{q}_d + \alpha e) + F\dot{q} + k_p r - e, \; k_p \in \mathbb{R}_{>0} \tag{39}$$

renders the set $\{(e, r) = 0_4\}$ globally exponentially stable. The robotic manipulator in (38) with controller in (39) comprise the system whose state will be estimated. Since the inertia matrix is positive definite [22, Property 1], an order reduction enables the robotic manipulator dynamics to be written as (1). Let $x_0 \triangleq [x_1^\top, x_2^\top, x_3]^\top \in \mathbb{R}^5$, where $x_1 \triangleq q$, $x_2 \triangleq \dot{q}$, and $x_3 \triangleq \beta$. Then, $\dot{x}_0 = f(x_0) + d$, such that

$$f(x_0) = \begin{bmatrix} x_2 \\ -M(x_1)^{-1}(C(x_1, x_2)x_2 + Fx_2 - \tau) \\ 1 \end{bmatrix}.$$

Observe that $d = 0.15[\sin(30\beta)\cdot 1_2^\top, \sin(20\beta)\cdot 1_2^\top, 0]^\top \in \mathbb{R}^5$ is used to model process noise in both $q$ and $\dot{q}$, and the control input $\tau$ can be written as a function of $x_0$.

A five-agent sensor network is used to estimate $x_0$. The output matrix of agent $i$ is $C_i \triangleq \mathsf{e}_i^\top$, where $\mathsf{e}_i \in \mathbb{R}^5$ denotes the $i$th standard basis vector, e.g., $\mathsf{e}_2^\top = [0\ 1\ 0\ 0\ 0]$. The adjacency matrix of the sensor network and simulation parameters are

$$\mathcal{A} = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \end{bmatrix}, \quad \begin{array}{ccc} \alpha = 4 & k_p = 5 & N = 5 \\ n = 5 & m = 1 & \kappa = 1.2 \\ \delta = 1 & \ell = 5 & n_1 = 5 \\ n_2 = 10 & n_3 = 10 & n_4 = 10 \\ n_5 = 4 & n_6 = 4 & L = 4, \end{array}$$

$L_1 = 6$, $L_2 = 11$, $L_3 = 11$, $L_4 = 11$, $L_5 = 4$, $p = 4$, $\rho = 1.15$, $k_2 = 1.5$, $\epsilon = 75.3566$, and $\Gamma_i = \mathrm{diag}(0.3, 0.5, 0.2, 0.4)$. Using these parameters, $k_1 = 2.8225$, $\mathsf{c}_1 = 3.4767 \times 10^3$, $\mathsf{c}_2 = 0.0286$, and the solution to the BMI in (10) is $K_1 = 21.0371 \cdot I_5$ (computed using CVX MATLAB toolbox [23], [24]). For the projection operator parameters $\theta_0$ and $\varepsilon$ in [17, Eq. 4], $\theta_0 = 21$ and $\varepsilon = 10$. Using (11), $\phi_1(s) \triangleq [s^\top, 1]^\top$, $\phi_k(s) \triangleq [\mathrm{tansig}(s)^\top, 1]^\top$ for $k = 2, 3, 4$, and $\phi_5(s) = \sigma(s) \triangleq \mathrm{tansig}(s) \triangleq 2/(1 + \exp(-2\,s)) - 1$, which are defined elementwise for vector-valued inputs. Furthermore, $x_0(0) = [5, 5, 0, 0, 0]^\top$, and $\hat{x}_i(0) = 0_5$ for $i \in \mathcal{V}$. All DNN weights are randomly initialized as matrices with elements in [0, 0.1]. All simulations are conducted in MATLAB using the Hybrid Systems Toolbox [25]. The first simulation investigates the performance of the observer when the inner layer weights are held constant and only the outer layer weights are updated with (13). The second simulation is identical to the first with the exception that the inner layer weights are trained using the Levenberg–Marquardt algorithm [26] and collected data. The step size used to train the inner layer weights is lower bounded by $10^{-3}$. The training loss function is the mean squared error (MSE) between the estimated output generated by a known input and the corresponding known output. Each training iteration lasts until the MSE is less than $10^{-2}$ or a maximum of 30 training epochs elapse. For each training iteration, 70% of the data were used for training, 15% were used for validation, and 15% were used for testing. Figs. 1–4 and Table I summarize the results.

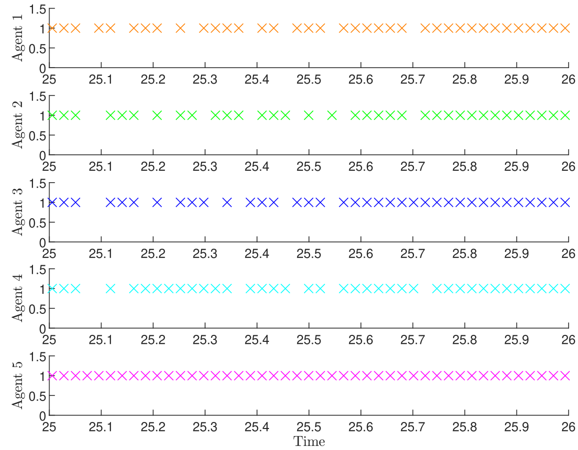


Fig. 3. With respect to the second simulation, the event times of each agent are depicted during $t \in [25, 26]$ while using (22) and (23). A $\times$ represents a communication event for an agent, and a white space corresponds to a period of noncommunication.

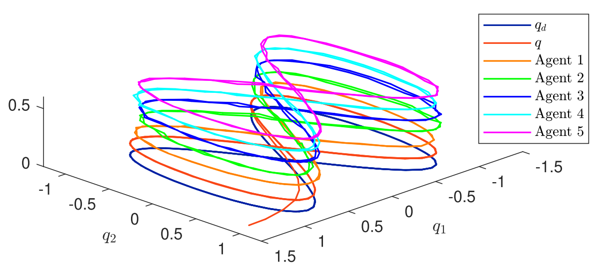


Fig. 4. Angular position trajectories at steady state ($t \geq 30$) for the two-link robotic manipulator and five-agent sensor network.

TABLE I
STATE ESTIMATION RMSE WITH AND WITHOUT DNN LEARNING

| Agent | RMSE without DNN learning | RMSE with DNN learning | Percent change |
|---|---|---|---|
| 1 | 3.7418 | 0.8464 | -77.37 |
| 2 | 3.7243 | 0.7670 | -79.40 |
| 3 | 3.8546 | 0.7740 | -79.92 |
| 4 | 3.7052 | 0.8013 | -78.37 |
| 5 | 3.9610 | 0.8353 | -78.91 |

For each agent $i \in \mathcal{V}$, the root-mean-square error (RMSE) of $\|e_{1,i}(t)\|$ during $t \in [30, 60]$ is presented for the first and second simulations, i.e., without and with training of the inner layer DNN weights.

## VIII. CONCLUSION

A distributed state observer for an uncertain nonlinear system is developed. The observer employs event-triggered communication to exchange information between agents, leverages DNNs and supervised learning to improve state estimation, provides robustness to bounded disturbances, and renders the state estimation error UUB as shown through a nonsmooth Lyapunov stability analysis. Each agent is assigned their own DNN, which they train themselves. The outer layer weights are updated with output feedback to ensure stability, while the inner layer weights are trained using the Levenberg–Marquardt algorithm with collected input-output data, i.e., the estimated state and estimated state derivative. Employing fully trained DNNs allowed each agent to reduce their state reconstruction error by approximately 77% relative to the case where the inner layer weights were held constant. Furthermore, the performance improvement only required a single training iteration, where additional training iterations can be executed as desired. A key observation is that high-quality data, such as that generated from frequent communication, leads to better training and improved state estimation. Future work includes utilizing DNNs to learn the system's dynamics and disturbance, developing distributed consensus algorithms for the DNN weights to provide model synchronization, using stochastic communication graph models, and exploring the tradeoff between triggering and state estimation using DNNs.

## APPENDIX A
## ALGORITHM FOR AGENT $i$

**Require:** $\delta, \epsilon, \kappa \in \mathbb{R}_{>0}$, $k_2 > 1/\kappa$, $\rho \geq \widetilde{\chi}_{\max}$, $N \in \mathbb{Z}_{>1}$, $n, L, \ell \in \mathbb{Z}_{>0}$, $\mathcal{L}$, $\{C_i\}_{i\in\mathcal{V}}$, $\{n_q\}_{q\in[\ell+1]} \subset \mathbb{Z}_{\geq 1}^{\ell+1}$, $\{L_q\}_{q\in[\ell]} \subset \mathbb{Z}_{\geq 1}^{\ell}$, $\{\phi_q : \mathbb{R}^{n_q} \to \mathbb{R}^{L_q}\}_{q\in[\ell]}$, $\sigma : \mathbb{R}^{n_{\ell+1}} \to \mathbb{R}^L$, $\{\widehat{W}_{q,i}(0) \in \mathbb{R}^{L_q \times n_{q+1}}\}_{q\in[\ell]}$, $\widehat{W}_i(0) \in \mathbb{R}^{L\times n}$, $\mathcal{N}_i$, $\{\tilde{x}_j(0)\}_{j\in\mathcal{N}_i\cup\{i\}} \subset \mathbb{R}^n$, $\hat{x}_i(0) \in \mathbb{R}^n$, $\Gamma_i \in \mathbb{R}^{L\times L}$, $\mathtt{dt} \in \mathbb{R}_{>0}$, $t = 0$, $k = 0$, $T_1 < T_2 \in \mathbb{R}_{>0}$, DataArray, DeployFlag $= 1$

- $\mathbf{L} \leftarrow \mathcal{L} \otimes I_n$, $C \leftarrow \mathrm{diag}(C_1, C_2, \ldots, C_N)$
- $k_1 \leftarrow k_2 + \rho^2/\delta$, $\alpha \leftarrow k_2 - 1/\kappa$
- Compute $K_1 : \frac{1}{2}\mathbf{K}_1 C^\top C + \frac{1}{2} C^\top C \mathbf{K}_1 + \mathbf{K}_1 \mathbf{L} \geq k_1 I_{nN}$
- $\mathtt{C}_1 \leftarrow k_2/2 + \kappa\|\mathbf{K}_1\mathbf{L}\|^2/2$, $\mathtt{C}_2 \leftarrow k_2/(4\|\mathbf{L}\|^2)$
- **while** True **do**
  - Measure $y_i$, $\hat{y}_i \leftarrow C_i\hat{x}_i$, $e_{3,i} \leftarrow \hat{y}_i - y_i$
  - $z_i \leftarrow \sum_{j\in\mathcal{V}} a_{ij}(\tilde{x}_j - \tilde{x}_i)$, $\omega_i \leftarrow \mathrm{vec}(\widehat{W}_i)$
  - $\widehat{\Phi}_i(\hat{x}_i) \leftarrow (\widehat{W}_{\ell,i}^\top \phi_\ell \circ \widehat{W}_{\ell-1,i}^\top \phi_{\ell-1} \circ \ldots \circ \widehat{W}_{1,i}^\top \phi_1)(\hat{x}_i)$
  - $\dot{\hat{x}}_i \leftarrow \widehat{W}_i^\top \sigma(\widehat{\Phi}_i(\hat{x}_i)) + K_1(z_i - C_i^\top e_{3,i})$
  - $\dot{\omega}_i \leftarrow \mathrm{proj}(-\mathrm{vec}(\Gamma_i \sigma(\widehat{\Phi}_i(\hat{x}_i)) e_{3,i}^\top C_i), \omega_i)$
  - $\hat{x}_i \leftarrow \hat{x}_i + \mathtt{dt}\cdot\dot{\hat{x}}_i$, $\omega_i \leftarrow \omega_i + \mathtt{dt}\cdot\dot{\omega}_i$
  - $t \leftarrow t + \mathtt{dt}$, $k \leftarrow k + 1$
  - Reshape $\omega_i$: $\widehat{W}_i \leftarrow \mathrm{vec}^{-1}(\omega_i)$
  - $e_{2,i} \leftarrow \tilde{x}_i - \hat{x}_i$
  - **if** $\mathtt{C}_2\|z_i\|^2 - \mathtt{C}_1\|e_{2,i}\|^2 + \epsilon/N = 0$ **then**
    - Broadcast $\hat{x}_i$, $\tilde{x}_i \leftarrow \hat{x}_i$
  - **end if**
  - **if** agent $j \in \mathcal{N}_i$ broadcasts **then**
    - $\tilde{x}_j \leftarrow \hat{x}_j$
  - **end if**
  - **if** $t < T_1$ **then**
    - DataArray$(k) \leftarrow (\hat{x}_i, \dot{\hat{x}}_i)$
  - **end if**
  - **if** $T_1 \leq t \leq T_2$ **then**
    - Train inner DNN weights using Levenberg–Marquardt algorithm, DataArray, and $\{\phi_q\}_{q\in[\ell]}$. Store new inner DNN weights in $\{\widehat{W}_{q,i}^{\text{new}}\}_{q\in[\ell]}$
  - **end if**
  - **if** $t > T_2$ and DeployFlag $= 1$ **then**
    - DeployFlag $\leftarrow 0$
    - **for** $q = 1 : \ell$ **do**
      - $\widehat{W}_{q,i} \leftarrow \widehat{W}_{q,i}^{\text{new}}$
    - **end for**
  - **end if**
- **end while**

## Acknowledgment

Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of sponsoring agencies.

## References

[1] H. Mostafaei, "Energy-efficient algorithm for reliable routing of wireless sensor networks," *IEEE Trans. Ind. Electron.*, vol. 66, no. 7, pp. 5567–5575, Jul. 2018.

[2] X. Ge, Q.-L. Han, X.-M. Zhang, L. Ding, and F. Yang, "Distributed event-triggered estimation over sensor networks: A survey," *IEEE Trans. Cybern.*, vol. 50, no. 3, pp. 1306–1320, Mar. 2019.

[3] X. He, W. Xue, and H. Fang, "Consistent distributed state estimation with global observability over sensor network," *Automatica*, vol. 92, pp. 162–172, 2018.

[4] F. M. Zegers, D. P. Guralnik, and W. E. Dixon, "Event/self-triggered multi-agent system rendezvous with graph maintenance," in *Proc. IEEE Conf. Decis. Control*, 2021, pp. 1886–1891.

[5] F. M. Zegers, P. Deptula, J. M. Shea, and W. E. Dixon, "Event/self-triggered approximate leader-follower consensus with resilience to Byzantine adversaries," *IEEE Trans. Autom. Control*, vol. 67, no. 3, pp. 1356–1370, Mar. 2022.

[6] G. Battistelli, L. Chisci, and D. Selvi, "A distributed Kalman filter with event-triggered communication and guaranteed stability," *Automatica*, vol. 93, pp. 75–82, 2018.

[7] X. Ge, Q.-L. Han, and Z. Wang, "A threshold-parameter-dependent approach to designing distributed event-triggered $H_\infty$ consensus filters over sensor networks," *IEEE Trans. Cybern.*, vol. 49, no. 4, pp. 1148–1159, Apr. 2018.

[8] D. Yu, Y. Xia, L. Li, and D.-H. Zhai, "Event-triggered distributed state estimation over wireless sensor networks," *Automatica*, vol. 118, 2020, Art. no. 109039.

[9] G. Battistelli and L. Chisci, "Stability of consensus extended Kalman filter for distributed state estimation," *Automatica*, vol. 68, pp. 169–178, 2016.

[10] D. Ding, Z. Wang, and Q.-L. Han, "A set-membership approach to event-triggered filtering for general nonlinear systems over sensor networks," *IEEE Trans. Autom. Control*, vol. 65, no. 4, pp. 1792–1799, Apr. 2019.

[11] Q. Li, B. Shen, Z. Wang, and W. Sheng, "Recursive distributed filtering over sensor networks on Gilbert–Elliott channels: A dynamic event-triggered approach," *Automatica*, vol. 113, 2020, Art. no. 108681.

[12] G. Joshi and G. Chowdhary, "Deep model reference adaptive control," in *Proc. IEEE Conf. Decis. Control*, 2019, pp. 4601–4608.

[13] G. Joshi, J. Virdi, and G. Chowdhary, "Design and flight evaluation of deep model reference adaptive controller," in *Proc. AIAA Scitech Forum*, 2020, Art. no. 1336.

[14] R. Sun, M. L. Greene, D. M. Le, Z. I. Bell, G. Chowdhary, and W. E. Dixon, "Lyapunov-based real-time and iterative adjustment of deep neural networks," *IEEE Control Syst. Lett.*, vol. 6, pp. 193–198, Jan. 2022.

[15] W. Rudin, *Principles of Mathematical Analysis*, 3rd ed. New York, NY, USA: McGraw-hill1976.

[16] G. Joshi, J. Virdi, and G. Chowdhary, "Asynchronous deep model reference adaptive control," in *Proc. PMLR Conf. Robot Learn. Virtual*, 2020, pp. 984–1000.

[17] Z. Cai, M. S. de Queiroz, and D. M. Dawson, "A sufficiently smooth projection operator," *IEEE Trans. Autom. Control*, vol. 51, no. 1, pp. 135–139, Jan. 2006.

[18] C. Li and Z. Qu, "Distributed estimation of algebraic connectivity of directed networks," *Syst. Control Lett.*, vol. 62, no. 6, pp. 517–524, 2013.

[19] J. Lu, C. Y. Tang, P. R. Regier, and T. D. Bow, "Gossip algorithms for convex consensus optimization over networks," *IEEE Trans. Autom. Control*, vol. 56, no. 12, pp. 2917–2923, Dec. 2011.

[20] B. Paden and S. Sastry, "A calculus for computing Filippov's differential inclusion with application to the variable structure control of robot manipulators," *IEEE Trans. Circuits Syst.*, vol. CS-34, no. 1, pp. 73–82, Jan. 1987.

[21] D. Shevitz and B. Paden, "Lyapunov stability theory of nonsmooth systems," *IEEE Trans. Autom. Control*, vol. 39, no. 9, pp. 1910–1914, Sep. 1994.

[22] W. E. Dixon, M. S. de Queiroz, F. Zhang, and D. M. Dawson, "Tracking control of robot manipulators with bounded torque inputs," *Robotica*, vol. 17, no. 2, pp. 121–129, 1999.

[23] M. Grant and S. Boyd, "Graph implementations for nonsmooth convex programs," in *Recent Advances in Learning and Control* (Lecture Notes in Control and Information Sciences), V. Blondel, S. Boyd, and H. Kimura, Eds. Berlin, Germany: Springer, 2008, pp. 95–110.

[24] M. Grant and S. Boyd, "CVX: MATLAB software for disciplined convex programming, version 2.1," 2014. [Online] Available http://cvxr.com/cvx

[25] R. Sanfelice, D. Copp, and P. Ñañez, "A toolbox for simulation of hybrid systems in MATLAB/Simulink: Hybrid equations (HyEQ) toolbox," in *Proc. Intern. Conf. Hybrid Syst, Comput. Control*, 2013, pp. 101–106.

[26] M. I. Lourakis et al., "A brief description of the Levenberg-Marquardt algorithm implemented by levmar," *Found. Res. Tech.*, vol. 4, no. 1, pp. 1–6, 2005.