

We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

7,200

Open access books available

192,000

International authors and editors

210M

Downloads

Our authors are among the

154

Countries delivered to

TOP 1%

most cited scientists

14%

Contributors from top 500 universities



WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index
in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com



Chapter

Eigen-Analysis of Multi-Agent Systems and Large Scale Systems Using Data Driven and Machine Learning Algorithms

Kenneth McDonald, Zhihua Qu and Azwirman Gusrialdi

Abstract

Eigenvalue analysis is central in stability analysis and control design of linear dynamic systems. While eigen-analysis is a standard tool, determining eigenvalues of multi-agent systems and/or interconnected dynamical systems remains challenging due to the sheer size of such systems, changes of their topology, and limited information about subsystems' dynamics. In this chapter, a set of scalable, data-driven estimation and machine learning algorithms are presented to determine eigenvalue(s) and in turn stability of such large-scale complex systems. We begin with distributed algorithms that estimate all the eigenvalues of multi-agent cooperative systems, where their subsystems are modeled as a single integrator and interconnected by local communication networks. The algorithms are then extended to the data-driven version that estimate the dominant eigenvalues of large-scale interconnected systems with unknown dynamical model. Subsequently, we study input-output stability of subsystems and extend eigen-analysis to investigation of passivity shortage using the input-output data. This analysis is then further extended to machine learning algorithms by which stability properties of unknown subsystems can be learned. These results are illustrated by examples.

Keywords: stability, eigenvalues, multi-agent systems, consensus algorithm, input-output stability, passivity and passivity shortage, machine learning

1. Introduction

Eigen-analysis is a fundamental and well known concept in linear algebra and linear systems theory. Technically, it deals with eigenvalues and eigenvectors of matrices or linear transformations. When applied to linear systems, eigenanalysis quantifies characteristics of dynamic responses, reveals such properties as stability and robustness, and provides closed-form solutions. As applications move toward multi-agent systems, large scale networked systems, and machine learning, eigen-analysis remains to be an effective approach for qualitative and quantitative analyses. This chapter aims to illustrate this fact by focusing upon a few of contemporary topics.

This chapter begins with a summary of most foundational results in Section 2. The section contains the following specific results. Section 2.1 provides eigenanalysis of linear time-invariant systems, their solution, and their stability by summarizing the basic results from [1, 2]. Section 2.2 introduces Lyapunov direct method [3] and presents quadratic Lyapunov analysis in terms of eigenvalues. The Lyapunov direct method applies to both linear and nonlinear dynamic systems. One way to bridge analyses of linear and nonlinear dynamic systems is to parameterize their stability analysis. To this end, Section 2.3 introduces dissipativity theory [4], which allows for the classification of linear and nonlinear systems into passive and passivity-short systems, providing a unified framework for their analyses. In Section 2.4, eigenanalysis is applied to investigate stability analysis of multi-agent systems, including cooperative and networked systems [5], and the result enables a scalable and modular design [6] of networked control systems in terms of two key parameters on system's input-output relationship. In Section 2.5, input-output relationship is explicitly derived for linear time invariant systems for the purpose of performing machine learning as well as the subsequent stability analysis and control design.

Section 3 explores the eigen-analysis of multi-agent and large scale systems, with a particular focus on systems that may have unknown models. The first part addresses the problem of distributed eigenvalue estimation in multiagent systems, where each agent has access only to local information. While various distributed algorithms have been proposed to tackle this issue (e.g., [7–10]), existing approaches such as the power iteration [7, 8] and consensus-based algorithm [9] are typically limited to estimating only the dominant eigenvalues. Even algorithms that can estimate all eigenvalues, such as [10], are often restricted to specific types of matrices, such as rowstochastic ones. To address these limitations, Section 3.1 presents distributed algorithms capable of estimating all eigenvalues for any irreducible matrix, broadening the applicability of eigenvalue estimation methods. The discussion then extends to the challenge of distributed dominant eigenvalue estimation for unknown linear time-invariant systems within autonomous, large scale systems. Existing data-driven techniques, including dynamic mode decomposition [11, 12], power iteration [13], prony method [14], distributed optimization-based approach [15], and Hankel matrix [16], each face drawbacks. These include centralization, applicability only to Laplacian matrices, or limitations to matrices with distinct eigenvalues. To overcome these challenges, two distributed datadriven algorithms are presented in Section 3.2, which estimate eigenvalues by learning local models and applying model reduction techniques, making them suitable for handling large scale models.

Different from Section 3, Section 4 addresses stability analysis and control design through machine learning. Specifically, a data driven algorithm is presented to learn the two key parameters. To demonstrate connection and effectiveness, the two parameters are calculated using both approaches of eigenanalysis and machine learning. The machine learning approach bypasses the step of model identification and hence is more direct and efficient in design and analysis. Combined with the results in Section 2.4, machine learning can be applied to multi-agent systems as well as large scale systems.

2. Preliminaries

2.1 Analysis of linear time-invariant systems and their stability

Eigenvalue analysis is both fundamental and straightforward to investigate stability of linear time-invariant systems of form

$$\dot{x} = Ax + Bu, \quad y = Cx + Du, \quad (1)$$

where $x \in \mathfrak{R}^n$ is the state, $u \in \mathfrak{R}^m$ is the control input, and $y \in \mathfrak{R}^l$ is the output. Defining the *matrix exponential function* e^{At} as

$$e^{At} = \sum_{j=0}^{\infty} \frac{1}{j!} A^j t^j \quad (2)$$

we know that e^{At} has the property of

$$\frac{d}{dt} e^{At} = A e^{At} = e^{At} A.$$

Hence, the solution to system (1) is

$$\dot{x}(t) = e^{At} x(0) + \int_0^t e^{A(t-\tau)} B u(\tau) d\tau \quad (3)$$

in which $e^{A(t-t_0)}$ is also called the state transition matrix.

Given matrix A , its eigenvalues λ_i and eigenvectors v_i are defined as

$$A v_i = \lambda_i v_i, \quad i = 1, \dots, n.$$

Matrix $A \in \mathfrak{R}^{n \times n}$ has n eigenvalues and, if the number of eigenvectors is r but less than n , it always has n eigenvectors and generalized eigenvectors. Assembling these eigenvectors and generalized eigenvectors into matrix S , we have

$$S^{-1} A S = J, \quad J = \begin{bmatrix} J_1 & & \\ & \ddots & \\ & & J_r \end{bmatrix}, \quad J_i = \begin{bmatrix} \lambda_i & 1 & 0 & \dots \\ 0 & \ddots & \ddots & \\ \vdots & \ddots & \ddots & 1 \\ 0 & \dots & 0 & \lambda_i \end{bmatrix} \in \mathfrak{R}^{n_i \times n_i},$$

where J is the Jordan canonical form with diagonal blocks J_i , and n_i is the geometric multiplicity of eigenvalue λ_i .

It follows from the structure of J and the Taylor series expansion in (2) that

$$e^{At} = S \begin{bmatrix} e^{J_1 t} & & \\ & \ddots & \\ & & e^{J_r t} \end{bmatrix} S^{-1}, \quad e^{J_i t} = \begin{bmatrix} e^{\lambda_i t} & t e^{\lambda_i t} & \dots & \frac{t^{n_i-1}}{(n_i-1)!} e^{\lambda_i t} \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & t e^{\lambda_i t} \\ 0 & 0 & \dots & e^{\lambda_i t} \end{bmatrix}. \quad (4)$$

Therefore, the following necessary and sufficient conditions can be concluded from (3) and (4):

- i. System (1) with $u = 0$ is Lyapunov stable if and only if none of its eigenvalues is in the right open half plane and those on the imaginary axis are of geometrical multiplicity one.

ii. System (1) with $u = 0$ is asymptotically stable if and only if matrix A is Hurwitz, i.e., its eigenvalues are all in the left open half plane.

iii. System (1) is input-to-state stable if and only if it is asymptotically stable.

iv. System (1) with $u = 0$ is exponentially stable if and only if it is asymptotically stable.

If A is not Hurwitz, control u can be designed to achieve stability when pair $\{A, B\}$ is controllable.

2.2 Stability analysis of linear time-varying systems

Consider linear time-varying system:

$$\dot{x} = A(t)x + B(t)u, \quad y = C(t)x + D(t)u \quad (5)$$

where $x \in \mathfrak{R}^n$ is the state, $u \in \mathfrak{R}^m$ is the input, and $y \in \mathfrak{R}^p$ is the output. It is known that pointwise eigenvalues of time-varying matrix $A(t)$ being in the left open half plane *do not* imply stability of system (5), and a counterexample can be found in [3]. Instead, eigen-analysis of linear time varying system should be performed through the Lyapunov direct method.

Consider the autonomous system:

$$\dot{x} = A(t)x, \quad x \in \mathfrak{R}^n. \quad (6)$$

For linear systems, Lyapunov function can always be chosen to be a quadratic function of form

$$V(x, t) = x^T P(t)x,$$

where $P(t)$ is a symmetric matrix. Its time derivative along trajectories of system (6) is also quadratic as

$$\dot{V}(x, t) = -x^T Q(t)x,$$

where $P(t)$ and $Q(t)$ are related by the so-called differential Lyapunov equation

$$\dot{P}(t) = -A^T(t)P(t) - P(t)A(t) - Q(t). \quad (7)$$

To determine whether system (6) is asymptotically stable or not, the following three-step backward process needs to be applied: (a) choose $Q(t)$ to be symmetric, uniformly bounded (in the sense that the maximum eigenvalue is uniformly bounded from above as $\lambda_{\max}(Q(t)) \leq \bar{c} < \infty$), and positive definite (in the sense that in the sense that the minimum eigenvalue is uniformly above zero as $\lambda_{\min}(Q(t)) \geq \underline{c} > 0$). The simplest choice is $Q(t) = I$. (b) Solve $P(t)$ from Eq. (7). (c) System (6) is asymptotically stable if and only if solution $P(t)$ is uniformly bounded and positive definite. Should matrix $A(t)$ be constant, Eq. (7) is algebraic, and solution P is constant.

2.3 Analysis of nonlinear systems

Consider the following nonlinear affine system

$$\dot{x} = f(x, t) + g(x, t)u, \quad y = h(x, t), \quad (8)$$

where $x \in \mathfrak{R}^n$, $u \in \mathfrak{R}^m$, and $y \in \mathfrak{R}^l$ are the state, input, and output, respectively. System (8) includes system (6) as a special case. Stability of system (8) can be investigated using Lyapunov direct method, as in Section 2.2. Lyapunov function takes the general form of $V(x, t)$ and have the following time derivative along the trajectories of system (8):

$$\dot{V} = \frac{\partial V}{\partial t} + \left(\frac{\partial V}{\partial x} \right) [f(x, t) + g(x, t)u]. \quad (9)$$

To show asymptotic stability under $u = 0$, one need to choose a positive definite function $\eta(x)$ and solve for Lyapunov function $V(x, t)$ from the following partial differential equation:

$$\frac{\partial V}{\partial t} + \left(\frac{\partial V}{\partial x} \right) f(x, t) = -\eta(x). \quad (10)$$

Stability can be determined by checking whether solution $V(x, t)$ is both positive definite and decrescent (i.e., upper and lower bounded by positive definite functions $\gamma_1(x), \gamma_2(x)$ as $\gamma_1(x) \leq V(x, t) \leq \gamma_2(x)$).

To investigate input-output relationship of linear and nonlinear systems, we can use the dissipativity theory. System (8) is said to be dissipative with respect to a positive semi-definite (p.s.d.) storage function $V(x)$ and a supply rate function $\Phi(u, y)$ if $V(0) = 0$ and if, for all $x_0 \in \mathcal{X}$,

$$V(x(\infty)) - V(x(0)) \leq \int_0^t \Phi(u(\tau), y(\tau)) d\tau. \quad (11)$$

As a Lyapunov function, the so-called storage function represents a broader concept of the energy stored within the system. Consequently, the above inequality implies that the stored energy at any given time, as described by the storage function, is always less than or equal to the total energy supplied to the system, including the initial energy.

Should the supply function be quadratic, inequality (11) is satisfied with

$$\Phi(u, y) = -\eta(x) + u^T y - \frac{\epsilon}{2} \|u\|^2 - \frac{\rho}{2} \|y\|^2, \quad (12)$$

where $\eta(x)$ is a positive semi-definite function. If the storage function is positive definite, it is a Lyapunov function, and system (8) is asymptotically stable. Depending upon the values of parameters ϵ and ρ , several sub-classes of dissipativity are given in **Table 1**.

It is well known that passive linear systems must be of relative degree 0 or 1, Lyapunov stable, and also minimum phase (or inversely Lyapunov stable).

This means that most of the stable systems are not passive. Since the engineered systems, such as teleoperation of an n -link robot [17] and synchronous generator,

Sub-classes	ϵ	ρ
Passive	$\epsilon, \rho \geq 0$	
Input strictly passive (ISP)	> 0	≥ 0
Output strictly passive (OSP)	≥ 0	> 0
Input-output strictly passive (IOSP)	> 0	> 0
Passivity short (PS)	either $\epsilon < 0$ or $\rho < 0$	
Input-feedforward passivity short (IFPS)	< 0	≥ 0
Output-feedback passivity short (OFPS)	≥ 0	< 0

Table 1.
Sub-classes of dissipativity and their parameter values.

would have bounded outputs when their inputs are bounded, these systems are typically input-feedforward passivity short (IFPS). It is known that, with an appropriate self-feedback control, a stabilizable linear system can achieve the IFPS property [18]. It is shown in the rest of the chapter that both passivity and passivity shortage enable modular analysis/design of complex systems, indirect eigen-analysis using input-output data, and machine learning.

2.4 Multi-agent systems and cooperative networked systems

A multi-agent system consists of cooperative agents with simple dynamics:

$$\dot{y}_i = u_i, \quad i \in \mathcal{N}, \quad (13)$$

where $\mathcal{N} = \{1, \dots, n\}$ is the set of agents. These agents communicate through a local communication network of graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, where $\mathcal{E} \subseteq \mathcal{N} \times \mathcal{N}$ is the edge set and $e_{ij} \in \mathcal{E}$ implies that y_j is sent by the j th agent to the i th agent, or $j \in \mathcal{N}_i$ with $\mathcal{N}_i \subset \mathcal{N}$ is the i th agent's neighbor set. Graph \mathcal{G} is generally directed, i.e., $e_{ij} \in \mathcal{E}$ does not necessarily mean $e_{ji} \in \mathcal{E}$ or vice versa. Graph \mathcal{G} is said to be strongly connected if every node is connected by directed edges to any other node. These local interactions enable the following cooperative consensus protocol:

$$u_i = \sum_{j \in \mathcal{N}_i} w_{ij} (y_j - y_i), \quad (14)$$

where $w_{ij} > 0$ are weights. The resulting cooperative system can be expressed as

$$\dot{y} = -Ly, \quad (15)$$

where

$$L = [L_{ij}], \quad L_{ij} = \begin{cases} 0 & \text{if } j \neq i \text{ and } j \notin \mathcal{N}_i \\ -w_{ij} & \text{if } j \neq i \text{ and } j \in \mathcal{N}_i \\ \sum_{j \in \mathcal{N}_i} w_{ij} & \text{if } j = i \end{cases} \quad (16)$$

is the so-called Laplacian. Laplacian L is always Lyapunov stable, $\lambda_1(L) = 0$ is the eigenvalue(s) closest to the imaginary axis, and the corresponding eigenvector is the vector of 1 s. Should \mathcal{G} be strongly connected, $\lambda_1(L) = 0$ is unique, and all of y_i converge to the same consensus value. If \mathcal{G} is undirected and connected, Laplacian L can be symmetric (with $w_{ij} = w_{ji}$), and its eigenvalues values $\lambda_i(L)$ are all real and the property that

$$0 = \lambda_1(L) < \lambda_2(L) \leq \dots \leq \lambda_n(L). \quad (17)$$

While multi-agent systems are easy to analyze, networked systems from applications often have heterogeneous dynamics in the form of

$$\dot{z}_i = f_i(z_i, u_i), \quad y_i = h_i(z_i), \quad (18)$$

where $z_i \in \mathfrak{R}^{n_i}$ and $y_i \in \mathfrak{R}^l$ are the state and the output of the i^{th} system, respectively. System (18) includes multi-agent system (13) as a special case. Instead of assembling all the dynamics from system (18) and analyzing the stability together, one can use the concepts of passivity and passivity shortage to perform modular analysis and design. To this end, first assume that system (18) has storage function $V_i(z_i)$ and the following dissipativity property:

$$\dot{V}_i \leq y_i^T u_i - \epsilon_i u_i^T u_i - \rho_i y_i^T y_i \quad (19)$$

where ϵ_i and ρ_i can assume values according to **Table 1**. Next, assume that Laplace L is symmetric and connected, and revise the consensus protocol (14) by incorporating cooperative control gain $\kappa > 0$ as

$$u_i = \kappa \sum_{j=1}^n w_{ij} (y_j - y_i). \quad (20)$$

Then, choosing the following Lyapunov function

$$V = \sum_{i=1}^n V_i, \quad (21)$$

and taking the time derivative along the trajectories of system (18) under control (20) yield

$$\begin{aligned} \dot{V} &\leq \sum_{i=1}^n [y_i^T u_i - \epsilon_i u_i^T u_i - \rho_i y_i^T y_i] \\ &= -y^T Q y, \end{aligned} \quad (22)$$

where

$$Q = \kappa L + \kappa^2 \text{diag}\{\epsilon_i\} L^2 + \text{diag}\{\rho_i\} I. \quad (23)$$

It is obvious that the overall system of heterogeneous subsystems reaches consensus if matrix Q has the property of $\lambda_{\min}(Q) \geq 0$. This is ensured if the following scalar, quadratic inequality admits positive solution κ :

$$\kappa\lambda_2(L) + \kappa^2 \left(\min \left\{ 0, \min_i \epsilon_i \right\} \right) \lambda_n^2(L) + \left(\min_i \rho_i \right) \geq 0. \quad (24)$$

A quick analysis of the above inequality reveals the following stability property for cooperative networked systems:

- i. If all the systems in (18) are passive, consensus can be achieved for any $\kappa \in (0, \infty)$.
- ii. If all the systems in (18) are either passive or IFPS, there exists $\bar{\kappa} > 0$ such that consensus can be achieved for any $\kappa \in (0, \bar{\kappa})$.
- iii. If all the systems in (18) are either passive or OFPS, there exists $\underline{\kappa} > 0$ such that consensus can be achieved for any $\kappa \in (\underline{\kappa}, \infty)$.

If the systems in (18) are a mixture of being passive, IFPS, and OFPS, consensus can still be achieved by changing common gain κ into individual gains κ_i . These results show that, for cooperative consensus control (20) or its individual gain version, choices of cooperative control gains are modular and plug-and-play in spite of heterogeneous dynamics (18).

2.5 Data-driven modeling

For linear systems, data-driven modeling refers to the approach of building models based solely on input-output data, without requiring explicit knowledge of the system's internal dynamics or parameters. This method is particularly useful when the system is a black or gray box, or its internal parameters are inaccessible. In this section, input-output relationship is established using discretization so that machine learning of key parameters can be done later in Section 4.

Consider linear system (1). Its discretized version is: given $x(0) = 0$ and sampling period T_s ,

$$x_{l+1} = A_d x_l + B_d u_l, \quad y_l = C x_l + D u_l, \quad A_d = e^{AT_s}, \quad B_d = A^{-1}(A_d - I)B. \quad (25)$$

To further represent the system in terms of inputs and outputs [19], consider the representation given by

$$y_l = \sum_{\tau=0}^l g_\tau u_{l-\tau}, \quad g_0 = D, \quad g_\tau = CA_d^{\tau-1}B_d, \forall \tau > 0. \quad (26)$$

Passive systems have relative degree r to be either 0 or 1. On the other hand, passivity short systems may have $r \geq 1$, causing the convoluted matrix G_d to become singular. This can be avoided by utilizing its reduced order representation as needed. Let $Y = G_d U$, where N is sufficiently large,

$$Y = [y_r \ y_{r+1} \ \cdots \ y_{N-1}], \quad U = [u_0 \ u_1 \ \cdots \ u_{N-1-r}], \quad (27)$$

$$G_d = \begin{bmatrix} g_r & 0 & 0 & \cdots & 0 \\ g_{r+1} & g_r & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & 0 \\ g_{N-1} & g_{N-2} & g_{N-3} & \cdots & g_r \end{bmatrix} \in \mathfrak{R}^{(N-r) \times (N-r)}. \quad (28)$$

It follows that, for sufficiently small

$$T_s > 0,$$

$$\int_0^\infty u^T y ds = T_s U^T Y = T_s U^T G_d U, \quad \int_0^\infty u^T u ds = T_s U^T U. \quad (29)$$

Next, the relationship between the reversed input and output sequences is described. Let $P \in \mathbb{R}^{N \times N}$ be the exchange (or reversal) matrix whose elements are 0 except for anti-diagonal elements being 1's. If $Y = G_d U$, the response under $U' = PU$ is

$$Y' = G_d P U, \quad (30)$$

where

$$y'_l = \sum_{\tau=0}^l g_\tau u'_{l-\tau} = \sum_{\tau=0}^l g_\tau u_{N-1-l+\tau}. \quad (31)$$

Therefore, defining

$$Y'' = P Y', \quad (32)$$

we have

$$y''_l = y'_{N-1-l} = \sum_{\tau=0}^{N-1-l} g_\tau u_{l+\tau} = \sum_{\tau'=l}^{N-1} g_{\tau'-l} u_{\tau'}. \quad (33)$$

The above expression can be rewritten in a compact form as

$$P \underbrace{\begin{bmatrix} u' \\ G_d (P U) \end{bmatrix}}_{Y'} = P G_d P U = G_d^T U. \quad (34)$$

Eqs. (29) and (34) form the input-output relationship that enable machine learning.

3. Distributed algorithms for estimating the eigenvalues of multi-agent and large scale interconnected systems

We start this section by presenting a distributed algorithm for estimating all the eigenvalues in multi-agent systems where the agent is modeled as a single integrator. Next, we discuss how to extend the approach to the case of large scale interconnected systems with unknown system matrix and whose subsystem's dynamics is given by a

linear-time invariant system. To this end, distributed data-driven estimation algorithms are presented which only rely on the collected measurements of the system's states (i.e., offline data).

3.1 Model-based distributed algorithm for estimating eigenvalues in multi-agent systems

Consider a multi-agent system consisting of n number of agents and whose overall dynamics is given by

$$\dot{y} = Ay, \quad (35)$$

where $y = [y_1, \dots, y_n]^T$ and $y_i \in \mathbb{R}$ denotes the output of the i -th agent. It is assumed that agent i only has access to the local information, that is the i -th row of matrix A , denoted by vector $[A]_{i*}$. Furthermore, the agents can communicate/exchange information with each other via a communication network whose topology is similar to the sparsity of matrix A and is given by a strongly connected directed graph \mathcal{G} , that is matrix A is irreducible. For example, matrix A can be a Laplacian or adjacency matrix, as described in Section 2.4. The agents aim to collaboratively estimate all the eigenvalues of A by using only their local information $[A]_{i*}$.

3.1.1 Model-based distributed eigenvalues estimation algorithm

In order to estimate all the eigenvalues of A in a distributed manner, all the agents cooperatively perform the following steps whose detailed analysis can be found in [20]:

1. First, all the agents collaboratively transform matrix A into a nonsingular matrix $\bar{A} = [\bar{a}_{ij}]$ defined as $\bar{A} = A + cI_n$ with $c \in \mathbb{R}$. Based on Gershgorin theorem, the constant c can be chosen cooperatively by the agents to ensure that $|\bar{a}_{ii}| > \sum_{j \neq i} |\bar{a}_{ij}|$ for all $i = \{1, \dots, n\}$. To that end, agent i first sets $c_i(0) = \epsilon_i + \sum_j |a_{ij}|$ for arbitrary value of $\epsilon_i > 0$ to ensure $|\bar{a}_{ii}| > \sum_{j \neq i} |\bar{a}_{ij}|$. Next, all the agents should choose a common value c from all the values of $c_i(0)$ so that $|\bar{a}_{ii}| > \sum_{j \neq i} |\bar{a}_{ij}|$ for all $i = \{1, \dots, n\}$. Specifically, the value of c can be chosen as $c = \max_i c_i(0)$ which can be computed distributively by performing the following maximum consensus protocol [21] for n iterations

$$c_i(k+1) = \max_{j \in \mathcal{N}_i \cup \{i\}} c_j(k), \quad k = 0, 1, \dots, n. \quad (36)$$

2. After constructing a nonsingular matrix \bar{A} from matrix A , each agent then distributively computes \bar{A}^{-1} , denoted by Z , by solving a system of linear equations $\bar{A}Z = I_n$ using its own local information $[\bar{A}]_{i*}$. To this end, each agent implements the following update rule

$$\hat{Z}_i(k+1) = \hat{Z}_i(k) - \frac{1}{|\mathcal{N}_i|} P_i \left(|\mathcal{N}_i| \hat{Z}_i(k) - \sum_{j \in \mathcal{N}_i} \hat{Z}_j(k) \right), \quad (37)$$

where matrix $\hat{Z}_i(k) \in \mathbb{R}^{n \times n}$ denotes the local estimation of $Z = \bar{A}^{-1}$ by the i -th agent at the k -th iteration and whose initial value is chosen to satisfy $[\bar{A}]_{i*}^T \hat{Z}_i(0) = [I_n]_{i*}^T$. Moreover, matrix $P_i = P_i^T \in \mathbb{R}^{n \times n}$ is an orthogonal projection on the kernel of vector $[\bar{A}]_{i*}$, namely $P_i = I_n - \frac{1}{[\bar{A}]_{i*}^T [\bar{A}]_{i*}} [\bar{A}]_{i*} [\bar{A}]_{i*}^T$. Each agent's local estimate $\hat{Z}_i(k)$ under (37) will then converge to \bar{A}^{-1} as $k \rightarrow \infty$.

3. Each agent computes all the eigenvalues of A by exploiting the relationship between $\lambda_i(A)$ and $\lambda_i(Z)$ given by $\lambda_i(A) = \frac{1}{\lambda_i(Z)} - c$.

One of the benefits of the above method is that each agent is not only able to distributively estimate all the eigenvalues of A but also the corresponding eigenvectors. Specifically, noting that both matrices A and \bar{A}^{-1} share the same eigenvectors each subsystem can compute distributively the eigenvectors of matrix A from the learned matrix \bar{A}^{-1} .

Remark 1. Distributed algorithm (37) converges exponentially to \bar{A}^{-1} [20]. Furthermore, it can be observed from (37) that each agent needs to exchange n^2 values with its neighbors and require to store also n^2 values. One may then ask why each agent does not just flood its row $[A]_{i*}$ to its neighbors so that each agent can then construct matrix A . In contrast to (37), the flooding strategy is not locally adaptable under topology changes [20]. Moreover, convergence of update rule (37) is also guaranteed under time-delay and asynchronous setting [22].

3.1.2 An illustrative example

Consider a multi-agent system consisting of four omnidirectional mobile robots whose kinematic model are given by (13) where y_i denotes its position. Since the robot can move in any directions, its kinematic model can be decoupled for each axis and thus in this example, without loss of generality, we only focus on the motion control of the robots for one axis. The goal is to design control input (velocity) u_i , which depends on the positions of some other robots obtained via a communication network, so that all the robots gather at a common location, e.g., for recharging their batteries. This problem is also known as rendezvous problem [5]. To this end, one can design a consensus protocol given in (14) and assuming the network topology is strongly connected it is ensured that all the four robots gather at a common location. The overall system's closed-loop dynamics can then be written as in (35) where matrix $A = -L$. For example, Laplacian matrix L can be designed as

$$L = \begin{bmatrix} 0.5 & -0.5 & 0 & 0 \\ 0 & 0.4 & 0 & -0.4 \\ -0.8 & -0.2 & 1 & 0 \\ 0 & 0 & -0.8 & 0.8 \end{bmatrix} \quad (38)$$

whose eigenvalues equal to 0, 1.2616, $0.7192 \pm 0.5367i$. The second smallest eigenvalue of L measures the network connectivity and convergence rate for reaching consensus [23] while the third smallest eigenvalue provides a metric for ensuring robust connectivity in the presence of single robot failures [24]. The robots can

cooperatively estimate the eigenvalues of L using the steps presented in previous subsection. Specifically, the robots set $c = 3$ and initial values $\hat{Z}_i(0)$ as

$$\hat{Z}_1(0) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ -2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad \hat{Z}_2(0) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & -\frac{5}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

$$\hat{Z}_3(0) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad \hat{Z}_4(0) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{10}{8} \\ 0 & 0 & 0 & 0 \end{bmatrix}. \quad (39)$$

After executing update rule (37) for 100 iterations, all local estimations $\hat{Z}_i(k)$ converge to \bar{L}^{-1} , that is

$$\hat{Z}_i(100) = \begin{bmatrix} 0.2859 & 0.0421 & 0.0009 & 0.0044 \\ 0.0014 & 0.2947 & 0.0062 & 0.0310 \\ 0.0573 & 0.0232 & 0.2505 & 0.0024 \\ 0.0121 & 0.0049 & 0.0527 & 0.2637 \end{bmatrix}, i = \{1,2,3,4\}. \quad (40)$$

After distributively estimating \bar{L}^{-1} , robot j can then calculate the eigenvalues of matrix L given by $\frac{1}{\lambda_i(\hat{Z}_j(100))} - 1$.

3.2 Distributed and data-driven algorithms for estimating eigenvalues of a large scale interconnected system

In this subsection, we extend the setting in the previous subsection to large scale (physically) interconnected systems where its subsystem's dynamics can be modeled as an LTI system, for example power system [25] and thermal model of large buildings [26], which makes the dimension of the overall system very high. It is worth noting that in some applications one may only be interested in estimating the dominant eigenvalues of the overall matrix A . For example, in power systems the dominant eigenvalues, *aka* inter-area oscillation modes, play an important role for wide-area monitoring applications. These slow eigenvalues arise from the oscillations between the coherent areas in power system which may lead to small-signal stability concern and thus needs to be constantly monitored. However, the high dimension of the overall system prevents one from using distributed estimation algorithm presented in the previous subsection. In addition, the large-scale system model (i.e., matrix A) is also often unknown/not available in practice due to geographical constraint or it may change because of perturbation which calls for data-driven methods.

Let us consider a large scale interconnected system divided into r nonoverlapping and coherent clusters where the j -th cluster consists of n_j subsystems. Since the clusters are coherent, one can represent the i -th cluster with an equivalent subsystem whose state $\tilde{x}_i \in \mathbb{R}^p$ is the averaged state of all the subsystems in that

cluster, i.e., $\tilde{x}_i = \frac{\sum_{j=1}^{n_i} w_j x_j}{\sum_{j=1}^{n_i} w_j}$ where $w_i > 0$ denotes some weights. Hence, the reduced order model of the large scale interconnected system can be written as

$$\dot{\tilde{x}} = A_r \tilde{x}, \quad (41)$$

where $\tilde{x} = [\tilde{x}_1^T, \dots, \tilde{x}_r^T]^T$ and A_r is the reduced system matrix whose eigenvalues correspond to the dominant/slow eigenvalues of the original system matrix A . For details of the modeling, one can refer to [27]. It is assumed that the i -th subsystem has access only to its own sampled state $x_i(k) \triangleq x_i(t)|_{t=kT}$, $k = 0, 1, \dots$ where T denotes the sampling time. The discrete-time model of (41) is then given by

$$\tilde{x}(k+1) = A_d \tilde{x}(k) \quad (42)$$

where matrix $A_d = e^{A_r T}$. The relation between eigenvalues of both A_r and A_d is given by $\lambda_i(A_d) = e^{\lambda_i(A_r)T}$.

If the eigenvalues of A_r are distinct, one can readily apply the Prony method [14, 28] to estimate the eigenvalues of A_r using the averaged state $\tilde{x}(k)$. In the following, we present an alternative distributed and data-driven method to estimate the eigenvalues of A_r which does not require them to be distinct.

3.2.1 Distributed model learning algorithms

Briefly speaking, the idea is to first learn in a distributed fashion the reduced model A_r from the averaged state $\tilde{x}(k)$ [29]. Without loss of generality and for the sake of simplicity, it is assumed that there exists a virtual agent in each cluster which collects measurement $x_i(k)$ from all subsystems in the i -th cluster to calculate the average state in the corresponding cluster and cooperatively learns the reduced model with the other virtual agents. To that end, we assume that the communication network topology between the virtual agents is given by a strongly connected directed graph.

Assuming that the sampling time is sufficiently small, one can approximate A_d as

$$A_d = I + A_r T \quad (43)$$

whose eigenvalues are given by $\lambda_i(A_d) = 1 + T\lambda_i(A_r)$. Each virtual agent i then collects the following sampled averaged state

$$\begin{aligned} X_i &= [\tilde{x}_i(k_0), \dots, \tilde{x}_i(k_{m-1})] \in \mathbb{R}^{p \times m}, \\ Y_i &= [\tilde{x}_i(k_0 + 1), \dots, \tilde{x}_i(k_{m-1} + 1)] \in \mathbb{R}^{p \times m}, \end{aligned} \quad (44)$$

where m denotes the amount of data used for learning. Note that the index $\{k_0, k_1, \dots, k_{m-1}\}$ does not need to be sequential. Defining matrices $X = [X_1^T, \dots, X_r^T]^T$ and $Y = [Y_1^T, \dots, Y_r^T]^T$, we have the following relation

$$Y = A_d X. \quad (45)$$

Furthermore, let us set $m = pr$. Given that matrix X is nonsingular, matrix A_d can then be learned by computing

$$A_d = YX^{-1} \quad (46)$$

using only local information available to each virtual agent. To this end, the virtual agents first estimate X^{-1} using update rule similar to (37). Once all the virtual agents compute the estimate of X^{-1} , the i -th virtual agent can then learn the $(p(i-1) + 1) - th$ until the pi -th rows of matrix A_d , denoted by $A_{d,i}$, as follows:

$$A_{d,i} = Y_i X^{-1}. \quad (47)$$

Finally, using the learned $A_{d,i}$ the virtual agents can distributively estimate the eigenvalues $\lambda_i(A_d)$ (and $\lambda_i(A_r)$ accordingly) using the method presented in Section 3.1.

One key assumption in the method described above is that all the virtual agents are able to construct matrix X_i from its own state measurement such that the matrix X is nonsingular. When the matrix X is ill-conditions for any given set of sampled measurements, one can alternatively learn matrix A_d by solving a least square problem. To that end, each virtual agent first constructs the measurement matrices X_i, Y_i in (44) with $m > p^2r$ samples and from (45) satisfy the relation $Y_i = A_{d,i}X$. Next, the agent constructs a vector $a_i \in \mathbb{R}^{p^2r}$ whose entries equal to the entries of unknown matrix $A_{d,i}$. Equation $Y_i = A_{d,i}X$ can be written as

$$X_i^r a_i = h_i, \quad (48)$$

where matrix $X_i^r \in \mathbb{R}^{pm \times p^2r}$ and vector $h_i \in \mathbb{R}^{pm}$ are constructed from matrices X and Y_i , respectively. Agent i can then learn the entries of $A_{d,i}$ by solving the following least square problem

$$\hat{a}_i = \arg \min_{a_i} \frac{1}{2} \|X_i^r a_i - h_i\|_2^2. \quad (49)$$

Remark 2. In order to construct matrix X_i^r and learn the local model, agent i needs to collect X_j from all other agents which requires all-to-all bidirectional communication between the virtual agents. This communication requirement and the size of vector a_i can be reduced if the sparsity structure of matrix A_d is known in advance. After learning the local model, distributed algorithms presented in Section 3.1.1, and whose complexity is discussed in Remark 1, can be adopted to distributively estimate the eigenvalues.

Remark 3. If the structure or property of matrix A_d is known, one can then incorporate this side information as constraints in solving (49).

Remark 4. When the measurements are noisy, one can perform data preprocessing by filtering the noise or smoothing the data before learning the local model.

3.2.2 An illustrative example

Consider an interconnected system of 16 undamped oscillators, divided into 4 coherent clusters as shown in **Figure 1**. Dynamics of the i -th oscillator is given by

$$\Delta \ddot{\delta}_i = - \sum_{j \in \mathcal{N}_i} \frac{\Delta \delta_i - \Delta \delta_j}{r_{ij}}, \quad (50)$$

where $\Delta\delta_i$ denotes the phase angle of oscillator i and r_{ij} represents the reactance of tie lines connecting oscillators i and j . We set r_{ij} of the intra-cluster and intercluster tie lines to be 0.001 per unit and 1 per unit respectively. The reduced order matrix A_r is given by $A_r = \begin{bmatrix} 0 & I \\ L_r & 0 \end{bmatrix}$ where $-L_r$ is a weighted Laplacian matrix and the state $\tilde{x} = [\Delta\delta_{r,1}, \dots, \Delta\delta_{r,4}, \Delta\dot{\delta}_{r,1}, \dots, \Delta\dot{\delta}_{r,4}]^T$ with $\Delta\delta_{r,i}$ denotes the average phase angle of the equivalent oscillator in the i -th cluster. Using the slow-fast time-scale separation principle [30], matrix L_r can be analytically calculated as

$$L_r = \begin{bmatrix} -0.4997 & 0.2498 & 0.0001 & 0.2498 \\ 0.2498 & -0.4997 & 0.2498 & 0.0001 \\ 0.0001 & 0.2498 & -0.4997 & 0.2498 \\ 0.2498 & 0.0001 & 0.2498 & -0.4997 \end{bmatrix}. \quad (51)$$

In order to learn distributively matrix A_r , a virtual agent is assigned to each cluster which can communicate with each other as shown in **Figure 1**. Each virtual agent computes the average state for each area as illustrated in **Figure 2a**. Each virtual agent then distributively learn the corresponding rows of A_r by solving (49) where the number of data $m = 1000$ and the index $k_{j+1} - k_j = 10$ in (44) for all j , see **Figure 2b**. Note that the sampling time equals to $T = 0.001s$. In addition, the virtual agent also incorporates the side information regarding the sparsity structure of A_r and the property of the Laplacian $-L_r$, discussed in Section 2.4, when solving its least square problem. The learned model is given by

$$\hat{L}_r = \begin{bmatrix} -0.5004 & 0.2487 & 0.0014 & 0.2503 \\ 0.2496 & -0.5001 & 0.2497 & 0.0008 \\ 0.0007 & 0.2504 & -0.5006 & 0.2496 \\ 0.2507 & 0.0016 & 0.2486 & -0.5009 \end{bmatrix}. \quad (52)$$

The comparison between the data and the trajectories of the reduced dynamical system using the learned model is shown in **Figure 2b**. It can be observed that the virtual agents are able to learn accurately the reduced order model A_r . Finally, the

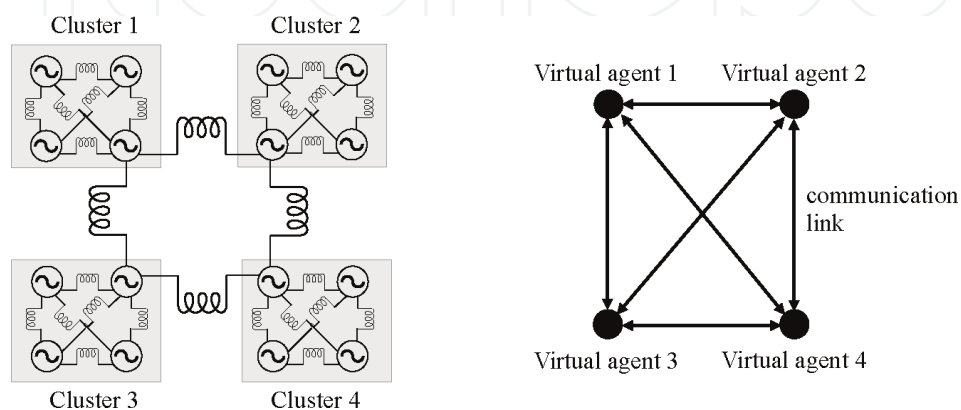


Figure 1. Left: 4-cluster 16 second-order oscillators. Right: virtual agents representing each cluster and their communication network topology.

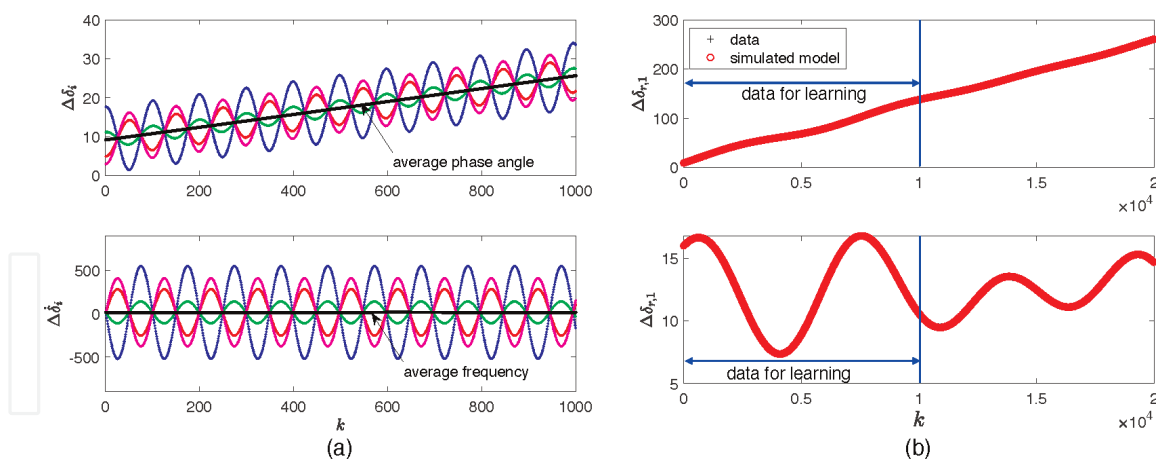


Figure 2. (a) Snapshots of true states and average states of oscillators in cluster 1; (b) Comparison between data and trajectories generated using the learned reduced order model in cluster 1.

comparison between eigenvalues of the analytically calculated matrix A_r and the eigenvalues of the learned matrix \hat{A}_r is given by

$$\begin{aligned} \text{true eigenvalues} &: 0, 0, \pm 0.7070i, \pm 0.7070i, \pm 0.9996i, \\ \text{estimated eigenvalues} &: 0, 0, \pm 0.7074i, \pm 0.7091i, \pm 0.9994i. \end{aligned}$$

4. Machine learning for passive and passivity-short systems

In this section, two methods are explored to determine the indices ϵ and ρ of passive and passivity-short systems. These indices can be found using the eigenvalue-based approach of linear matrix inequality (LMI) and the data-driven technique proposed in [19, 31] for passive systems. The LMI approach requires perfect knowledge of the system but offers the highest level of precision in determining the passivity indices, as it avoids the errors associated with discrete-time approximations, such as those introduced by sampling and discretization. In comparison, the data-driven approach is less precise but offers an approach of finding passivity or passivity short indices solely and directly from input output data, providing the alternative for unknown systems and avoiding the step of model identification.

4.1 Direct method: Linear matrix inequality (LMI) approach

The LMI approach is an eigenvalue based method that leverages the continuous time system for defining an inequality condition that expresses constraints on a system using matrices, where the passivity constraint is integrated directly into the condition. By formulating the passivity conditions as LMIs, convex optimization techniques can be utilized to solve for the indices.

Consider system (1) and the Lyapunov function $V = 0.5x^T Sx$, where S is a positive definite matrix. The time derivative of Lyapunov function is given by

$$\dot{V} = \frac{1}{2}x^T (SA + A^T S)x + x^T S B u, \quad (53)$$

which satisfy the following dissipativity condition

$$\dot{V} \leq u^T y - \frac{\epsilon}{2} \|u\|^2 - \frac{\rho}{2} \|y\|^2, \quad (54)$$

if and only if the following matrix inequality (i.e., in terms of its minimum eigenvalue being non-negative) hold:

$$W(\epsilon, \rho) = \begin{bmatrix} -A^T S - SA - \rho C^T C & -SB + C^T - \rho C^T D \\ -B^T S + C - \rho D^T C & -\epsilon I + D + D^T - \rho D^T D \end{bmatrix} \geq 0. \quad (55)$$

Leveraging semi-definite program solvers for optimization, passivity indices ϵ and ρ can be determined by following the procedure defined in Algorithm 1.

Algorithm 1 Passivity Indices Using LMI

- 1: Solve $\tilde{\epsilon} = \arg \max_S \epsilon$ subject to $W(\epsilon, 0) \geq 0, S > 0$
 - 2: **if** $\tilde{\epsilon} > 0$ **then**
 - 3: **The system is passive**
 - 4: The ISP index $\epsilon^* = \tilde{\epsilon}$
 - 5: The OSP index ρ^* is determined by $\rho^* = \arg \max_S \rho$, subject to $W(0, \rho) \geq 0, S > 0$
 - 6: The IOSP indices ρ (or ϵ) can be found for any fixed value of ϵ (or ρ) less than its upper bound using ϵ^* (or $\rho^*) = \arg \max_S \epsilon$ (or ρ^*), subject to $W(\epsilon, \rho) \geq 0, S > 0$
 - 7: **end if**
 - 8: **if** $\tilde{\epsilon} < 0$ **then**
 - 9: **The system is passivity short**
 - 10: Set $\epsilon = -1$ to admit unique solutions
 - 11: The OFPS index ρ^* is determined by $\rho^* = \arg \max_S \rho$, subject to $W(-1, \rho) \geq 0, S > 0$
 - 12: The IFPS index ϵ is determined, for any $\rho \in [0, \rho^*)$, using $\epsilon = s \epsilon$, subject to $W(\epsilon, \rho) \geq 0, S > 0$
 - 13: **end if**
-

Remark 5. The above LMI solution is computationally efficient as it has the complexity of linear programming. Should the system dynamics are linear but contain parameterizable and bounded uncertainties, one could adopt the model of so-called interval systems. In this case, the matrices $\{A, B, C, D\}$ may have their known parts and their uncertain parts. For example, system matrix A becomes $A + \Delta A$, where entries of ΔA belong to certain known intervals. In such cases, the above LMI solution can be extended to these interval systems, and the reader is referred to relevant literature.

4.2 Indirect method: Data-driven approach

In previous sections, analytical methods for determining passivity indices, using model-based approaches, were examined. While these methods provide precise methods for determining passivity, they are explicitly dependent on the mathematical model of the system. However, in practical applications, the system model cannot be obtained or determined due to the complex nature of the system or unknown knowledge of parameters. To address these challenges, we extend the data-driven

approaches in [19, 31] to find passivity and passivity short indices on both the input and output channels of a model using data only.

Assume that system (1) can be tested but its model is unknown. Then, its output data can be collected upon feeding an input into the system. If $x(0) = 0, x(\infty) = 0$, we have

$$0 \leq \int_0^{\infty} u^T y ds - \frac{\epsilon}{2} \int_0^{\infty} \|u\|^2 ds - \frac{\rho}{2} \int_0^{\infty} \|y\|^2 ds \quad (56)$$

where $\epsilon, \rho > 0$ are the largest possible values to satisfy the above inequality. It can be seen that, if ρ (or ϵ) is given then

$$\epsilon \leq \frac{\int_0^{\infty} (u^T y + y^T u - \rho \|y\|^2) ds}{\int_0^{\infty} \|u\|^2 ds} \quad \text{or} \quad \rho \leq \frac{\int_0^{\infty} (u^T y + y^T u - \epsilon \|u\|^2) ds}{\int_0^{\infty} \|y\|^2 ds} \quad (57)$$

where ϵ (or ρ) > 0 is the largest value satisfying the above inequality. This can be further refined into objective functions

$$\epsilon = \max_U f_{\epsilon}(U) \quad \text{or} \quad \rho = \max_U f_{\rho}(U) \quad (58)$$

where

$$f_{\epsilon}(U) = \frac{U^T Y + Y^T U - \rho Y^T Y}{U^T U} \quad \text{or} \quad f_{\rho}(U) = \frac{U^T Y + Y^T U - \epsilon U^T U}{Y^T Y}. \quad (59)$$

To solve Eq. (58), a gradient descent method can be implemented using the update rule

$$U^{(k+1)} = U^{(k)} - \delta^{(k)} \nabla_{f_z} (U^{(k)}) \quad (60)$$

where $z \in [\rho, \epsilon]$, the optimal step size $\delta^{(k)}$ is given by Eq. (67) and $\nabla_{f_z} (U^{(k)})$ is the gradient of f_z defined in Algorithm 3 of the Appendix.

Algorithm 2 provides the procedure to determine passivity indices ϵ and ρ in a similar manner to Algorithm 1, and Algorithm 3 defines the gradient descent function used in Algorithm 2.

Remark 6. The above data-driven approach is iterative and computationally simple, and its convergence property depends upon the specific numerical search algorithm used. In the following examples, the standard gradient search algorithm is used, and its convergence is optimized by online implementing the optimal stepsize derived in the Appendix.

4.3 Illustrative examples

In this section, we demonstrate the effectiveness of the proposed algorithms on simple second order systems and a real world synchronous generator system of order 6.

Algorithm 2 Passivity Indices Using Data Driven Indirect Approach

- 1: Solve the following where $\rho = 0$

$$\tilde{\epsilon} = \text{PassivityGradientDescent}(f_\epsilon, \nabla f_\epsilon, U_\epsilon^{(0)}, \delta_\epsilon, K, 0)$$
 - 2: **if** $\tilde{\epsilon} > 0$ **then**
 - 3: **The system is passive**
 - 4: The ISP index $\epsilon^* = \tilde{\epsilon}$
 - 5: The OSP index ρ^* is determined by

$$\rho^* = \text{PassivityGradientDescent}(f_\rho, \nabla f_\rho, U_\rho^{(0)}, \delta_\rho, K, 0)$$
 - 6: The IOSP indices ρ (or ϵ) can be found for any fixed value of ϵ (or ρ) less than its upper bound using

$$\epsilon^* = \text{PassivityGradientDescent}(f_\epsilon, \nabla f_\epsilon, U_\epsilon^{(0)}, \delta_\epsilon, K, \rho)$$

$$\rho^* = \text{PassivityGradientDescent}(f_\rho, \nabla f_\rho, U_\rho^{(0)}, \delta_\rho, K, \epsilon)$$
 - 7: **end if**
 - 8: **if** $\tilde{\epsilon} < 0$ **then**
 - 9: **The system is passivity short**
 - 10: Set $\epsilon = -1$ to admit unique solutions
 - 11: The OFPS index ρ^* is determined by

$$\rho^* = \text{PassivityGradientDescent}(f_\rho, \nabla f_\rho, U_\rho^{(0)}, \delta_\rho, K, \epsilon)$$
 - 12: The IFPS index ρ is determined, for any $\rho \in [0, \rho^*)$, using

$$\epsilon^* = \text{PassivityGradientDescent}(f_\epsilon, \nabla f_\epsilon, U_\epsilon^{(0)}, \delta_\epsilon, K, \rho)$$
 - 13: **end if**
-

4.3.1 Simple 2nd order systems

Consider the following two input-output stable, continuous time systems:

$$\dot{x} = \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u \quad y = [1 \quad 0]x + 0.5u. \quad (61)$$

$$\dot{x} = \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u \quad y = [1 \quad 0]x. \quad (62)$$

These systems are discretized using a first-order hold with a sampling rate of 0.01 and $T_s = 2500$. The initial inputs were chosen to be

$$U_\epsilon^{(0)}(t) = \frac{\sin(0.2\pi t)}{\|\sin(0.2\pi t)\|}, \quad U_\rho^{(0)}(t) = \frac{\sin(2\pi t)}{\|\sin(2\pi t)\|} \quad (63)$$

Figure 3 presents the passivity index results for passive system (61), and **Figure 4** shows the results for passivity short system (62). In both figures, the Nyquist diagrams provide visual confirmation of the systems' stability. The convergence of the ISP and IFPS indices, ϵ , is depicted in the top-right plot, illustrating how, over the prescribed number of iterations k , the estimate of ϵ from the indirect approach converges to the true value of ϵ given by the direct approach. Similarly, the convergence of OSP and OFPS indices ρ are shown in the bottom-left plots. Finally, the bottom-right plots present the Pareto fronts for IOSP and IOPS indices. The Pareto front represents the trade-offs between ϵ and ρ , where optimality is achieved by minimizing one parameter while solving for the other. For passive systems, the feasible region is finite and constrained to the first quadrant, indicating that both ϵ and ρ

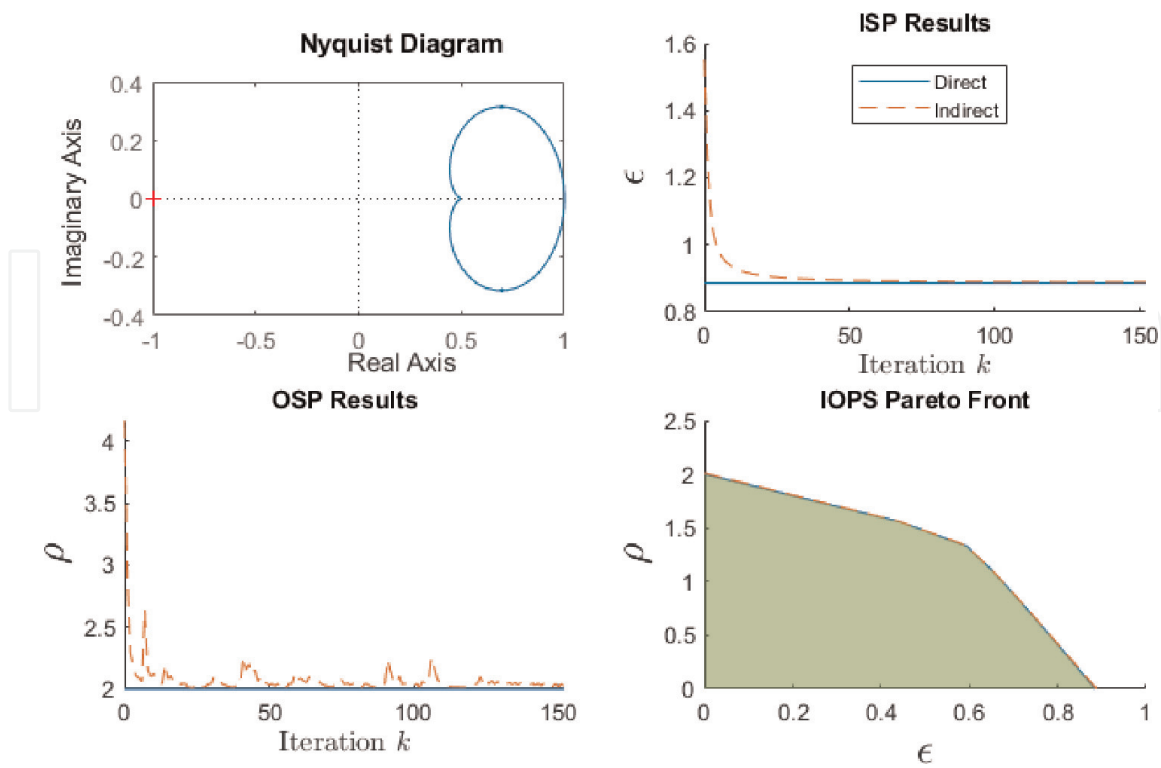


Figure 3. Passivity results for system (61). Nyquist diagram (top left), convergence of indirect ϵ to direct ϵ (top right), convergence of indirect ρ to direct ρ (bottom left), pareto front (bottom right).

remain non-negative. In contrast, the passivity-short system exhibits an unbounded region, with ϵ extending to negative infinity, reflecting the greater flexibility and reduced constraints inherent in passivity-short conditions.

4.3.2 A real-world example

Consider the following 6th order model of a turbine generator [32], in form of system (1) with matrices:

$$A = \begin{bmatrix} 0 & 377 & 0 & 0 & 0 & 0 \\ -0.2673 & 0 & -0.2946 & 0 & 0 & 0.211 \\ -0.2763 & 0 & -0.580 & 0.1695 & 0 & 0 \\ -66.3405 & 0 & -535.1923 & -20 & 0 & 0 \\ 0 & -0.09 & 0 & 0 & -3.333 & 0 \\ 0 & 0 & 0 & 0 & 1.0 & -1.0 \end{bmatrix},$$

$$B = [0 \ 0 \ 0 \ 0 \ 3.3333 \ 0]^T$$

$$C = [1.2668 \ 0 \ 1.3966 \ 0 \ 0 \ 0],$$

$$D = [0].$$

Compared to that in [32], the above is a reduced order model in which the excitation control dynamics are removed. The discretized system is obtained using a least squares approximation with a sampling rate of 0.01 and $T_s = 2500$ and it can be

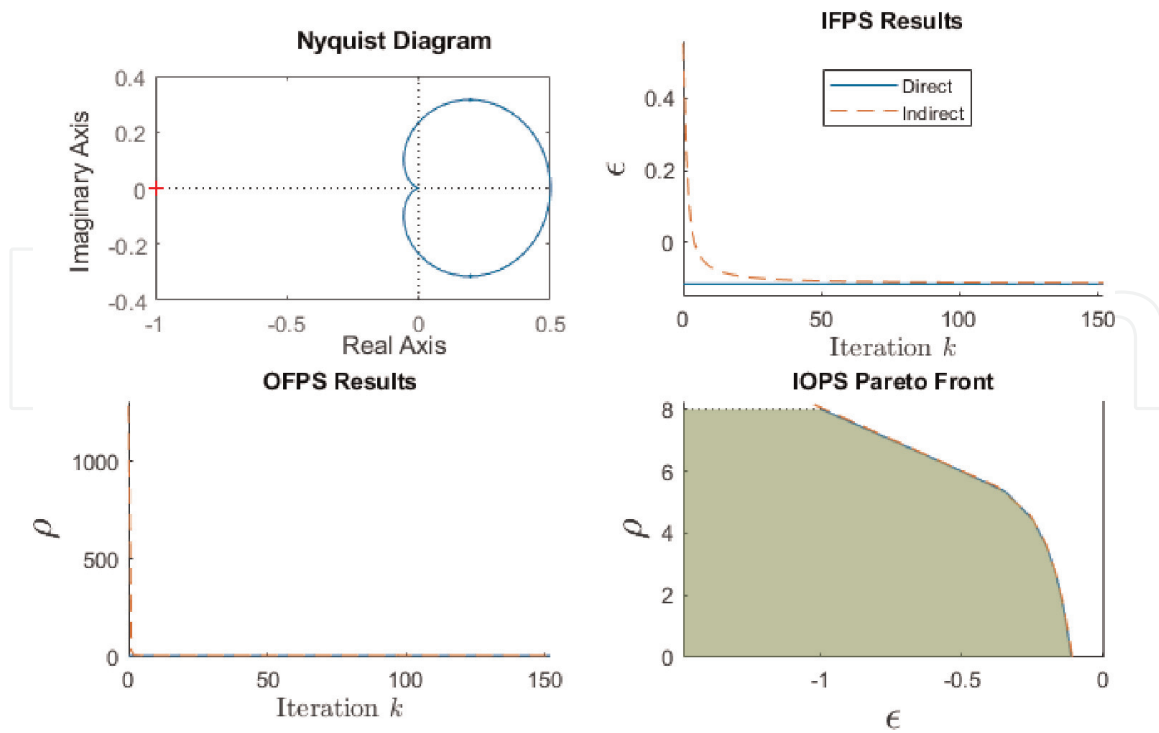


Figure 4. Passivity results for system (62). Nyquist diagram (top left), convergence of indirect ϵ to direct ϵ (top right), convergence of indirect ρ to direct ρ (bottom left), pareto front (bottom right).

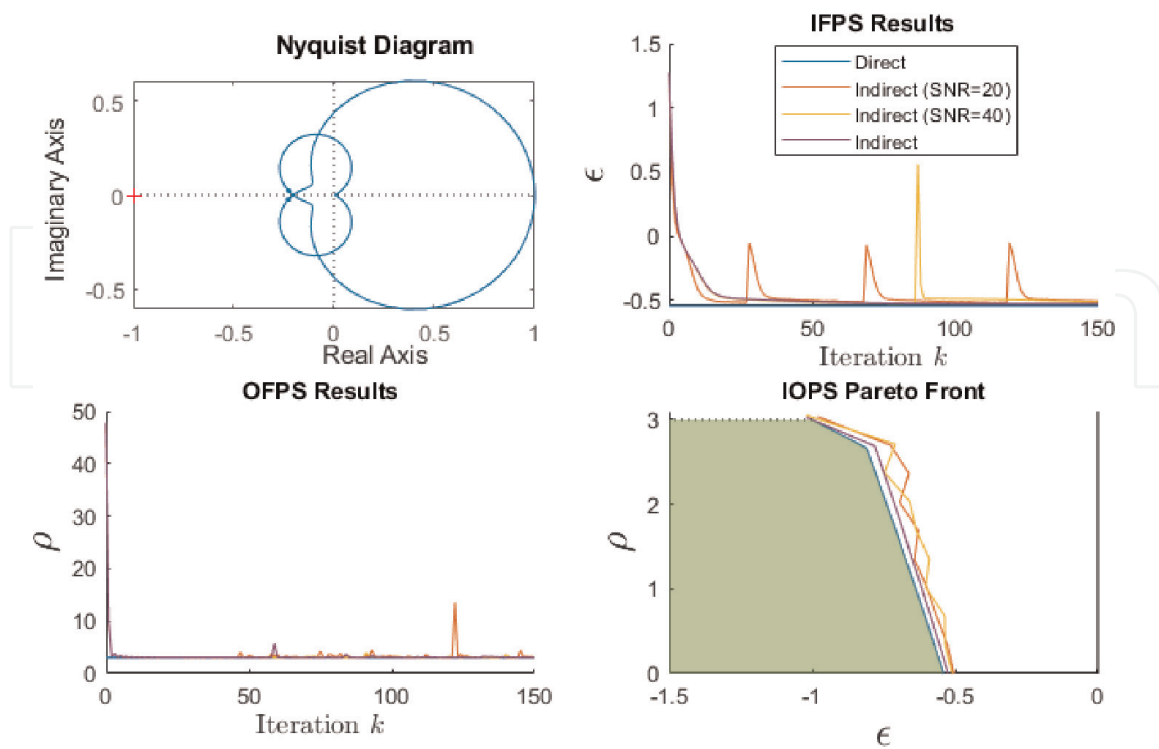


Figure 5. Passivity results for turbine generator. Nyquist diagram (top left), convergence of indirect ϵ , with and without noise, to direct ϵ (top right), convergence of indirect ρ to direct ρ with and without noise (bottom left), pareto front with and without noise (bottom right).

shown that the system is passivity short. The initial inputs from (63) are reused, and to demonstrate practicality, the indirect approach was solved with and without the presence of noise. A white Gaussian noise, with a specified signal to noise ratio (SNR) given in dB, was applied to the outputs of the system that are determined directly from U . **Figure 5** provides the results for the turbine generator system when both direct and indirect passivity approaches applied as well as with and without noises.

5. Conclusion

In this chapter the problem of determining eigenvalues, and thereby determining stability, for interconnected and multi-agent dynamic systems is investigated. To overcome challenges posed by system size, complexity, topology changes, and incomplete information about subsystems, scalable, data driven estimation and machine learning algorithms are proposed. First, two distributed algorithms capable of estimating all the eigenvalues of multi-agent cooperative systems and/or large-scale interconnected systems, including dominant eigenvalues, was proposed. Additionally, the input-output stability of subsystems is investigated and the extension of eigen-analysis is made to study passivity constraints using system matrices or purely input output data. This analysis was further advanced by leveraging machine learning algorithms to learn the passivity properties without prior knowledge of the system dynamics. The effectiveness of the proposed algorithms was demonstrated through numerical examples. The work presented in this chapter contributes to laying a more concrete foundation for scalable stability analysis, enabling the use of data-driven and machine learning algorithms in complex interconnected systems.

Open source code availability

The repository containing the implemented algorithms presented in this chapter can be acquired at <https://forms.gle/z3T75RMRrWM7GSfb9>.

Acknowledgements

The work of Zhihua Qu was supported in part by the U.S. Department of Energy's under Award DE-EE0007998, Award DE-EE0009028, Award DEEE0009152, Award DE-EE0009339, and Award DE-AC05-76RL01830. The work of Azwirman Gusrialdi was supported by the Research Council of Finland under Academy Project 330073.

A. Appendix

The gradient descent algorithm to determine the passivity index, as well as the derivation for the optimal step size of the indirect approach is provided.

A.1 Gradient descent for passivity

To minimize the optimization functions (58), we employ the well-known strategy of gradient descent. This method involves iteratively updating U by moving in the

direction opposite to the gradient of the function f at each step, thereby approaching the minimum of f . Specifically, the passivity gradient descent approach is provided in Algorithm 3 where an input U is chosen initially, then for k iterations, various projections of the input are passed to the system to calculate $f(U)$, its gradient $\nabla f(U)$, and the optimal stepsize δ . At the end of each iteration, $\nabla f(U)$, and δ are used to update the input U by Eq. (60).

A.2 Optimal Stepsize δ

To increase the rate of convergence of estimation, the stepsize $\delta^{(k)}$ can be updated each iteration. Let f_z , where $z \in [\epsilon, \rho]$ be generalized to f , then the optimal step size is given by

$$\delta^{(k)} = \arg \min_{\delta \in \mathbb{R}} f\left(U^{(k)} - \delta^{(k)} \nabla f\left(U^{(k)}\right)\right). \quad (64)$$

Algorithm 3 Passivity Gradient Descent

- 1: Choose an initial control input $U^{(0)} \in \mathbb{R}^{1 \times N}$
 - 2: **for** k K **do**
 - 3: Simulate to get $Y_1^{(k)}, Y_2^{(k)}$, and $Y_3^{(k)}$, under inputs $U^{(k)}, PU^{(k)}$ and $PY_1^{(k)}$.
 - 4: **if** ρ is given **then**
 - 5: Calculate $\epsilon^{(k)} = f_\epsilon(U^{(k)})$
 - 6: Calculate $\nabla f(U^{(k)}) = \nabla f_\epsilon^{(k)} = \frac{2(Y_1 + PY_2 - \rho PY_3)}{U^T U} - 2f_\epsilon(U) \frac{U}{U^T U}$
 - 7: Simulate to get $Y_4^{(k)}, Y_5^{(k)}$, and $Y_6^{(k)}$ under input $\nabla f_\epsilon^{(k)}, P\nabla f_\epsilon^{(k)}$ and $PY_4^{(k)}$
 - 8: Calculate the step size $\delta^{(k)}$ by Eq. (67) with

$$M_1 = \begin{bmatrix} (U^{(k)})^T (Y_1^{(k)} + PY_2^{(k)} - \rho PY_3^{(k)}) & -(U^{(k)})^T (Y_4^{(k)} + PY_5^{(k)} - \rho PY_6^{(k)}) \\ -(\nabla f_\epsilon^{(k)})^T (Y_1^{(k)} + PY_2^{(k)} - \rho PY_3^{(k)}) & (\nabla f_\epsilon^{(k)})^T (Y_4^{(k)} + PY_5^{(k)} - \rho PY_6^{(k)}) \end{bmatrix}$$

$$M_2 = \begin{bmatrix} (U^{(k)})^T U^{(k)} & -(U^{(k)})^T \nabla f_\epsilon^{(k)} \\ -(\nabla f_\epsilon^{(k)})^T U^{(k)} & (\nabla f_\epsilon^{(k)})^T \nabla f_\epsilon^{(k)} \end{bmatrix}$$
 - 9: Update $U^{(k+1)} = U^{(k)} - \delta^{(k)} \nabla f_\epsilon^{(k)}$
 - 10: **else if** ϵ is given **then**
 - 11: Calculate $\rho^{(k)} = f_\rho(U^{(k)})$
 - 12: Calculate $\nabla f_\rho(U^{(k)}) = \nabla f_\rho^{(k)} = \frac{2(Y_1 + PY_2 - \epsilon U)}{Y_1^T Y_1} - 2f_\rho(U) \frac{Y_3}{Y_1^T Y_1}$
 - 13: Simulate to get $Y_4^{(k)}, Y_5^{(k)}$, and $Y_6^{(k)}$ under input $\nabla f_\rho^{(k)}, P\nabla f_\rho^{(k)}$ and $PY_4^{(k)}$
 - 14: Calculate the step size $\delta^{(k)}$ by Eq. (67) with

$$M_1 = \begin{bmatrix} (U^{(k)})^T (Y_1^{(k)} + PY_2^{(k)} - \epsilon U^{(k)}) & -(U^{(k)})^T (Y_4^{(k)} + PY_5^{(k)} - \epsilon \nabla f_\rho^{(k)}) \\ -(\nabla f_\rho^{(k)})^T (Y_1^{(k)} + PY_2^{(k)} - \epsilon U^{(k)}) & (\nabla f_\rho^{(k)})^T (Y_4^{(k)} + PY_5^{(k)} - \epsilon \nabla f_\rho^{(k)}) \end{bmatrix}$$

$$M_2 = \begin{bmatrix} (U^{(k)})^T PY_3 & -(U^{(k)})^T PY_6 \\ -(\nabla f_\rho^{(k)})^T PY_3 & (\nabla f_\rho^{(k)})^T PY_6 \end{bmatrix}$$
 - 15: Update $U^{(k+1)} = U^{(k)} - \delta^{(k)} \nabla f_\rho^{(k)}$
 - 16: **end if**
 - 17: Run simulation to get $Y_1^{(k+1)}$ under input $U^{(k+1)}$
 - 18: **end for**
-

It follows from Eq. (59) that

$$f(U - \delta \nabla f(U)) = \frac{[1 \ \delta] M_1 [1 \ \delta]^T}{[1 \ \delta] M_2 [1 \ \delta]^T} = \frac{M_{1,11} + 2M_{1,12}\delta + M_{1,22}\delta^2}{M_{2,11} + 2M_{2,12}\delta + M_{2,22}\delta^2} \quad (65)$$

and that

$$\begin{aligned} & \frac{1}{2} (M_{2,11} + 2M_{2,12}\delta + M_{2,22}\delta^2)^2 \frac{\partial f}{\partial \delta} \\ &= (M_{1,12}M_{2,11} - M_{2,12}M_{1,11}) + (M_{1,22}M_{2,11} - M_{2,22}M_{1,11})\delta \\ &+ (M_{1,22}M_{2,12} - M_{2,22}M_{1,12})\delta^2 \\ &\triangleq c + b\delta + a\delta^2, \end{aligned} \quad (66)$$

from which optimal value δ^* can be solved by setting the above expression equal to zero. That is,

$$\delta^* = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}, \quad (67)$$

in which the (smaller) positive solution should be chosen.

Author details


Kenneth McDonald¹, Zhihua Qu^{1*} and Azwirman Gusrialdi²

1 University of Central Florida, Orlando, United States

2 Tampere University, Tampere, Finland

*Address all correspondence to: qu@ucf.edu

IntechOpen

© 2024 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/4.0>), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. 

References

- [1] Kailath T. Linear Systems. Englewood Cliffs, NJ: Prentice Hall; 1980
- [2] Ogata K. Modern Control Engineering. Englewood Cliffs, NJ: Prentice Hall; 1990
- [3] Khalil HK. Nonlinear Systems. Upper Saddle River, NJ: Prentice Hall; 2002
- [4] Willems JC. Dissipative dynamical systems part i: General theory. Archive for Rational Mechanics and Analysis. 1972;**45**:321-351. DOI: 10.1007/BF00276493
- [5] Qu Z. Cooperative Control of Dynamical Systems. London: Springer Verlag; 2009
- [6] Zhihua Q, Simaan MA. Modularized design for cooperative control and plug-and-play operation of networked heterogeneous systems. Automatica. 2014;**50**(9):2405-2414
- [7] Poonawala HA, Spong MW. Decentralized estimation of the algebraic connectivity for strongly connected networks. In: American Control Conference. Chicago, IL, USA: IEEE; 2015. pp. 4068-4073
- [8] Gusrialdi A, Zhihua Q, Hirche S. Distributed link removal using local estimation of network topology. IEEE Transactions on Network Science and Engineering. 2019;**6**(3):280-292
- [9] Alizadeh R, Bijani S, Shakeri F. Distributed consensus-based estimation of the leading eigenvalue of a non-negative irreducible matrix. Parallel Computing. 2024;**122**:103113. DOI: 10.1016/j.parco.2024.103113
- [10] Charalambous T, Rabbat MG, Johansson M, Hadjicostis CN. Distributed finite-time computation of digraph parameters: Left-eigenvector, out-degree and spectrum. IEEE Transactions on Control of Network Systems. 2015;**3**(2):137-148
- [11] Fernández OD, Tiistola S, Gusrialdi A. Real-time data-driven electromechanical oscillation monitoring using dynamic mode decomposition with sliding window. In: 11th IFAC Symposium on Control of Power and Energy Systems. IFAC-PapersOnLine; 2022. pp. 158-163
- [12] Fernández OD, Iqbal M, Gusrialdi A. An improved dynamic mode decomposition for real-time electromechanical oscillation monitoring in power systems: The impact of ultra-low frequency modes and its removal strategy. IET Generation Transmission and Distribution. 2023;**17**(20):4574-4591
- [13] Gusrialdi A, Zhihua Q. Distributed data-driven power iteration for strongly connected networks. In: European Control Conference. Delft, Netherlands: IEEE; 2021. pp. 87-92
- [14] Khazaei J, Fan L, Jiang W, Manjure D. Distributed prony analysis for real-world pmu data. Electric Power Systems Research. 2016;**133**:113-120
- [15] Deplano D, Congiu C, Giua A, Franceschelli M. Distributed estimation of the laplacian spectrum via wave equation and distributed optimization. In: 22nd IFAC World Congress. Yokohama, Japan: IFAC-PapersOnLine; 2023. pp. 6952-6957
- [16] Hayhoe M, Barreras F, Preciado VM. A dynamical approach to efficient eigenvalue estimation in general multiagent networks. Automatica. 2022; **140**:110234
- [17] Venkateswaran DB, Qu Z. Passivity-short bilateral teleoperation with

- communication delays. In: IEEE International Conference on Systems, Man, and Cybernetics. Miyazaki, Japan: IEEE; 2018. pp. 1275-1281
- [18] Joo Y, Harvey R, Zhihua Q. Preserving and achieving passivity-short property through discretization. IEEE Transactions on Automatic Control. 2020;**65**(10):4265-4272. DOI: 10.1109/TAC.2019.2954361
- [19] Tanemura M, Azuma S-i. Efficient data-driven estimation of passivity properties. IEEE Control Systems Letters. 2019;**3**(2):398-403. DOI: 10.1109/LCSYS.2018.2887241
- [20] Gusrialdi A, Zhihua Q. Distributed estimation of all the eigenvalues and eigenvectors of matrices associated with strongly connected digraphs. IEEE Control Systems Letters. 2017;**1**(2):328-333
- [21] Nejad BM, Attia SA, Raisch J. Max-consensus in a max-plus algebraic setting: The case of fixed communication topologies. In: International Symposium on Information, Communication and Automation Technologies. Sarajevo, Bosnia and Herzegovina: IEEE; 2009. pp. 1-7
- [22] Liu J, Mou S, Stephen Morse A. Asynchronous distributed algorithms for solving linear algebraic equations. IEEE Transactions on Automatic Control. 2018;**63**(2):372-385
- [23] Asadi MM, Khosravi M, Aghdam AG, Blouin S. Generalized algebraic connectivity for asymmetric networks. In: American Control Conference. Boston, MA, USA: IEEE; 2016. pp. 5531-5536
- [24] Zareh M, Sabattini L, Secchi C. Distributed Laplacian Eigenvalue and Eigenvector Estimation in Multi-Robot Systems. Cham: Springer International Publishing; 2018. pp. 191-204
- [25] Liu J, Gusrialdi A, Hirche S, Monti A. Joint controller-communication topology design for distributed wide-area damping control of power systems. In: 18th IFAC World Congress. Milano, Italy: IFAC-PapersOnLine; 2011. pp. 519-525
- [26] Moroşan P-D, Bourdais R, Dumur D, Buisson J. Building temperature regulation using a distributed model predictive control. Energy and Buildings. 2010;**42**(9):1445-1452
- [27] Chow JH. Power System Coherency and Model Reduction. Vol. 84. New York, NY: Springer; 2013
- [28] Gusrialdi A, Qu Z. Data-driven distributed algorithms for estimating eigenvalues and eigenvectors of interconnected dynamical systems. In: Proceedings of 21st IFAC World Congress. Berlin, Germany: IFAC-PapersOnLine; 2020. pp. 52-57
- [29] Gusrialdi A, Chakraborty A, Zhihua Q. Distributed learning of mode shapes in power system models. In: Proceedings of IEEE Conference on Decision and Control. Miami, FL: IEEE; 2018. pp. 4002-4007
- [30] Chow J, Kokotovic P. Time scale modeling of sparse dynamic networks. IEEE Transactions on Automatic Control. 1985;**30**(8):714-722
- [31] Koch A, Montenbruck JM, Allgöwer F. Sampling strategies for data-driven inference of input-output system properties. IEEE Transactions on Automatic Control. 2021;**66**(3):1144-1159. DOI: 10.1109/TAC.2020.2994894
- [32] Smaili YA, Alouani AT. An H/sub infinity / governor exciter controller design for a power system. In: [Proceedings 1992] The First IEEE Conference on Control Applications. Vol. 2. Dayton, OH, USA: IEEE; 1992. pp. 770-775. DOI: 10.1109/CCA.1992.269750