2018 IEEE Conference on Decision and Control (CDC)
Miami Beach, FL, USA, Dec. 17-19, 2018

Network Identification: A Passivity and Network Optimization Approach

Miel Sharf and Daniel Zelazo

Abstract— **The theory of network identification, namely identifying the interaction topology among a known number of agents, has been widely developed for linear agents over recent years. However, the theory for nonlinear agents remains less extensive. We use the notion maximal equilibrium-independent passivity (MEIP) and network optimization theory to present a network identification method for nonlinear agents. We do so by introducing a specially designed exogenous input, and exploiting the properties of networked MEIP systems. We then specialize on LTI agents, showing that the method gives a distributed cubic-time algorithm for network reconstruction in that case. We also discuss different methods of choosing the exogenous input, and provide an example on a neural network model.**

I. Introduction

Multi-agent systems have been widely studied in recent years, as they present both a variety of applications and a deep theoretical framework. They have been employed across numerous domains, including flocking, formation control, robotics rendezvous, social networks, and distributed estimation [1], [2]. One of the most important aspects in multi-agents systems, both in theory and in practice, is the information-exchange layer, governing which agents interact with each other. Identifying the underlying network of a multi-agent system from measurements is of great importance in many applications. One example is systems biology, in which measurements are used to understand the connection between genes in regulatory networks [3]. Another example is international finance, in which past exchange rates between different currencies are used to determine their influence on one another, giving a useful guide for understanding the causal relationship between individual currencies [4]. Other fields with similar problems include social networks [5], neuroscience [6], [7], communication networks [8] and ecology [9].

The problem of network identification has been widely studied for linear agents. Seminal works dealing with network identification include [10], providing exact reconstruction for tree-like graphs, and [11] in which sparse enough topologies can be identified from a small number of observations. Other important works include [12], using a node knockout method, and [13], presenting a sieve method for solving the network identification problems for consensus-seeking networks. More recent methods include auto-regressive models [14] and spectral methods [15]. However, a theory for network identification for interacting nonlinear agents is far less developed. We aim to provide in this work a network identification scheme for a wide range of systems, including nonlinear ones. Our approach relies on a concept widespread in multi-agent systems, namely passivity theory.

M. Sharf and D. Zelazo are with the Faculty of Aerospace Engineering, Israel Institute of Technology, Haifa, Israel. `msharf@tx.technion.ac.il`, `dzelazo@technion.ac.il`. This work was supported by the German-Israeli Foundation for Scientific Research and Development.

Passivity theory is a cornerstone of the theoretical frame work of networks of dynamical systems [16]. The main reason is that it allows for the analysis of multi-agent systems to be decoupled into two separate layers, the dynamic system layer and the information exchange layer. Passivity theory was first used to study the convergence properties of network systems in [17]. Many variations and extensions of passivity have been applied in different aspects of multi-agent systems. For example, the related concepts of incremental passivity or relaxed co-coercivity have been used to study various synchronization problems [18], [19], and more general frameworks including Port-Hamiltonian systems on graphs [20].

One prominent variant is maximal equilibrium-independent passivity (MEIP), which was applied in [21] in order to reinterpret the analysis problem for multi-agent system as a network optimization problem. Network optimization is a branch of optimization theory dealing with optimization of functions defined over graphs [22]. The main result of [21] showed that the asymptotic behavior of these networked systems is *(inverse) optimal* with respect to a family of network optimization problems. In fact, the steady-state input-output signals of both the dynamical systems and the controllers comprising the networked system can be associated to the optimization variables of either an *optimal flow* or an *optimal potential* problem; these are the two canonical dual network optimization problems described in [22]. The results of [21] were used in [23], [24] in order to solve the synthesis problem for multi-agent systems.

We aim to use this network optimization framework to provide a network identification scheme for multi-agent systems. We do so by injecting a constant exogenous output, and tracking the output of the agents. By appropriately designing the exogenous input, we are able to differentiate the outputs of the closed-loop system associated to different underlying graphs. The key idea in the proof is that the steady-state outputs are solutions to network optimization problems and they are one-to-one dependent on the exogenous input. Our contributions are stated as follows:

i) We introduce the notion of *indication vectors* for MEIP systems that are used for differentiating the output of networked systems with different underlying graphs.
ii) We propose various methods for constructing these indication vectors.

iii) We propose an algorithm exploiting the notion of indication vectors to *solve* a network detection problem.
iv) We show that in the case of linear time-invariant (LTI) systems, our solution gives a distributed $O(n^3)$ network detection algorithm, where n is the number of agents.

The rest of the paper is organized as follows. Section II surveys the relevant parts of the network optimization framework. Section III presents the problem formulation. Section IV presents the main technical tool used for building the network detection schemes, namely indication vectors, and shows different methods of constructing indication vectors. Section V uses indication vectors to design a network detection scheme for general MEIP agents. Lastly, we present a case study simulating the network detection methods discussed on a neural network.

Notations: We use basic notations from linear algebra. For a linear map $T : U \to V$ between vector spaces, we denote the kernel of T by $\ker T$, and the image of T by $\mathrm{Im}(T)$. Furthermore, if U is a subspace of an inner-product space X (e.g., $\mathbb{R}^d$), we denote the orthogonal complement of U by $U^\perp$. The notation $A \geq 0$ ($A > 0$) means the matrix A is positive semi-definite (positive definite). We also use basic notions from algebraic graph theory [25]. An undirected graph $\mathcal{G} = (\mathbb{V}, \mathbb{E})$ consists of a finite set of vertices $\mathbb{V}$ and edges $\mathbb{E} \subset \mathbb{V} \times \mathbb{V}$. We denote by $k = \{i, j\} \in \mathbb{E}$ the edge that has ends i and j in $\mathbb{V}$. For each edge k, we pick an arbitrary orientation and denote $k = (i, j)$. The incidence matrix of $\mathcal{G}$, denoted $\mathcal{E}_\mathcal{G} \in \mathbb{R}^{|\mathbb{E}|\times|\mathbb{V}|}$, is defined such that for edge $k = (i, j) \in \mathbb{E}$, $[\mathcal{E}_\mathcal{G}]_{ik} = +1$, $[\mathcal{E}_\mathcal{G}]_{jk} = -1$, and $[\mathcal{E}_\mathcal{G}]_{\ell k} = 0$ for $\ell \neq i, j$.

II. Network Optimization and MEIP Multi-Agent Systems

The role of network optimization theory in cooperative control was introduced in [21], and was used in [23], [24] to solve the synthesis problem for multi-agent systems. In this section, we provide an overview of the main results from these works.

A. The Closed-Loop and Steady-States

Consider a collection of agents interacting over a network $\mathcal{G} = (\mathbb{V}, \mathbb{E})$. Assign to each node $i \in \mathbb{V}$ (the agents) and each edge $e \in \mathbb{E}$ (the controllers) the dynamical systems,

$$\Sigma_i : \begin{cases} \dot{x}_i = f_i(x_i, u_i) \\ y_i = h_i(x_i, u_i) \end{cases}, \Pi_e : \begin{cases} \dot{\eta}_e = \phi_e(\eta_e, \zeta_e) \\ \mu_e = \psi_e(\eta_e, \zeta_e) \end{cases}. \quad (1)$$

We consider stacked vectors of the form $u = [u_1^T, \dots, u_{|\mathbb{V}|}^T]^T$ and similarly for y, ζ and μ and the operators Σ and Π. The network system is diffusively coupled with the controller input described by $\zeta = \mathcal{E}_\mathcal{G}^T y$, and the control input to each system by $u = -\mathcal{E}_\mathcal{G}\mu$. This structure is illustrated in Fig. 1 and we denote the closed-loop system above by the triple $(\mathcal{G}, \Sigma, \Pi)$.

Of interest for these systems are the steady-state solutions, if they exist, of the closed-loop. Suppose that $(\mathrm{u}, \mathrm{y}, \zeta, \mu)$ is a steady-state of the system. Then $(\mathrm{u}_i, \mathrm{y}_i)$ is a steady-state input-output pair of the i-th agent, and (ζ_e, μ_e) is a steady-state pair of the e-th edge. This motivates the following definition, originally introduced in [21].

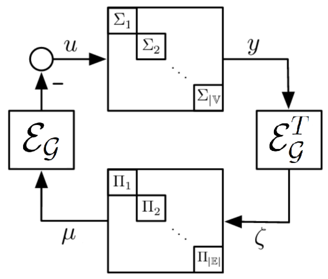


Fig. 1. Block-diagram of the closed loop.

Definition 1. *The* steady-state input-output relation k *of a dynamical system is the collection of all steady-state input-output pairs of the system. Given a steady-state input* u *and a steady-state* y*, we define*

$$k(\mathrm{u}) = \{\mathrm{y} : (\mathrm{u}, \mathrm{y}) \in k\} \textit{ and } k^{-1}(\mathrm{y}) = \{\mathrm{u} : (\mathrm{u}, \mathrm{y}) \in k\}.$$

Let k_i be the steady-state input-output relation for the i-th agent, γ_e be the steady-state input-output relation for the e-th controller, and k, γ be their stacked versions. Then, the network interconnection shown in Fig.1 imposes on the closed-loop steady-states $(\mathrm{u}, \mathrm{y}, \zeta, \mu)$ that $\mathrm{y} \in k(\mathrm{u})$, $\zeta = \mathcal{E}_\mathcal{G}^T \mathrm{y}$, $\mu \in \gamma(\zeta)$, and $\mathrm{u} = -\mathcal{E}_\mathcal{G}\mu$. Equivalently stated, y is a steady-state for the system $(\mathcal{G}, \Sigma, \Pi)$ if and only if $0 \in k^{-1}(\mathrm{y}) + \mathcal{E}_\mathcal{G}\gamma(\mathcal{E}_\mathcal{G}^T \mathrm{y})$. The above expression summarizes both the dynamic and algebraic constraints that must be satisfied by the network system to achieve a steady-state solution.

B. MEIP Systems and Convergence of the Closed-Loop

Convergence of the system $(\mathcal{G}, \Sigma, \Pi)$ can be guaranteed under a passivity assumption on the agent and controller dynamics [21].

Definition 2 (Maximal Equilibrium Independent Passivity [21])**.** *Consider the dynamical system of the form*

$$\Upsilon : \begin{cases} \dot{x} = f(x, u) \\ y = h(x, u), \end{cases} \quad (2)$$

with steady-state input-output relation r. The system Υ is said to be (output-strictly) maximal equilibrium independent passive *(MEIP) if the following conditions hold:*
i) *The system Υ is (output-strictly) passive with respect to any steady state pair $(\mathrm{u}, \mathrm{y}) \in r$.*
ii) *The relation r is maximally monotone. That is, if $(\mathrm{u}_1, \mathrm{y}_1), (\mathrm{u}_2, \mathrm{y}_2) \in r$ then either $(\mathrm{u}_1 \leq \mathrm{u}_2$ and $\mathrm{y}_1 \leq \mathrm{y}_2)$, or $(\mathrm{u}_1 \geq \mathrm{u}_2$ and $\mathrm{y}_1 \geq \mathrm{y}_2)$, and r is not contained in any larger monotone relation [26].*

Such systems include simple integrators, gradient systems, Hamiltonian systems on graphs, and others (see [21], [24] for more examples). We remark that the monotonicity requirement is used to prove existence of a closed-loop steady-state, see [21] or [24] for more details.

Theorem 1 ([21], [23]). *Consider the closed-loop system $(\mathcal{G},\Sigma,\Pi)$. Assume that the agents Σ_i are MEIP, and that the agents Π_e are output-strictly MEIP. Then the signals u, y, ζ, μ of the closed-loop system converge to some steady-state values $\mathrm{u}, \mathrm{y}, \zeta, \mu$ satisfying $0 \in k^{-1}(\mathrm{y}) + \mathcal{E}_{\mathcal{G}}\gamma(\mathcal{E}_{\mathcal{G}}^T \mathrm{y})$.*

III. Motivation and Problem Formulation

The problem of network identification we aim to solve can be stated as follows. Given a multi-agent system $(\mathcal{G},\Sigma,\Pi)$, determine the underlying graph structure $\mathcal{G}$ from the network measurements and an appropriately designed exogenous input w. Many works on network identification consider networks of consensus-seeking agents [12], [13],

$$\dot{x}_i = \sum_{\{i,j\}\in\mathbb{E}} \alpha_{ij}(x_j - x_i) + B_i w_i, \tag{3}$$

where w_i is the controlled exogenous input for the i-th agent, and $\alpha_{ij} = \alpha_{ji}$ are the coupling coefficients. We consider a more general case of (possibly nonlinear) agents interacting over a modified protocol,

$$\dot{x}_i = f_i(x_i) + \sum_{\{i,j\}\in\mathbb{E}} \alpha_{ij} g_{ij}(h_j(x_j) - h_i(x_i)) + B_i w_i, \tag{4}$$

where $x_i \in \mathbb{R}$, and $f_i, g_{ij}, h_i : \mathbb{R} \to \mathbb{R}$ are smooth functions. [1] Examples of systems governed by (4), for appropriate choice of functions f_i, g_{ij}, h_i, include traffic control models [27], neural networks [28], and the Kuramoto model for synchronizing oscillators [29]. We let f, g, h denote the stacked versions of f_i, g_{ij}, h_i.

In this work, we shall restrict ourselves to the case of $\alpha_{ij} = 1$, i.e., of unweighted graphs [12]. Furthermore, in the model (3), the standard assumption is that only certain agents can be controlled using the exogenous input w_i (i.e., $B_i = 0$ is possible), and one can observe the outputs of only certain agents. To simplify the presentation, we assume that the exogenous output w_i can be added to all agents, and that the output of all agents can be observed. In that case, we can assume without loss of generality that $B_i = 1$.

We note that the system (4) is a special case of the closed-loop presented in Fig. 1, where the agents and the controllers are given by

$$\Sigma_i : \begin{cases} \dot{x}_i &= f_i(x_i) + u_i + w_i \\ y_i &= h_i(x_i) \end{cases}, \ \ \Pi_{ij} : \zeta_{ij} = g_{ij}(\mu_{ij}), \tag{5}$$

and the network is connected using the diffusive coupling $\zeta = \mathcal{E}_{\mathcal{G}}^T y$ and $u = -\mathcal{E}_{\mathcal{G}}\mu$. We would like to use the mechanisms presented in Section II to establish network identification results. We make the following assumptions on the agents and controllers, allowing us to use the framework presented in section II. With this model, we will often write the closed-loop as $(\mathcal{G},\Sigma,g)$.

Assumption 1. *The systems Σ_i, for all $i \in \mathbb{V}$, are output-strictly MEIP. Furthermore, the controllers Π_e, for all $e \in \mathbb{E}$, are MEIP, i.e., g_{ij} are monotone ascending functions.*

[1]The functions g_{ij} are defined for all pairs, even those absent from the underlying graph. It is often assumed in multi-agent systems that each agent knows to run a given protocol (i.e., consensus).

Assumption 2. *The inverse of the steady-state input-output relation for each agent, $k_i^{-1}(\mathrm{y}_i)$, is a smooth function of y_i. Furthermore, we assume that $g_{ij}(\zeta_{ij})$ is a smooth function of ζ_{ij}, and that the derivative $\frac{dg_{ij}}{d\zeta_{ij}} > 0$ for all $\zeta_{ij} \in \mathbb{R}$.*

Assumption 2 implies that the integral function $K_i^\star$ associated with k_i^{-1} [21] is smooth and $\nabla K_i^\star = k_i^{-1}$. The assumption on g_{ij} implies that g_{ij} is strictly monotone ascending, and the stronger assumption is made mainly to avoid heavy technical tools.

We will also consider the special case where the agents and controllers are described by linear and time-invariant (LTI) dynamics. For such systems, the input-output relation k_i for each agent is linear and strictly monotone, and so is the function g_{ij}. When Σ_i is an integrator, the input-output relation is given as $\{(0, \mathrm{y}) :\ \mathrm{y} \in \mathbb{R}\}$. In these cases, k_i^{-1} is a linear function over $\mathbb{R}$. In particular, $k_i^{-1}(\mathrm{x}_i) = a_i \mathrm{x}_i$ for some constant $a_i \geq 0$. We can then define the matrix $A = \mathrm{diag}(a_1, \dots, a_n)$ such that $k^{-1}(\mathrm{x}) = A\mathrm{x}$. Similarly, we denote $g_{ij}(\mathrm{x}_{ij}) = b_{ij}\mathrm{x}_{ij}$, where $b_{ij} > 0$, and $B = \mathrm{diag}(\cdots, b_{ij}, \cdots) > 0$.

We can now formulate two fundamental problems of network detection that we will consider.

Problem 1. *Consider the network system $(\mathcal{G},\Sigma,\Pi)$ of the form (4) satisfying Assumptions 1 and 2 with known steady-state input-output relations for the agents and controllers. Design the control inputs w_i so that it is possible to differentiate the network system $(\mathcal{G},\Sigma,\Pi)$ from the network system $(\mathcal{H},\Sigma,\Pi)$, when $\mathcal{H} \neq \mathcal{G}$.*

Problem 2. *Consider the network system $(\mathcal{G},\Sigma,\Pi)$ of the form (4) satisfying Assumptions 1 and 2 with known steady-state input-output relations for the agents and controllers, but unknown network structure $\mathcal{G}$. Design the control inputs w_i such that together with the output measurements of the network, it is possible to reconstruct the graph $\mathcal{G}$.*

We aim for a solution of both problems, starting with the Problem 1 in the sequel. We will later show how to augment the algorithm solving Problem 1 to solve the harder problem Problem 2.

Note the framework developed in [21] requires constant signals for exogoneous inputs. Thus, we will consider constant w_i, and denote them as w_i. For similar reasons, we can only consider the output measurements of the system in steady-state when reconstructing the graph $\mathcal{G}$. However, in practice we may not be able to wait for the system to converge, and can only know the terminal state up to some approximation error. We will deal with this issue by giving a bound on the error one can tolerate.

IV. Distinguishing Between Different Networks

In this section, we develop the notion of indication vectors used for solving Problem 1, and provide different methods for constructing them.

A. The Basic Equation and Indication Vectors

We consider a network system of the form (5). We first study constant exogenous input vectors w that can differentiate between two different network systems $(\mathcal{G}, \Sigma, \Pi)$ and $(\mathcal{H}, \Sigma, \Pi)$. In this direction, we provide a result relating the constant exogenous inputs w to the network steady-states.

Proposition 1. *Under Assumptions 1 and 2, for any $\mathrm{w} \in \mathbb{R}^n$, the vector $\mathrm{y} \in \mathbb{R}^n$ is a steady-state of the closed loop system $(\mathcal{G}, \Sigma, \Pi)$ if and only if*

$$k^{-1}(\mathrm{y}) + \mathcal{E}_\mathcal{G} g\left(\mathcal{E}_\mathcal{G}^T \mathrm{y}\right) = -\mathrm{w}. \tag{6}$$

Proof. The result follows directly from Theorem 1 using $\gamma(\zeta) = g(\zeta)$, $k^{-1}(\mathrm{y}) = \mathrm{u} + \mathrm{w}$ and the network connection $\zeta = \mathcal{E}^T y,\ u = -\mathcal{E}\mu$. We note that this is an equality and not inclusion due to Assumption 2. □

We denote a solution of (6) as $\mathrm{y}_\mathcal{G}$. Furthermore, we note that (6) is graph dependent, and that the steady-state output $\mathrm{y}_\mathcal{G}$ can be measured from the network (as the network converges to a steady-state by Theorem 1). The idea now is to choose the bias vectors w wisely so that different graphs will have different terminal outputs.

Definition 3. *Consider a closed-loop system of the form* (4) *satisfying Assumptions 1 and 2, where the controllers g_{ij} have been determined for all possible pairs $\{i, j\}$. Let $\mathfrak{G}$ be a collection of graphs over n vertices. A vector $\mathrm{w} \in \mathbb{R}^n$ is called a $\mathfrak{G}$*-indication vector *if for any two graphs $\mathcal{G}, \mathcal{H} \in \mathfrak{G}$ with $\mathcal{G} \neq \mathcal{H}$, the steady-state output of $(\mathcal{G}, \Sigma, g)$ is different from the steady-state output of $(\mathcal{H}, \Sigma, g)$. In other words, one has that $\mathrm{y}_\mathcal{G} \neq \mathrm{y}_\mathcal{H}$.*

Assume for now that the agents and controllers are LTI. We can now restate (6) in a new manner, involving linear inequalities. In turn, this will manifest in stronger results later. The following result will be useful in the analysis.

Proposition 2. *If $A \neq 0$, then for any connected graph $\mathcal{G}$, the matrix $S = A + \mathcal{E}_\mathcal{G} B \mathcal{E}_\mathcal{G}^T$ is invertible.*

Proof. Note that $\mathcal{E}_\mathcal{G} B \mathcal{E}_\mathcal{G}^T \geq 0$ and that $A \geq 0$, implying that $S \geq 0$. Furthermore, the kernel of $\mathcal{E}_\mathcal{G} B \mathcal{E}_\mathcal{G}^T$ consists solely of the span of the all-ones vector, $\mathbb{1}$. It follows that $\mathbb{1}^T A \mathbb{1} = \sum_{i=1}^{|\mathbb{V}|} a_i > 0$, completing the proof. □

Proposition 2 allows an explicit form for (6) for the case $A \neq 0$ by inverting the matrix in question,

$$\mathrm{y}_\mathcal{G} = -(A + \mathcal{E}_\mathcal{G} B \mathcal{E}_\mathcal{G}^T)^{-1}\mathrm{w} = -X_{\mathcal{G}, A\neq 0}\mathrm{w}.$$

If $A = 0$, however, we note that for any $\mathcal{G} \in \mathfrak{G}$, the linear operator $\mathcal{E}_\mathcal{G} B \mathcal{E}_\mathcal{G}^T$ preserves $\mathrm{Im}(\mathcal{E}_\mathcal{G}) = \mathbb{1}^\perp$, and moreover, it is invertible when restricted to it. Thus, we denote the restriction of $\mathcal{E}_\mathcal{G} B \mathcal{E}_\mathcal{G}^T$ on $\mathbb{1}^\perp$ by $Y_\mathcal{G}$, and obtain

$$\mathrm{y}_\mathcal{G} = -Y_\mathcal{G} \mathrm{Proj}_{\mathbb{1}^\perp}\mathrm{w} = -X_{\mathcal{G}, A=0}\mathrm{w}.$$

These linear relations between the steady-state output $\mathrm{y}_\mathcal{G}$ and the constant exogenous input w allows for an easier statement of the definition of indication vectors.

Proposition 3. *For LTI agents and controllers, and for a vector $\mathrm{w} \in \mathbb{R}^n$, the following statements hold:*

i) *If $A \neq 0$, w is a $\mathfrak{G}$-indication vector if and only if for any two different graphs $\mathcal{G}, \mathcal{H} \in \mathfrak{G}$, we have $X_{\mathcal{G}, A\neq 0}\mathrm{w} \neq X_{\mathcal{H}, A\neq 0}\mathrm{w}$.*
ii) *If $A = 0$, w is a $\mathfrak{G}$-indication vector if and only if any two different graphs $\mathcal{G}, \mathcal{H} \in \mathfrak{G}$, we have $X_{\mathcal{G}, A=0}\mathrm{w} \neq X_{\mathcal{H}, A=0}\mathrm{w}$.*

We will use $X_\mathcal{G}$ for notational simplicity. We first note the following interesting property of $X_\mathcal{G}$.

Proposition 4. *If $\mathcal{G} \neq \mathcal{H}$ then $X_\mathcal{G} \neq X_\mathcal{H}$.*

Proof. Suppose first that $A \neq 0$. We can reconstruct the weighted graph Laplacian $\mathcal{E}_\mathcal{G} B \mathcal{E}_\mathcal{G}^T$ from $X_\mathcal{G}$ using the relation $\mathcal{E}_\mathcal{G} B \mathcal{E}_\mathcal{G}^T = -A + X_\mathcal{G}^{-1}$, thus $X_\mathcal{G} = X_\mathcal{H}$ implies $\mathcal{G} = \mathcal{H}$, as $B > 0$. If $A = 0$, we note that $Y_\mathcal{G} = -X_\mathcal{G}^{-1}$ on the set $\mathbb{1}^\perp$. This determines the graph Laplacian, as it is the projection of the weighted graph Laplacian on $\mathbb{1}^\perp = \ker(\mathcal{E}_\mathcal{G} B \mathcal{E}_\mathcal{G}^T)^\perp$. □

After restating the definition of indication vectors for LTI systems, we return to the case of general agents and controllers satisfying Assumptions 1 and 2. Given an indication vector w, we can quantify how much it can differentiate between different graphs. We do so with the following definition.

Definition 4. *The* separation index *of w, denoted $\varepsilon = \varepsilon(\mathrm{w})$, is defined as the minimal distance between $\mathrm{y}_\mathcal{G}$ and $\mathrm{y}_\mathcal{H}$ where $\mathcal{G} \neq \mathcal{H}$, i.e., $\varepsilon = \min_{\mathcal{G}\neq\mathcal{H}} \|\mathrm{y}_\mathcal{G} - \mathrm{y}_\mathcal{H}\|$, where the minimization is over all graphs in $\mathfrak{G}$.*

Remark 1. *The separation index ε acts as a bound on the error we can tolerate when computing the steady-state output of the closed-loop system. This error can be comprised of both numerical errors, as well as errors arising from early termination of the system (i.e., before reaching steady-state). Indeed, suppose we want to differentiate between $\mathcal{G}, \mathcal{H}$. We know that $\|\mathrm{y}_\mathcal{G} - \mathrm{y}_\mathcal{H}\| \geq \varepsilon$. Suppose we have the terminal output y of the closed-loop system for either $\mathcal{G}$ or $\mathcal{H}$. By the triangle inequality, if $\|\mathrm{y} - \mathrm{y}_\mathcal{G}\| < 0.5\varepsilon$ then $\|\mathrm{y} - \mathrm{y}_\mathcal{H}\| \geq 0.5\varepsilon$ and vice versa. Thus, if we know that the sum of the numerical and early termination errors is less than 0.5ε, we can correctly choose the underlying graph by choosing which of $\mathrm{y}_\mathcal{G}$ and $\mathrm{y}_\mathcal{H}$ is closer to y.*

Remark 2. *Consider the case of LTI agents and controllers. In that case, Proposition 3 implies that the relation between $\mathrm{y}_\mathcal{G}$ and w is linear. Thus, for any constant $\beta > 0$, the separation index satisfies $\varepsilon(\beta\mathrm{w}) = \beta\varepsilon(\mathrm{w})$.*

B. Methods of Choice for Indication Vectors

We now explore how to construct indication vectors for a multi-agent system of the form (4) satisfying Assumptions 1 and 2. In this sub-section, we present several methods for doing so.

Randomization: Our first approach is to construct the indication vectors via randomization. We claim that random vectors $\mathrm{w} \in \mathbb{R}^n$ are indication vectors with probability 1.

Theorem 2. *Let $\mathbb{P}$ be any continuous probability distribution on $\mathbb{R}^n$, and let $\mathfrak{G}$ be any collection of graphs over n nodes. Then $\mathbb{P}(\mathrm{w}$ is a $\mathfrak{G}$-indication vector$) = 1$.*

Proof. Recall that w is not an indication vector if and only if there are two distinct graphs $\mathcal{G}_1, \mathcal{G}_2$ and a vector $\mathrm{y} \in \mathbb{R}^{|\mathbb{V}|}$ such that

$$k^{-1}(\mathrm{y}) + \mathcal{E}_{\mathcal{G}_i} g(\mathcal{E}_{\mathcal{G}_i}^T \mathrm{y}) = -\mathrm{w},\ i = 1, 2.$$

Subtracting each equation from the other gives

$$\mathcal{E}_{\mathcal{G}_1} g(\mathcal{E}_{\mathcal{G}_1}^T \mathrm{y}) - \mathcal{E}_{\mathcal{G}_2} g(\mathcal{E}_{\mathcal{G}_2}^T \mathrm{y}) = 0. \tag{7}$$

For each $\mathcal{G}_1, \mathcal{G}_2$, the collection of solutions to (7) forms a set, and note that w is an indication vector if and only if the solutions y are not in any of these sets. Define

$$F(y) = \mathcal{E}_{\mathcal{G}_1} g(\mathcal{E}_{\mathcal{G}_1}^T \mathrm{y}) - \mathcal{E}_{\mathcal{G}_2} g(\mathcal{E}_{\mathcal{G}_2}^T \mathrm{y}),$$

so that $F : \mathbb{R}^n \to \mathbb{R}^n$ is a smooth function. Its differential is given by

$$\nabla F(y) = \mathcal{E}_{\mathcal{G}_1} \nabla g(\mathcal{E}_{\mathcal{G}_1}^T \mathrm{y}) \mathcal{E}_{\mathcal{G}_1}^T - \mathcal{E}_{\mathcal{G}_2} \nabla g(\mathcal{E}_{\mathcal{G}_2}^T \mathrm{y}) \mathcal{E}_{\mathcal{G}_2}^T,$$

where $\nabla g = \operatorname{diag}(\frac{dg_{ij}}{d\zeta_{ij}})$ is the derivative of g. Because $\frac{dg_{ij}}{d\zeta_{ij}} > 0$ by Assumption 2, $\nabla F(y)$ is the difference of two weighted graph Laplacians, with underlying graphs $\mathcal{G}_1, \mathcal{G}_2$ and positive weights. Thus ∇F never vanishes, and the implicit function theorem implies that the solutions of (7) form a zero measure set. Thus w is an indication vector if and only if the solutions y are not in the finite union of the zero measure sets defined by (7), i.e., a zero-measure set.

The mapping between $-\mathrm{w}$ and y, $-\mathrm{w} = k^{-1}(\mathrm{y}) + \mathcal{E}_{\mathcal{G}} g(\mathcal{E}_{\mathcal{G}}^T \mathrm{y}) = G(\mathrm{y})$, is smooth and strictly monotone, meaning that it is the gradient of a strictly convex and smooth function. Thus, the inverse function $\mathrm{y} = G^{-1}(\mathrm{w})$ is a smooth and strictly convex function, as the gradient of the dual function, which is also strictly convex and smooth [22]. This implies that G^{-1} is absolutely continuous [30], sending the zero measure sets to zero measure sets. In turn, the set that y has to avoid (for w to be an indication vector) is zero-measure, meaning that the corresponding set that w has to avoid is also zero measure. But $\mathbb{P}$ is a continuous probability measure, and thus the probability of the zero-measure set that w has to avoid is zero. This completes the proof. □

This method works under the Assumptions 1 and 2, but can produce stronger results when considering LTI agents and controllers. In particular, we can estimate the separation index of a randomly chosen vector.

Corollary 1. *Suppose the agents and controllers are LTI. Furthermore, suppose that w is sampled according to the standard Gaussian probability measure $\mathbb{P}$ on $\mathbb{R}^n$. Define $\beta = \min\{a_1, ..., a_n\}$ if $A \neq 0$, and $\beta = \min\{b_{ij}\}/\binom{n}{2}$ otherwise. Then for any $\delta > 0$, the separation index $\varepsilon = \varepsilon(\mathrm{w})$ satisfies $\delta \leq \varepsilon$ with probability $\geq 1 - 2^{n^2}(2\Phi(\delta/2\beta) - 1)$, where Φ is the cumulative distribution function of a standard Gaussian random variable.*

See [31] for the proof., omitted due to lack of space.

Remark 3. *Corollary 1 and Remark 2 give a viable method for assuring that the distance between different terminal states of the system (corresponding to different base graphs) is as large as desired. First, choose a desired degree of security p, which is the probability of the choice to be successful (say $p = .9999$). Choose δ so that $p \leq 1 - 2^{n^2}(2\Phi(\delta/2\beta) - 1)$. Now choose w randomly according to a standard Gaussian distribution, and multiply it by $1/\delta$.*

Let us present another, more constructive approach for designing indication vectors, building upon number theory.

Bases of Computation: For the rest of this subsection, we continue with LTI agents. We can apply this method if the elements of A are rational. In this case, the elements of $X_{\mathcal{G}}$ are all rational. The idea is that we can reconstruct the entries of $X_{\mathcal{G}}$ from $X_{\mathcal{G}}\mathrm{w}$ if w is of the form $\mathrm{w} = [1, M, M^2, ..., M^{n-1}]^T$ for M large enough.

Example 1. *Suppose that $C = [a, b, c]$ is a vector with positive integer entries having a numerator no greater than 9. Take $\mathrm{w} = [1, 10, 100]^T$. Then $C\mathrm{w} = a + 10b + 100c$ is a three-digit number, and we can reconstruct C by looking at the three digits individually - a is the unit digit, b is the tens digit, and c is the hundreds digit.*

We can generalize this to a more general framework.

Theorem 3. *Suppose that A, B are rational, the denominators of all entries of the matrices $\{X_{\mathcal{G}}\}_{\mathcal{G} \in \mathfrak{G}}$ divide D, and that the numerator (in absolute value) is no larger than N. Let M be any integer larger than $(2N+1)D$. Then the vector $\mathrm{w} = [1, M, ..., M^{n-1}]^T$ is a $\mathfrak{G}$-indication vector.*

Proof. Each element in the product $X_{\mathcal{G}}\mathrm{w}$ corresponds to a single row of $X_{\mathcal{G}}$ multiplied with w, so it's enough to reconstruct a row. We take a single row of $X_{\mathcal{G}}$ and mark it as $[\frac{p_1}{q_1}, \cdots, \frac{p_n}{q_n}]^T$, where $|p_i| \leq N$ and q_i divides D. We let $\mathrm{R} = (X_{\mathcal{G}}\mathrm{w})_i$. Therefore,

$$\mathrm{R} = \begin{bmatrix} \frac{p_1}{q_1} & \cdots & \frac{p_n}{q_n} \end{bmatrix} \mathrm{w} = \frac{p_1}{q_1} + \frac{p_1}{q_1} M + \cdots + \frac{p_n}{q_n} M^{n-1}.$$

We can define $m_i = \frac{D}{q_1}$, which is an integer no larger than D, and multiply both sides of the equation by D to obtain

$$D\,\mathrm{R} = m_1 p_1 + m_2 p_2 M + \cdots + m_n p_n M^{n-1}.$$

Note that $m_i p_i$ is an integer lying between $-ND$ and ND. We can add $\sum_{i=0}^{n-1}(NDM^i)$ to both sides of the equation, leading to

$$D\mathrm{R} + \sum_{i=0}^{n-1}(NDM^i) = (m_1 p_1 + ND) + \cdots + (m_n p_n + ND) M^{n-1}.$$

The left hand side is known, and the coefficients in the right hand side are integers between 0 and $2ND$. Thus, writing $D\mathrm{R} + \sum_{i=0}^{n-1}(NDM^i)$ in the system with radix M, the numbers $m_i p_i + ND$ can be computed by looking at the individual digits. Deducting ND and dividing by D gives all of the entries $\frac{p_i}{q_i}$, allowing reconstruction. □

V. Indication Vectors For Network Detection

Up to now, we have dealt with indication vectors, which give an easy way of solving Problem 1, i.e., differentiating between closed-loop systems of the form (4) which differ only on underlying graph level. We claim that we can go a step further and solve Problem 2, i.e., reconstructing the underlying graph of a system of the form (4). We now present the network reconstruction scheme based on indication vectors. The key notion that will allow us to take the leap is through the use of appropriate look-up tables.

Look-up tables are tables comprising of two columns, one called the key and the other called the value, that act like oracles and are designed to decrease runtime computations. The key is usually easy to come by, and the value is usually harder to find. Examples of look-up tables include mathematical tables, like logarithm tables and sine tables. Other examples include phone books and other databases like hospital or police records.

We now state the main result about network detection for general agents and controllers, focusing on the LTI case later.

Proposition 5. *Let $(\mathcal{G}, \Sigma, g)$ be a network system of the form* (4) *satisfying Assumptions 1 and 2. Then for any indication vector* w*, there exists an algoritgm solving Problem 2 using only a single exogenous output, namely* w.

Proof. Let $\mathfrak{G}$ be the collection of all graphs on n vertices. We construct a $\mathfrak{G}$-indication vector w using Theorem 2. Before running the system, we build a lookup table with keys being graphs $\mathcal{H} \in \mathfrak{G}$, and values being the outputs $\mathrm{y}_{\mathcal{H}}$, which can be computed by (6). Now, run the closed-loop system with the input w. By definition of a $\mathfrak{G}$-indication vector, we know that the terminal output y of closed-loop system completely classifies the underlying graph $\mathcal{G}$, i.e., different underlying graphs give rise to different terminal outputs. We can now reconstruct the graph $\mathcal{G}$ by comparing y to the values of the look-up table, finding the graph $\mathcal{H}$ minimizing $\|\mathrm{y} - \mathrm{y}_{\mathcal{H}}\|$. Then because w can differentiate the systems $(\mathcal{G}, \Sigma, g)$ and $(\mathcal{H}, \Sigma, g)$ if $\mathcal{G} \neq \mathcal{H}$, we must have that $\mathcal{G} = \mathcal{H}$. $\square$

Remark 4. *In the proof above, we assumed that the closed-loop system is run until the output converges. However, in practice, both numerical errors and early termination errors give us a skewed value of the true terminal output of the closed-loop system, as was discussed in Remark 1. In the algorithm presented above, we can tolerate an error of up to* $0.5\varepsilon(\mathrm{w})$ *in the value of* y.

Remark 5. *We should note that in order to implement the network detection scheme in the proof of Proposition 5, we need an observer with access to the look-up table, the output of all of the agents, and the input* w*. This network detection scheme is not distributed in the sense that it requires one observer to know the outputs of all of the agents. The size of the look-up table increases rapidly with the number of nodes if we don't assume anything about the underlying graph. One should note that should recall that the computation can be done offline, and that it can be completely parallelized - we are just comparing the entries of the table to the measured output. Furthermore, if we add additional assumptions on the graph (e.g., the underlying graph is a subgraph of some known graph), the size of the look-up table drops significantly.*

We can prove a stronger result for the case of LTI agents and controllers, namely a distributed implementation strategy and an analysis of the algorithm complexity.

Theorem 4. *Let $(\mathcal{G}, \Sigma, g)$ be a network system of the form* (4) *satisfying Assumptions 1 and 2, consisting of LTI agents and controllers, and that the matrices A, B have rational entries. Then there exists a distributed $O(n^3)$ algorithm solving Problem 2. It requires to run the system only once, with a specific constant exogenous input* w.

Remark 6. *In the case of LTI agents and controllers, finding the graph $\mathcal{G}$ is roughly equivalent to finding $X_{\mathcal{G}}$. The distributive nature of the algorithm is manifested in the fact that the i-th row of $X_{\mathcal{G}}$ is computed solely from the terminal output of the i-th agent.*

Proof. We pick an indication vector using the method of Theorem 3. The proof of Theorem 3 gives an easy way to reconstruct $X_{\mathcal{G}}$'s i-th row from the output of the i-th agent, taking $O(n)$ time. Doing this for all agents takes $O(n^2)$ times, and gives us $X_{\mathcal{G}}$. Afterward, we can reconstruct the graph Laplacian using the formula $\mathcal{E}_{\mathcal{G}} B \mathcal{E}_{\mathcal{G}}^T = -A - X_{\mathcal{G}}^{-1}$ in $O(n^3)$ time, and then find the underlying grpah by looking at the non-zero off-diagonal entries of it. $\square$

Note 1. *In the LTI case, we do not use look-up tables, but give a different solution relying on bases of computation. This allows us to have only a polynomial increase in time, and exempts us from worrying about storage issues.*

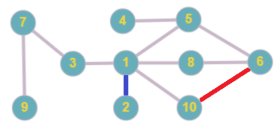
VI. Case Study : Neural Network

We consider a continuous neural network, as appearing in [32], on n neurons of one species. The governing ODE is,

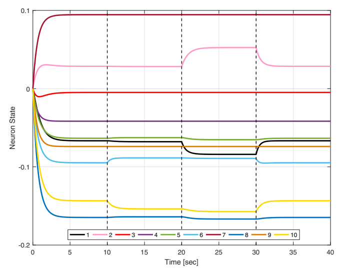
$$\dot{V}_i = -\frac{1}{\tau_i} V_i + b_i \sum_{j \sim i} (\tanh(V_j) - \tanh(V_i)) + \mathrm{w}_i \qquad (8)$$

where V_i is the voltage on the i-th neuron, $\tau_i > 0$ is the self-correlation time of the neurons, b_i is a coupling coefficient, and the external input w_i is any other input current to neuron i. This is a scenario similar to the one depicted in [33]. We run the system with 10 neurons. The correlation times were chosen randomly between 0.5_{sec} and 1_{sec}. The homogeneity requirement on the network forces us to take equal b_i-s over all agents, and we choose $b_i = 0.1$. We note that the agents, modeled by $\dot{x_i} = -\frac{1}{\tau_i} x_i + u_i; y_i = \tanh(x_i)$, are output stricly MEIP, as the following storage function can be used for to prove output-strict passivity with respect to $(\mathrm{u}_i, \tanh(\tau_i \mathrm{u}_i))$, $V_i(x_i) = \int_0^{x_i} \tanh(s) ds - \int_0^{\tau_i \mathrm{u}_i} \tanh(s) ds - \tanh(\tau_i \mathrm{u}_i)(x - \tau_i \mathrm{u}_i)$.

We choose an indication vector as in the proof of Proposition 5, and run the system with the original underlying graph, showing in Fig. 2(a). The output of the system can be seen in Fig. 2(b). We first run the system for 10 seconds (enough for convergence). After 10 seconds, the red edge in Fig. 2(a)



(a) The interaction graph. The red edge is cut after 10sec, and the blue edge is cut after 20sec.



(b) Trajectories of the neural network (8) with changes in the underlying network.

Fig. 2. Simulation of a neural network.

gets cut off. We can see that the output of agent #6 (in light blue) and agent #10 (in yellow) change meaningfully, so we are able to detect the change in the underlying graph. After ten more seconds, we also remove the blue edge. We see that again the outputs of two agents, #1 (in black) and #2 (in pink), are changed by a measurable amount, allowing to detect the second change in the underlying graph. Finally, after 20 seconds, we reintroduce both removed edges. We can see that the system returns to its original steady-state.

VII. Conclusion

In this work we presented a procedure operating on a network system that allows for the reconstruction of the underlying network assuming only prior knowledge on the agents. This was done through the novel notion of indication vectors, that were achieved for general maximally equilibrium-independent passive agents, allowing for detection of the underlying network in a very general case. We have found stronger results for LTI agents, allowing a distributed cubic-time reconstruction of the underlying network, while dealing with numerical errors present in the system. We have exhibited the use of indication vectors in network detection, and demonstrated the results in a simulation.

References

[1] M. Mesbahi and M. Egerstedt, *Graph Theoretic Methods in Multiagent Networks*. Princeton Series in Applied Mathematics, Princeton University Press, 2010.

[2] Y.-Y. Liu, J.-J. Slotine, and A.-L. Barabási, "Controllability of complex networks," *Nature*, vol. 473, pp. 167–173, may 2011.

[3] A. Julius, M. Zavlanos, S. Boyd, and G. J. Pappas, "Genetic network identification using convex programming," *IET Systems Biology*, vol. 3, pp. 155–166, May 2009.

[4] M. J. Naylor, L. C. Rose, and B. J. Moyle, "Topology of foreign exchange markets using hierarchical structure methods," *Physica A: Statistical Mechanics and its Applications*, vol. 382, no. 1, pp. 199 – 208, 2007. Applications of Physics in Financial Analysis.

[5] E. Zheleva, E. Terzi, and L. Getoor, "Privacy in social networks," *Synthesis Lectures on Data Mining and Knowledge Discovery*, vol. 3, no. 1, pp. 1–85, 2012.

[6] V. Sakkalis, "Review of advanced techniques for the estimation of brain connectivity measured with eeg/meg," *Computers in Biology and Medicine*, vol. 41, no. 12, pp. 1110 – 1117, 2011. Special Issue on Techniques for Measuring Brain Connectivity.

[7] S. L. Bressler and A. K. Seth, "Wienergranger causality: A well established methodology," *NeuroImage*, vol. 58, no. 2, pp. 323 – 329, 2011.

[8] P. Qin, B. Dai, B. Huang, G. Xu, and K. Wu, "A survey on network tomography with network coding," *IEEE Communications Surveys Tutorials*, vol. 16, pp. 1981–1995, Fourthquarter 2014.

[9] D. Urban and T. Keitt, "Landscape connectivity: A graph-theoretic perspective," *Ecology*, vol. 82, no. 5, pp. 1205–1218, 2001.

[10] D. Materassi and G. Innocenti, "Unveiling the connectivity structure of financial networks via high-frequency analysis," *Physica A: Statistical Mechanics and its Applications*, vol. 388, no. 18, pp. 3866–3878, 2009.

[11] B. M. Sanandaji, T. L. Vincent, and M. B. Wakin, "Exact topology identification of large-scale interconnected dynamical systems from compressive observations," in *Proceedings of the 2011 American Control Conference*, pp. 649–656, June 2011.

[12] M. Nabi-Abdolyousefi and M. Mesbahi, "Network identification via node knockout," *IEEE Transactions on Automatic Control*, vol. 57, pp. 3214–3219, Dec 2012.

[13] M. M. Marzieh Nabi-Abdolyousefi, "A sieve method for consensus-type network tomography," in *Controllability, Identification, and Randomness in Distributed Systems*, ch. 3, pp. 31–38, Springer Theses (Recognizing Outstanding Ph.D. Research), Springer, Cham, 2014.

[14] E. Nozari, Y. Zhao, and J. Corts, "Network identification with latent nodes via auto-regressive models," *IEEE Transactions on Control of Network Systems*, vol. PP, no. 99, pp. 1–1, 2017.

[15] A. Mauroy and J. M. Hendrickx, "Spectral identification of networks with inputs," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pp. 469–474, Dec 2017.

[16] H. Bai, M. Arcak, and J. Wen, *Cooperative Control Design: A Systematic, Passivity-Based Approach*. Communications and Control Engineering, Springer, 2011.

[17] M. Arcak, "Passivity as a design tool for group coordination," *IEEE Transactions on Automatic Control*, vol. 52, pp. 1380–1390, Aug. 2007.

[18] G.-B. Stan and R. Sepulchre, "Analysis of interconnected oscillators by dissipativity theory," *IEEE Transactions on Automatic Control*, vol. 52, no. 2, pp. 256 – 270, 2007.

[19] L. Scardovi, M. Arcak, and E. D. Sontag, "Synchronization of interconnected systems with applications to biochemical networks: An input-output approach," *IEEE Transactions on Automatic Control*, vol. 55, no. 6, pp. 1367–1379, 2010.

[20] A. J. van der Schaft and B. M. Maschke, "Port-hamiltonian systems on graphs," Sep. 2012. arXiv:1107.2006v2 [math.OC].

[21] M. Bürger, D. Zelazo, and F. Allgöwer, "Duality and network theory in passivity-based cooperative control," *Automatica*, vol. 50, no. 8, pp. 2051–2061, 2014.

[22] R. T. Rockafellar, *Network Flows and Monotropic Optimization*. Belmont, Massachusetts: Athena Scientific, 1998.

[23] M. Sharf and D. Zelazo, "A network optimization approach to cooperative control synthesis," *IEEE Control Systems Letters*, vol. 1, pp. 86–91, July 2017.

[24] M. Sharf and D. Zelazo, "Analysis and Synthesis of MIMO Multi-Agent Systems Using Network Optimization," *ArXiv e-prints*, Nov. 2017.

[25] C. Godsil and G. Royle, *Algebraic Graph Theory*. Graduate Texts in Mathematics, Springer, 1st ed., 2001.

[26] R. T. Rockafeller, "Characterization of the subdifferentials of convex functions," *Pacific Journal of Mathematics*, vol. 17, no. 3, pp. 497–510, 1966.

[27] M. Bando, K. Hasebe, A. Nakayama, A. Shibata, and Y. Sugiyama, "Dynamical model of traffic congestion and numerical simulation," *Phys. Rev. E*, vol. 51, pp. 1035–1042, Feb 1995.

[28] A. Franci, L. Scardovi, and A. Chaillet, "An input-output approach to the robust synchronization of dynamical systems with an application to the hindmarsh-rose neuronal model," in *2011 50th IEEE Conference on Decision and Control and European Control Conference*, pp. 6504–6509, Dec 2011.

[29] F. Drfler and F. Bullo, "Synchronization in complex networks of phase oscillators: A survey," *Automatica*, vol. 50, no. 6, pp. 1539 – 1564, 2014.

[30] J. Mal, "Absolutely continuous functions of several variables," *Journal of Mathematical Analysis and Applications*, vol. 231, no. 2, pp. 492 – 508, 1999.

[31] M. Sharf and D. Zelazo, "Network Identification: A Passivity and Network Optimization Approach," *ArXiv e-prints*, July 2018.

[32] L. Scardovi, M. Arcak, and E. D. Sontag, "Synchronization of interconnected systems with an input-output approach. part ii: State-space result and application to biochemical networks," in *IEEE Conference on Decision and Control (CDC)*, pp. 615–620, Dec 2009.

[33] C. J. Quinn, T. P. Coleman, N. Kiyavash, and N. G. Hatsopoulos, "Estimating the directed information to infer causal relationships in ensemble neural spike train recordings," *Journal of Computational Neuroscience*, vol. 30, pp. 17–44, Feb 2011.