# Distributed Optimization Algorithms for Wide-Area Oscillation Monitoring in Power Systems Using Interregional PMU-PDC Architectures

**Article** *in* IEEE Transactions on Smart Grid · September 2015

DOI: 10.1109/TSG.2015.2406578

CITATIONS
42

READS
208

**3 authors**, including:

Seyedbehzad Nabavi
North Carolina State University
**13** PUBLICATIONS **121** CITATIONS

SEE PROFILE

Jianhua Zhang
National Renewable Energy Laboratory
**10** PUBLICATIONS **167** CITATIONS

SEE PROFILE

# Distributed Optimization Algorithms for Wide-Area Oscillation Monitoring in Power Systems Using Interregional PMU-PDC Architectures

Seyedbehzad Nabavi, *Student Member, IEEE*, Jianhua Zhang, *Student Member, IEEE*, and Aranya Chakrabortty, *Senior Member, IEEE*

***Abstract*—In this paper, we present a set of distributed algorithms for estimating the electro-mechanical oscillation modes of large power system networks using synchrophasors. With the number of phasor measurement units (PMUs) in the North American grid scaling up to the thousands, system operators are gradually inclining toward distributed cyber-physical architectures for executing wide-area monitoring and control operations. Traditional centralized approaches, in fact, are anticipated to become untenable soon due to various factors such as data volume, security, communication overhead, and failure to adhere to real-time deadlines. To address this challenge, we propose three different communication and computational architectures by which estimators located at the control centers of various utility companies can run local optimization algorithms using local PMU data, and thereafter communicate with other estimators to reach a global solution. Both synchronous and asynchronous communications are considered. Each architecture integrates a centralized Prony-based algorithm with several variants of alternating direction method of multipliers (ADMM). We discuss the relative advantages and bottlenecks of each architecture using simulations of IEEE 68-bus and IEEE 145-bus power system, as well as an Exo-GENI-based software defined network.**

***Index Terms*—Alternating direction method of multipliers (ADMM), distributed optimization, phasor measurement units (PMUs), Prony, wide area oscillation monitoring.**

## I. Introduction

FOLLOWING the Northeast blackout of 2003, wide-area measurement system (WAMS) technology using phasor measurement units (PMUs) has largely matured for the North American grid [1]. However, as the number of PMUs scales up to the thousands in the next few years under the U.S. Department of Energy's smart grid demonstration initiative, independent system operators (ISOs) and utility companies are struggling to understand how the resulting gigantic volumes of real-time data can be efficiently harvested, processed, and utilized to solve wide-area monitoring and control problems for any realistic power system interconnection. It is intuitive that the current state-of-the-art centralized communication and information processing architecture of WAMS will no longer be sustainable under such a data explosion, and a completely distributed cyber-physical architecture will need to be developed [2]. In the Eastern Interconnection (EI) of the U.S. grid, for example, about 60 PMUs are currently streaming data via the Internet to a super phasor data concentrator (SPDC) which is handling about 100 000 data points per second. This architecture will no doubt become untenable as the EI scales up to more than 500 PMUs in the next few years. Research is currently being carried out by the data and network management task team of North American Synchrophasor Initiative (NASPI) on the implementation of this distributed architecture with the prime research focus being protocols, quality-of-service, latency, bandwidth, and security [3].

However, almost no attention has yet been paid to perhaps the most critical consequence of this envisioned distributed architecture—namely distributed algorithms [4]. Partly due to a lack of a cyber-physical research infrastructure and partly due to the priorities set forth by PMU installations, the NASPI community has not yet delved into investigating how the currently-used centralized algorithms for wide-area monitoring and control [5] can be translated into a distributed computing framework once the aforementioned decentralized WAMS architecture is realized in the next three to four years. Development of such algorithms will obviously be imperative not only for increasing reliability by eliminating single-point failures, but also for minimizing network transit. As shown in [6], transmitting data across a wide-area communication network is expensive, the links can be relatively slow, and the bandwidth-per-dollar will indeed grow slower than other computing resources leading to distributed PMU data processing as a natural choice.

Motivated by this challenge, in this paper, we propose three different distributed communication and computational architectures for one of the most critical wide-area monitoring applications, namely, modal estimation of electro-mechanical oscillations. Several centralized algorithms for solving this problem have been proposed over the past decade including the eigenvalue realization algorithm [7], Prony analysis [8], mode metering [9], and Hilbert–Huang transform [10]. However, all of these algorithms are based on centralized and offline techniques, and that too using only a handful (but observable) set of PMUs. In contrast, we formulate the mode estimation

Manuscript received May 15, 2014; revised September 22, 2014; accepted December 28, 2014. Date of publication March 10, 2015; date of current version August 19, 2015. This work was supported in part by the U.S. Department of Energy under Grant DE-OE 0000654, and in part by ABB Corporate Research. Paper no. TSG-00455-2014.

The authors are with the Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, NC 27606 USA (e-mail: snabavi@ncsu.edu; jzhang25@ncsu.edu; and achakra2@ncsu.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSG.2015.2406578

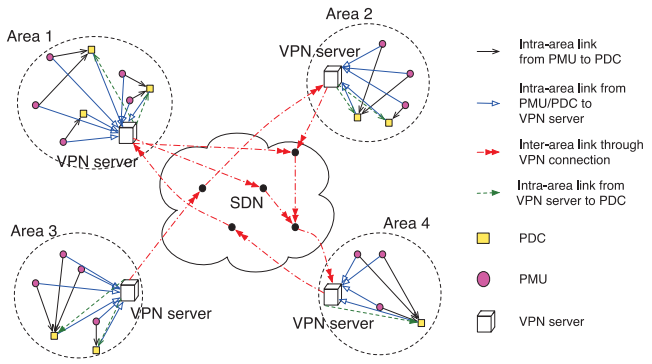


Fig. 1. Distributed architecture for wide-area PMU-PDC communications.

problem as a global consensus problem for the coefficients of the characteristic polynomial of the system, and then solve it using alternating direction method of multipliers (ADMM). The communication network required to execute this distributed estimation is shown in Fig. 1. The physical grid is assumed to be divided into multiple balancing regions or areas, which may or may not be coherent, but belong to different utility companies. PMUs in each area communicate their data in real-time to estimator(s) or Phasor Data Collectors (PDCs) located at the local control center via a virtual private network. These local PDCs can then share information between each other and also with a central PDC located at the ISO through a controllable wide-area network such as a software defined network (SDN). The key idea is to make use of this distributed network protocol to run local consensus at the PDCs inside each area, iteratively generate myopic estimates of the coefficients of the characteristic polynomial, and let the PDCs communicate either with each other or with the central PDC to reach a global solution using several variants of the ADMM [11]. Both synchronous and asynchronous communications are considered. Our proposed framework clearly demonstrates how ADMM can be a beneficial tool for distributed mode estimation in power systems, and what types of performance bottlenecks, accuracy issues, and computation delays it may result in. The innovation of this paper is, therefore, in proposing a bridge between the cyber and physical implementation of distributed WAMS. We illustrate our algorithms via offline and real-time simulations of three IEEE prototype power system models, and discuss the benefits and drawbacks of each algorithm in light of security and data privacy.

Distributed consensus algorithms in power systems have been reported in recent papers such as [12]–[16], but mostly in the context of distributed optimal power flow, distributed generation, and demand-side management, and not for wide-area oscillation monitoring. Preliminary results on the first architecture proposed in this paper have recently been reported in [17] and [18]. However, the results outlined in this paper are significantly more expansive than those in [17] and [18] including two new architectures with unique sets of distributed algorithms, a discussion of their convergence properties, the pertinent issue of asynchronous communication in real-world SDNs, and finally a case study of end-to-end delay evaluation using a U.S.-wide Exo-GENI network.

The remainder of this paper is organized as follows. Section II presents the power system model of interest. Section III describes the centralized Prony method. Section IV proposes the distributed modal estimation strategies. Sections V and VI show the simulation results. Section VII concludes this paper.

## II. Problem Formulation

We consider a power system consisting of $n$ generator buses and $n_l$ load buses. Each synchronous generator is modeled by a second-order swing equation, while each load bus is modeled by two algebraic equations for active and reactive power balance. We convert this differential-algebraic model to a completely differential model using standard techniques of Kron reduction [19] and arrive at a linearized state variable model for the $n$-machine system as

$$\begin{bmatrix} \Delta\dot{\delta}(t) \\ \Delta\dot{\omega}(t) \end{bmatrix} = \underbrace{\left[\begin{array}{c|c} \mathbf{0}_{n\times n} & \omega_s I_n \\ \hline \mathcal{M}^{-1}L & -\mathcal{M}^{-1}\mathcal{D} \end{array}\right]}_{A} \begin{bmatrix} \Delta\delta(t) \\ \Delta\omega(t) \end{bmatrix}$$

$$\mathbf{y}(t) = [\Delta\theta_1(t), \ldots, \Delta\theta_p(t)]^T \tag{1}$$

where $\Delta\delta = [\Delta\delta_1 \ \cdots \ \Delta\delta_n]^T$, $\Delta\omega = [\Delta\omega_1 \ \cdots \ \Delta\omega_n]^T$, $\mathcal{M} = \text{diag}(M_1, \ldots, M_n)$, and $\mathcal{D} = \text{diag}(D_1, \ldots, D_n)$, with $\Delta\delta_i$, $\Delta\omega_i$, $M_i$, and $D_i$ being the small-signal angle deviation, the small-signal frequency deviation, inertia, and mechanical damping of generator $i$, respectively. $I_n$ is the $(n \times n)$ identity matrix, and $\omega_s$ is the synchronous speed of the system. The definition of matrix $L$ is referred to in [17]. We consider the output vector $\mathbf{y}(t) \in \mathbb{R}^p$ to be a set of phase angle measurements $\Delta\theta_i(t)$, $i = 1, \ldots, p$, measured by PMUs at $p$ designated buses. Other outputs such as bus voltages and frequencies may also be considered but we restrict our analysis to phase angles only. The eigenvalues of $A$ are denoted by $(-\sigma_l \pm j\Omega_l)$, $(j \triangleq \sqrt{-1})$ $l = 1, \ldots, n$. Our objective is to estimate these $2n$ eigenvalues of $A$ from $\mathbf{y}(t)$ in a distributed fashion using multiple computational resources. For this purpose, we next describe how the commonly used Prony algorithm for modal estimation can be cast as a distributed optimization problem. We first recall the centralized Prony algorithm, and thereafter reformulate it as a distributed algorithm using three different cyber-physical architectures.

## III. Modal Estimation Using Prony Method

A generic expression for the solution of $\Delta\theta_i(t)$ in (1) can be written as

$$\Delta\theta_i(t) = \sum_{l=1}^{n} \left( r_{il} e^{(-\sigma_l + j\Omega_l)t} + r_{il}^* e^{(-\sigma_l - j\Omega_l)t} \right). \tag{2}$$

Each component in the right-hand side of (2) is referred to as a mode, where $r_{il}$ is the residue of mode $l$ reflected in the $i$th output. Sampling $\Delta\theta_i(t)$ with a uniform sampling period of $T$, a generic expression for the $z$-transform of $\Delta\theta_i(k) \triangleq \Delta\theta_i(t)|_{t=kT}$, $(k = 0, 1, \ldots, m-1)$, with $m$ being the total number of measured samples, can be rewritten as

$$\Delta\theta_i(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2} + \cdots + b_{2n} z^{-2n}}{1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_{2n} z^{-2n}}. \tag{3}$$

We recall that our objective is to estimate $\sigma_l$, $\Omega_l$, and $r_{il}$ in (2). We next state three steps of the Prony algorithm by which this can be achieved from (3) [8].

*Step 1:* The first step of the Prony algorithm is to find $a_1$ through $a_{2n}$ by solving

$$\underbrace{\begin{bmatrix} \Delta\theta_i(2n) \\ \Delta\theta_i(2n+1) \\ \vdots \\ \Delta\theta_i(2n+\ell) \end{bmatrix}}_{\mathbf{c}_i} = \underbrace{\begin{bmatrix} \Delta\theta_i(2n-1) & \cdots & \Delta\theta_i(0) \\ \Delta\theta_i(2n) & \cdots & \Delta\theta_i(1) \\ \vdots & & \vdots \\ \Delta\theta_i(2n+\ell-1) & \cdots & \Delta\theta_i(\ell) \end{bmatrix}}_{H_i} \underbrace{\begin{bmatrix} -a_1 \\ -a_2 \\ \vdots \\ -a_{2n} \end{bmatrix}}_{\boldsymbol{a}} \tag{4}$$

where $\ell$ is an integer satisfying $2n + \ell \leq m - 1$. Let us concatenate $\mathbf{c}_i$ and $H_i$ in (4) for PMU $i = 1, \ldots, p$. One can find $\boldsymbol{a}$ by solving a least squares (LS) problem defined as

$$\min_{\boldsymbol{a}} \frac{1}{2} \left\| \begin{bmatrix} H_1 \\ \vdots \\ H_p \end{bmatrix} \boldsymbol{a} - \begin{bmatrix} \mathbf{c}_1 \\ \vdots \\ \mathbf{c}_p \end{bmatrix} \right\|^2 \tag{5}$$

where $\|\cdot\|$ denotes the 2-norm of a vector.

*Step 2:* Once $\boldsymbol{a}$ is computed, the next step is to find the roots of the discrete-time characteristic polynomial as shown in the denominator of (3). Let these roots be denoted by $\bar{z}_l$, $l = 1, \ldots, 2n$. Finally, the eigenvalues of $A$ in (1) can be calculated as $\ln(\bar{z}_l)/T$.

*Step 3:* The final step is to find the residues $r_{il}$ in (2). This can be done by forming the following Vandermonde equation, and solving it for the residues $r_{i1}$ through $r_{in}$:

$$\begin{bmatrix} \Delta\theta_i(0) \\ \Delta\theta_i(1) \\ \vdots \\ \Delta\theta_i(m) \end{bmatrix} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ (\bar{z}_1)^{1/T} & (\bar{z}_2)^{1/T} & \cdots & (\bar{z}_{2n})^{1/T} \\ \vdots & \vdots & & \vdots \\ (\bar{z}_1)^{m/T} & (\bar{z}_2)^{m/T} & \cdots & (\bar{z}_{2n})^{m/T} \end{bmatrix} \begin{bmatrix} r_{i1} \\ r_{i1}^* \\ \vdots \\ r_{in} \\ r_{in}^* \end{bmatrix}. \tag{6}$$

*Note 1:* The method cannot estimate the order of the system $2n$. If $n$ is not known *a priori*, it is usually considered to be a large number, and thereafter the modes with negligible residues are discarded. However, $n$ is limited by the number of available measurements as well as the computational memory.

*Note 2:* The method can be performed in real-time. That is, one may solve (5) while iteratively updating $H_i$, and $\mathbf{c}_i$ as new measurements become available.

## IV. Proposed Strategies for Distributed Modal Estimation

In this section, we show how Step 1 of the centralized approach delineated in Section III can be recast as a distributed optimization problem using intra- and interregional PMU-PDC architectures. We wish to clarify at the very outset that the term PDC in our algorithms is used in a much broader sense, and not just as a data aggregator. It essentially refers to any computing agent that can process PMU data and run algorithms on them. In other words, as long as the data from a given PMU get communicated to a computing station, whether it be a hardware PDC, software PDC, a local server, or even a data center—we would collectively refer to all of these processing units as a PDC for convenience. We assume the Cyber-Physical Systems infrastructure of the grid to be divided into five distinct layers. The lowest layer contains the stochastic variations in loads and events due to nature and human activities. The second level is the physical power system model. The third and fourth layers consist of real-time PMU data processing and computation at the substation level and the control center level, respectively. Between the second, third, and fourth layers, data communication architecture is configured to handle massive amounts of PMU data using the network shown in Fig. 1. The topmost layer is the application layer, which for our purpose is estimation of the eigenvalues of the matrix $A$ in (1) using the distributed communication protocol of this network. If necessary, an internal hierarchy of multiple area-level decentralization layers can also be created (in fact, our third architecture is based on this assumption). Too many sub-layers, however, can lead to unacceptable processing delays and latency violations. Based on these intuitions, we next describe our proposed computational architectures.

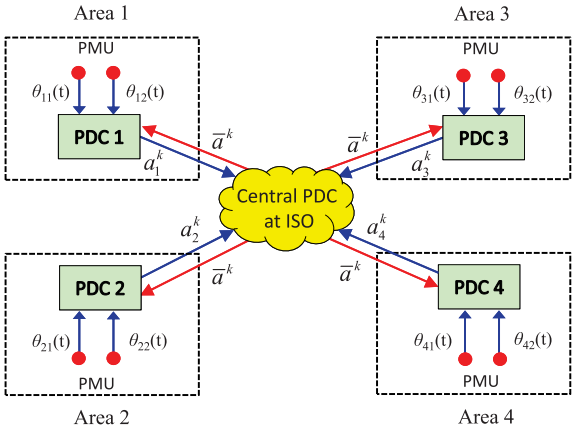


Fig. 2. Architecture 1 using S-ADMM for a 4-area network.

### A. Architecture 1: Distributed Prony Using Standard ADMM

The LS problem (5) can be regarded as a global consensus problem over a network of $N$ utility companies or areas. We assume every area to be equipped with one aggregated PDC as shown in Fig. 2. The consensus problem can be described as

$$\begin{aligned} \underset{\boldsymbol{a}_1, \ldots, \boldsymbol{a}_N, \mathbf{z}}{\text{minimize}} \quad & \sum_{j=1}^{N} \frac{1}{2} \|\hat{H}_j \boldsymbol{a}_j - \hat{\mathbf{c}}_j\|^2 \\ \text{subject to} \quad & \boldsymbol{a}_j - \mathbf{z} = 0, \quad \text{for } j = 1, \ldots, N. \end{aligned} \tag{7}$$

Here, $\hat{H}_j \triangleq [H_{j,1}^T \; H_{j,2}^T \; \cdots \; H_{j,N_j}^T]^T$, $\hat{\mathbf{c}}_j \triangleq [\mathbf{c}_{j,1}^T \; \mathbf{c}_{j,2}^T \; \cdots \; \mathbf{c}_{j,N_j}^T]^T$, where $N_j$ is the total number of PMUs in Area $j$, and $H_{j,i}$ and $\mathbf{c}_{j,i}$ are constructed as in (4) from the time samples of $\Delta\theta_{j,i}$, which is the $i$th PMU measurement in Area $j$, $i = 1, \ldots, N_j$. The global consensus solution, denoted by $\mathbf{z} \in \mathbb{R}^{2n}$, is the solution of (5) that is obtained when the local estimates of the $N$ regional PDCs, denoted by $\boldsymbol{a}_j$, $j = 1, \ldots, N$, reach the same value.

The standard ADMM (S-ADMM) estimation method uses the Lagrangian multiplier approach to solve (7) in an iterative, distributed way. Following [11] the augmented Lagrangian for (7) can be computed as

$$L_\rho = \sum_{j=1}^{N}\left(\frac{1}{2}\left\|\hat{H}_j\boldsymbol{a}_j - \hat{\mathbf{c}}_j\right\|^2 + \mathbf{w}_j^T(\boldsymbol{a}_j - \mathbf{z}) + \frac{\rho}{2}\left\|\boldsymbol{a}_j - \mathbf{z}\right\|^2\right) \quad (8)$$

where $\boldsymbol{a}_j$ and $\mathbf{z}$ are the vectors of the primal variables, $\mathbf{w}_j$ is the vector of the dual variables or the Lagrange multipliers associated with (7), and $\rho > 0$ denotes a penalty factor.

Before the S-ADMM algorithm starts, the central PDC fixes the order of the system $2n$, and the initial height of the Hankel blocks $(\ell + 1)$ for all local PDCs as shown in (4). Each local PDC $j$ (i.e., the PDC located in Area $j$) then waits until the $(2n + \ell)$th sample of the measurement arrives. In order to ensure the real-time nature of the algorithm, at iteration $k$, each local PDC $j$ constructs the matrices $\hat{H}_j^k$ and $\hat{\mathbf{c}}_j^k$ as follows:

$$\hat{H}_j^k \triangleq \left[\left(H_{j,1}^k\right)^T \cdots \left(H_{j,N_j}^k\right)^T\right]^T,\ \hat{\mathbf{c}}_j^k \triangleq \left[\left(\mathbf{c}_{j,1}^k\right)^T \cdots \left(\mathbf{c}_{j,N_j}^k\right)^T\right]^T \quad (9)$$

where

$$H_{j,i}^k \triangleq \begin{bmatrix} \Delta\theta_{j,i}(2n-1) & \cdots & \Delta\theta_{j,i}(0) \\ \Delta\theta_{j,i}(2n) & \cdots & \Delta\theta_{j,i}(1) \\ \vdots & & \vdots \\ \Delta\theta_{j,i}(m^k-1) & \cdots & \Delta\theta_{j,i}(m^k-2n) \end{bmatrix} \quad (10a)$$

$$\mathbf{c}_{j,i}^k \triangleq \left[\Delta\theta_{j,i}(2n)\ \Delta\theta_{j,i}(2n+1) \cdots \Delta\theta_{j,i}\left(m^k\right)\right]^T \quad (10b)$$

for $i = 1, \ldots, N_j$. Here, $\Delta\theta_{j,i}(m^k)$ is the most recent measurement sample available to PDC $j$ from the $i$th PMU in its area at iteration $k$. Using (9) and (10), the S-ADMM algorithm is illustrated in Algorithm 1.

Since (7) is a convex optimization problem, as $k \to \infty$, $\mathbf{z}^k$ in (12) converges to the global minimum of (7), and so does each individual $\boldsymbol{a}_j^k$ due to consensus. Let these optimal values be denoted as $\mathbf{z}^*$ and $\boldsymbol{a}_j^*$. Once $\mathbf{z}^* = \boldsymbol{a}_1^* = \cdots = \boldsymbol{a}_N^*$ is calculated, every local PDC can compute the eigenvalues of $A$ using Step 2 of the Prony algorithm described in Section III. It can also compute the mode residues using Step 3.

*Note:* The conventional ADMM algorithm for consensus problems converges at the rate of $\mathcal{O}(1/k)$, as shown in [11]. However, unlike [11], where the objective function to be minimized is assumed to be time-invariant, (11) in our proposed S-ADMM varies over time. If we hold $\hat{H}_j^k$ and $\hat{\mathbf{c}}_j^k$ to be constant at $\hat{H}_j^0$ and $\hat{\mathbf{c}}_j^0$, respectively, for all $j$ and $k$, then the $\mathcal{O}(1/k)$ convergence rate can be guaranteed.

### *B. Architecture 1 With Asynchronous Communication*

One assumption behind Architecture 1 is that all local PDCs are performing their respective optimization steps with equal speed, and the communication delays between the local PDCs and the central PDC are also equal, i.e., they are synchronous. However, in reality, the PDCs may not be perfectly synchronized with each other due to differences in their processing speeds as well as due to various communication delays such as routing, queuing, and transfer delays in the SDN shown in Fig. 1. One possible solution to overcome this asynchrony is to force the central PDC to wait until it receives data from all local PDCs. In that case, the total end-to-end delay for each iteration will be dependent on the slowest communication link, and hence the entire algorithm may become very slow. An alternative approach would be to use an asynchronous version of Algorithm 1. Motivated by the results in [20], [21], we next state a variant of asynchronous ADMM (A-ADMM). In this method, the central PDC receives the updates only from a subset of the $N$ local PDCs at every iteration $k$, referred to as active PDCs. Let this set be denoted by $\mathcal{S}^k$. It then calculates $\mathbf{z}^{k+1}$ using the most recent local estimates from all PDCs. Let $T^{k+1}$ be the time instant at which $\mathbf{z}^{k+1}$ is computed. The central PDC then broadcasts $(\mathbf{z}^{k+1}, T^{k+1})$ to every local PDC. Upon receiving $T^{k+1}$, each local PDC $j$ then constructs $\hat{H}_j^{k+1}$ and $\hat{\mathbf{c}}_j^{k+1}$ from (9) by setting $m^{k+1}$ to be the sample index that is closest to the time instant $T^{k+1}$. Note that $\Delta\theta(m^{k+1})$ may not be the most recent measurement sample while constructing $\hat{H}_j^{k+1}$ and $\hat{\mathbf{c}}_j^{k+1}$ matrices. However, in order to ensure that all PDCs use the same time-window of the measurements to form $\hat{H}_j^{k+1}$ and $\hat{\mathbf{c}}_j^{k+1}$, they all use the same value of $m^{k+1}$ as decided globally by the central PDC at every iteration $k+1$. The A-ADMM algorithm adapted for (7) is shown in Algorithm 2.

Note that the iteration numbers $k$ and $k+1$ are communicated between the PDCs in Steps 3c) and 3f) to keep track of the order of the receiving data. This architecture is similar to Architecture 1 shown in Fig. 2. The only difference is that more information is exchanged between the central and local PDCs. The aforementioned A-ADMM algorithm converges to the minimizer of (7) with the rate of $\mathcal{O}(1/k)$ if none of the local PDCs is allowed to be dormant all throughout. In other words, for all $\mathcal{S}^k \in \mathcal{P}$, where $\mathcal{P} \triangleq 2^{\{1,\ldots,N\}}$ is the

**Algorithm 1** Distributed Prony Using S-ADMM

1) Each PDC $j$ initializes $\boldsymbol{a}_j^0$, $\mathbf{z}^0$, and $\mathbf{w}_j^0$, $j = 1, \ldots, N$.
2) At iteration $k$:
   a) PDC $j$ constructs $\hat{H}_j^k$ and $\hat{\mathbf{c}}_j^k$ from (9).
   b) PDC $j$ updates $\boldsymbol{a}_j$ as

$$\begin{aligned}\boldsymbol{a}_j^{k+1} &= \arg\min_{\boldsymbol{a}_j} L_\rho \\ &= \left(\left(\hat{H}_j^k\right)^T\hat{H}_j^k + \rho I_{2n}\right)^{-1} \\ &\quad \left(\left(\hat{H}_j^k\right)^T\hat{\mathbf{c}}_j^k - \mathbf{w}_j^k + \rho\mathbf{z}^k\right). \end{aligned} \quad (11)$$

   c) PDC $j$ transmits $\boldsymbol{a}_j^{k+1}$ to the central PDC.
   d) Central PDC calculates

$$\mathbf{z}^{k+1} \triangleq \bar{\boldsymbol{a}}^{k+1} = \frac{1}{N}\sum_{j=1}^{N}\boldsymbol{a}_j^{k+1}. \quad (12)$$

   e) Central PDC broadcasts $\mathbf{z}^{k+1}$ to all local PDCs.
   f) PDC $j$ updates $\mathbf{w}_j$ as

$$\mathbf{w}_j^{k+1} = \mathbf{w}_j^k + \rho\left(\boldsymbol{a}_j^{k+1} - \mathbf{z}^{k+1}\right). \quad (13)$$

**Algorithm 2** Distributed Prony Using A-ADMM

1) The central PDC initializes $T^0$ and sends it to all local PDCs.
2) PDC $j$ initializes $\boldsymbol{a}_j^0$, $\mathbf{z}^0$, and $\mathbf{w}_j^0$, for $j = 1, \ldots, N$.
3) At iteration $k$.
   a) Given $T^k$, PDC $j$ constructs $\hat{H}_j^k$ and $\hat{\mathbf{c}}_j^k$ from (9) using $m^k$ decided from $T^k$.
   b) PDC $j$ updates $\boldsymbol{a}_j$ as

$$\boldsymbol{a}_j^{k+1} = \left(\left(\hat{H}_j^k\right)^T \hat{H}_j^k + \rho I_{2n}\right)^{-1} \left(\left(\hat{H}_j^k\right)^T \hat{\mathbf{c}}_j^k - \mathbf{w}_j^k + \rho \mathbf{z}^k\right).$$

   c) PDC $j$ transmits $\boldsymbol{a}_j^{k+1}$, $\mathbf{w}_j^k$, and $k$ to the central PDC.
   d) The central PDC receives the values of $\boldsymbol{a}_j^{k+1}$, $\mathbf{w}_j^k$, and $k$ only from the active PDCs $j \in \mathcal{S}^k$.
   e) The central PDC updates $\mathbf{z}$ as

$$\mathbf{z}^{k+1} = \frac{1}{N}\sum_{j=1}^{N}\left(\boldsymbol{a}_j^{k+1} + (1/\rho)\mathbf{w}_j^k\right),$$

   where $\boldsymbol{a}_j^{k+1} = \boldsymbol{a}_j^k$, and $\mathbf{w}_j^k = \mathbf{w}_j^{k-1}$ for $j \notin \mathcal{S}^k$.
   f) The central PDC broadcasts $\mathbf{z}^{k+1}$, $k+1$, and $T^{k+1}$ to all local PDCs.
   g) PDC $j$ updates $\mathbf{w}_j$ as

$$\mathbf{w}_j^{k+1} = \mathbf{w}_j^k + \rho\left(\boldsymbol{a}_j^{k+1} - \mathbf{z}^{k+1}\right), \quad j \in \mathcal{S}^k,$$
$$\mathbf{w}_j^{k+1} = \mathbf{w}_j^k, \quad j \notin \mathcal{S}^k.$$

set of all subsets of $N$ PDCs, $\mathcal{S}^k$ must be active infinitely often with probability 1 [20]. By virtue of this algorithm, the real-time nature for solving (7) can still be maintained despite asynchronous delays in the SDN.

### *C. Architecture 2: Distributed Prony Using Distributed ADMM*

Consider again the problem (7). The S-ADMM and the A-ADMM algorithms discussed for Architecture 1 need a central PDC to update $\mathbf{z}$ at each iteration, and to broadcast it back to the local active PDCs. Although this architecture preserves the data privacy between the $N$ PDCs, it is not very resilient as the central PDC is directly amenable to failure under extraneous attacks. This problem can be resolved by resorting to a completely distributed version of Architecture 1 as shown in Fig. 3. The resulting distributed algorithm for solving (7) is is referred to as distributed ADMM, proposed in [22]. We state a variant of that algorithm, and refer to it as D-ADMM, as follows. In this formulation, each active PDC at each iteration communicates directly with a subset of other active PDCs determined by a communication graph $\mathcal{G}$. Therefore, the need for the central PDC no longer exists. The set of nodes of the communication graph $\mathcal{G}$, denoted by $\mathcal{V}(\mathcal{G})$, are the indexed PDCs of the network, i.e., $\mathcal{V}(\mathcal{G}) = \{1, 2, \ldots, N\}$. The edge set of $\mathcal{G}$ is denoted by $\mathcal{E}(\mathcal{G})$, where $e_{jv} \in \mathcal{E}(\mathcal{G})$ determines the existence of a communication link between

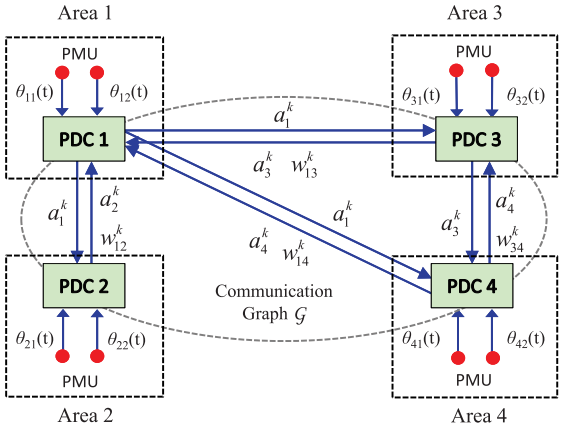


Fig. 3. Architecture 2 using D-ADMM for a given communication graph $\mathcal{G}$ for a 4-area network.

PDCs $j$ and $v$. We consider the following assumptions for the graph $\mathcal{G}$: 1) $\mathcal{G}$ is known *a priori* and is considered to be fixed during the iterations and 2) $\mathcal{G}$ is a simple[1] and connected[2] graph. Before stating the main algorithm, we define an alternative representation of (7) using the communication graph $\mathcal{G}$ as follows:

$$\underset{\boldsymbol{a}_1,\ldots,\boldsymbol{a}_N}{\text{minimize}} \quad \sum_{j=1}^{N}\frac{1}{2}\left\|\hat{H}_j\boldsymbol{a}_j - \hat{\mathbf{c}}_j\right\|^2 \tag{14}$$
$$\text{subject to } \boldsymbol{a}_j - \boldsymbol{a}_v = 0, \text{ for } e_{jv} \in \mathcal{E}(\mathcal{G}).$$

Since $\mathcal{G}$ is connected, then the reformulated problem (14) is equivalent to (7) [22]. Now, let us define two sets of predecessors and successors of PDC $j$, denoted, respectively, by $P_j$ and $S_j$ as

$$P_j = \{\text{PDC } v \mid e_{jv} \in \mathcal{E}(\mathcal{G}),\ v < j\} \tag{15a}$$
$$S_j = \{\text{PDC } v \mid e_{jv} \in \mathcal{E}(\mathcal{G}),\ v > j\}. \tag{15b}$$

Let $n(P_j)$ and $n(S_j)$ denote the number of elements of the sets $P_j$ and $S_j$, respectively. Also, let us define the Lagrangian associated with (14) at iteration $k$ as

$$L_\rho'^k = \frac{1}{2}\sum_{j=1}^{N}\left(\left\|\hat{H}_j^k\boldsymbol{a}_j - \hat{\mathbf{c}}_j^k\right\|^2 + \rho\left(\sum_{v\in P_j}\left\|\boldsymbol{a}_v^{k+1} - \boldsymbol{a}_j - \frac{1}{\rho}\mathbf{w}_{vj}^k\right\|^2 + \sum_{v\in S_j}\left\|\boldsymbol{a}_j - \boldsymbol{a}_v^k - \frac{1}{\rho}\mathbf{w}_{jv}^k\right\|^2\right) - \frac{1}{\rho}\left(\sum_{v\in P_j}\left\|\mathbf{w}_{vj}^k\right\|^2 + \sum_{v\in S_j}\left\|\mathbf{w}_{jv}^k\right\|^2\right)\right) \tag{16}$$

where $\mathbf{w}_{jv}$ is the dual variable associated with the edge $e_{jv} \in \mathcal{E}(\mathcal{G})$, and $\rho > 0$ is the penalty factor. Using these definitions, we present Algorithm 3 for solving (14) using D-ADMM.

At every iteration $k$, the primal variables $\boldsymbol{a}_j^k$ are updated sequentially starting from PDC 1 to $N$ using the most recent available values of $\boldsymbol{a}_v$ for $v$ belonging to its predecessors and

[1] A graph with no self-loop or multiple edges between two nodes.
[2] A graph that has at least one path between any two arbitrary nodes.

**Algorithm 3** Distributed Prony Using D-ADMM

1) Each PDC $j$ initializes $\boldsymbol{a}_j^0$ and $\mathbf{w}_{vj}^0$, $j \in \mathcal{V}(\mathcal{G})$, $e_{vj} \in \mathcal{E}(\mathcal{G})$.
2) At iteration $k$, every PDC $j$, $j = 1, \ldots, N$:
   a) Receives the update of $\boldsymbol{a}_v^{k+1}$ for all $v \in P_j$.
   b) Updates $\hat{H}_j$ and $\hat{\mathbf{c}}_j$ using (9).
   c) Updates $\boldsymbol{a}_j$ as

$$\boldsymbol{a}_j^{k+1} = \arg\min_{\boldsymbol{a}_j} L_\rho'^k = \left(\left(\hat{H}_j^k\right)^T \hat{H}_j^k + \rho\big(n(P_j) + n(S_j)\big) I_{2n}\right)^{-1} \boldsymbol{\kappa}_j^k, \tag{17}$$

   where $\boldsymbol{\kappa}_j^k = (\hat{H}_j^k)^T \hat{\mathbf{c}}_j^k + \sum_{v \in S_j} \mathbf{w}_{jv}^k - \sum_{v \in P_j} \mathbf{w}_{vj}^k + \rho(\sum_{v \in P_j} \boldsymbol{a}_v^{k+1} + \sum_{v \in S_j} \boldsymbol{a}_v^k)$.
   d) Updates all $\mathbf{w}_{vj}$ for $l \in P_j$ as

$$\mathbf{w}_{vj}^{k+1} = \mathbf{w}_{vj}^k - \rho\left(\boldsymbol{a}_v^{k+1} - \boldsymbol{a}_j^{k+1}\right).$$

   e) Sends $\boldsymbol{a}_j^{k+1}$ to all PDCs in $P_j \cup S_j$.
   f) Sends $\mathbf{w}_{vj}^{k+1}$ to $v \in P_j$.
   g) Receives $\boldsymbol{a}_v^{k+1}$ and $\mathbf{w}_{jv}^{k+1}$ from all $v \in S_j$.

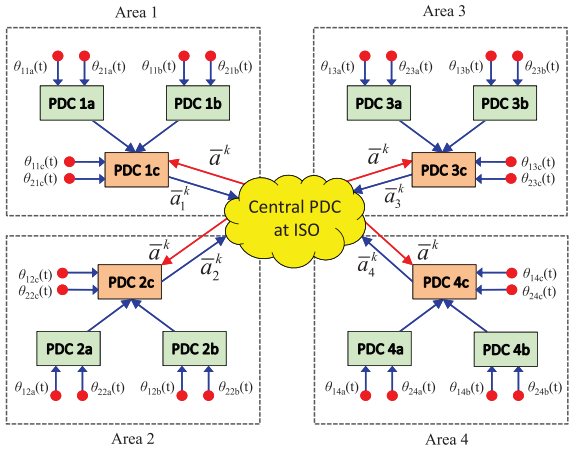


Fig. 4. Architecture 3 using H-ADMM for a network consisting of four areas.

successors. PDC $j$ also updates the dual variables $\mathbf{w}_{vj}$ for $v$ belonging to $P_j$. This algorithm also has a convergence rate of $\mathcal{O}(1/k)$, and can be realized using A-ADMM [22].

### *D. Architecture 3—Distributed Prony Using Hierarchical ADMM*

Let us consider again the Prony formulation in Architecture 1 where every area is assumed to contain only one aggregated PDC. However, if $p$, the number of PMUs is large, a better strategy will be to create multiple hierarchical layers of PDCs so that the computational load of the global estimation gets divided, as shown in Fig. 4. Let us divide the entire network into $r$ computational areas, where each area contains multiple layers of PDCs. Each PDC receives measurements from a subset of the total number of PMUs in that area. We assume these subsets to be disjoint.

**Algorithm 4** Distributed Prony Using the H-ADMM

1) Every PDC $j$ initializes $\boldsymbol{a}_j^0$, $\mathbf{z}^0$, and $\mathbf{w}_j^0$, $j = 1, \ldots, N$.
2) At each iteration $k$:
   a) PDC $j$ belonging to layer $\ell$ of Area $l$ constructs $\hat{H}_j^k$ and $\hat{\mathbf{c}}_j^k$ from (9).
   b) PDC $j$ receives the values of $\mathbf{z}_v^{k+1}$ from all PDCs $v$ belonging to $Q_j$.
   c) PDC $j$ computes $\boldsymbol{a}_j^{k+1}$ using (11).
   d) PDC $j$ calculates $\mathbf{z}_j^{k+1} = \boldsymbol{a}_j^{k+1} + \sum_{v \in Q_j} \mathbf{z}_v^{k+1}$.
   e) If PDC $j$ is not the leader PDC of Area $l$, it transmits $\mathbf{z}_j^{k+1}$ to the PDC $v$ belonging to $U_j$. If PDC $j$ is the leader PDC $l$, then it transmits $\mathbf{z}_l^{k+1} \triangleq \mathbf{z}_j^{k+1}$ to the central PDC.
   f) The central PDC receives $\mathbf{z}_l^{k+1}$ from the leader PDC $l$, $l = 1, \ldots, r$.
   g) The central PDC calculates $\mathbf{z}^{k+1} = (1/N) \sum_{l=1}^r \mathbf{z}_l^{k+1}$.
   h) The central PDC broadcasts $\mathbf{z}^{k+1}$ to the $r$ leader PDCs through an interarea communication network.
   i) The leader PDC $l$ broadcasts $\mathbf{z}^{k+1}$ to all PDCs in Area $l$ through an intraarea communication network.
   j) Every PDC $j$ in Area $l$ computes $\mathbf{w}_j^{k+1}$ using (13).

For each PDC $j$ in layer $\ell$ of Area $l$, let $Q_j$ denote the set of the PDCs in layer $\ell - 1$ from which it receives information, and let $U_j$ denote the set containing a single PDC in layer $\ell + 1$ to which it sends information. Also, without loss of generality, let us assume that the final layer in every area consists of exactly one PDC. This PDC is referred to as the leader PDC for any Area $l$, $l = 1, \ldots, r$. Every leader PDC is connected to the central PDC at the ISO through an interarea communication link. For example, in the system shown in Fig. 4, we have $r = 4$ areas with two layers of PDCs in each. PDCs 1b and 1c belong to layer 1 while PDC 1c is the leader PDC of Area 1. Similar notations have been used for the other areas. Using these definitions, distributed Prony using the Hierarchical ADMM (H-ADMM) is described in Algorithm 4.

Note that the solution of H-ADMM is equal to that of an equivalent S-ADMM problem with $N$ areas, each containing exactly one PDC. Table I compares the three proposed architectures in terms of their various properties.

## V. Case Studies

### *A. IEEE 68-Bus Model*

To verify the distributed Prony algorithms described in Section IV, we first consider the IEEE 68-bus system shown in Fig. 5. The system is divided into five areas, each with one local PDC and three PMUs as shown in Fig. 5. The simulated measurements are obtained using the power systems toolbox (PST) nonlinear dynamics simulation routine `s_simu` and the data file `data16m.m` [23]. The synchronous generators in this model are assumed to be sixth order for the

TABLE I
COMPARISON BETWEEN THREE DIFFERENT ARCHITECTURES

| Architecture | Resilience | Data Privacy | Convergence Rate | Sensitivity to Network Delay | Data Volume Per PDC |
|---|---|---|---|---|---|
| S-ADMM | low, single point failure | high | $\mathcal{O}(1/k)$ | low sensitivity to inter-area delay | high |
| D-ADMM | high, collaborative | low | $\mathcal{O}(1/k)$ | high sensitivity to inter-area delay | high |
| H-ADMM | low, single point failure | low | $\mathcal{O}(1/k)$ | high sensitivity to intra-area delay | low |

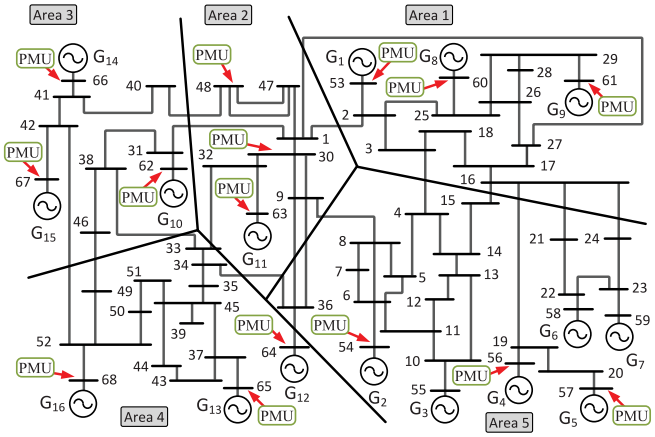


Fig. 5. IEEE 68-bus model.

sake of practicality. A three-phase fault is considered occurring at the line connecting buses 1 and 2. The fault starts at $t = 0.1$ s, clears at bus 1 at $t = 0.15$ s and at bus 2 at $t = 0.20$ s. The measurements are downsampled and the sampling period $T$ is increased up to up to 0.2 s. Our objective is to estimate the post-fault interarea oscillation modes of the system. Since there are 16 generators, our proposed algorithms should ideally solve a 96th order polynomial. However, several of these 96 eigenvalues have negligible residues in (2), as a result of which, the practical order can be chosen to be a much smaller number. In fact, in our simulations we show that choosing $2n = 40$ yields a satisfactory estimates of the interarea modes. The initial ten samples (2 s) of the measurements are gathered before starting the optimization iterations. We set $\rho = 10^{-9}$.

*1) Results of Distributed Prony Using S-ADMM:* We first deploy the Prony algorithm using S-ADMM. Fig. 6(a) shows how the estimates of $\sigma$ and $\Omega$ per iteration converges to their global values for four selected slow modes after 50 iterations. The dashed lines show the actual values of $\sigma$ and $\Omega$ for these four modes obtained from PST. Fig. 8 also compares the errors per iteration of S-ADMM with the other three algorithms. As the curves show, the error in all algorithms converge to zero asymptotically.

*2) Results of Distributed Prony Using A-ADMM:* For the asynchronous case, the active PDCs in each iteration are chosen randomly with equal probability of 0.5 for a PDC to be either active or dormant. Fig. 6(b) shows the estimates of $\sigma$ and $\Omega$ per iteration for each of the four selected modes. Compared to S-ADMM, the convergence of this method is slower due to the stochastic nature of the algorithm.

*3) Results of Distributed Prony Using D-ADMM:* For the distributed case, we consider the two communication graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ shown in Fig. 7. Fig. 6(c) and (d) shows the mode estimation per iteration for the four selected eigenvalues using $\mathcal{G}_1$ and $\mathcal{G}_2$, respectively. These figures show that the estimates of $\sigma$ and $\Omega$ per iteration converges to their actual values asymptotically using both $\mathcal{G}_1$ and $\mathcal{G}_2$. However, $\mathcal{G}_2$ has more number of communication links compared to $\mathcal{G}_1$, which makes it a less-favorable choice.

*4) Results of Distributed Prony Using H-ADMM:* We now consider each of the three PMUs in every area to be equipped with its own PDC. Two of these PMU-PDC pairs are considered to be in layer 1, while the third pair is considered to be in layer 2. Fig. 6(e) shows the four selected eigenvalues converge to their global values. Fig. 8 also shows the estimation error per iteration using this algorithm.

It should be noted that Fig. 8 only compares the estimation results per iteration for the proposed algorithms, and not the time needed to execute an iteration. It should be noted that every iteration of H-ADMM involves sequential communication between the hierarchies of PDCs, and hence, the actual time for completing one iteration of H-ADMM will be larger than that of S-ADMM. Similarly, every iteration of D-ADMM may be slower than a corresponding iteration of S-ADMM since the former involves sequential communication, while the latter involves parallel communication between PDCs.

Table II shows the estimates of the four interarea eigenvalues obtained from the four algorithms, and compares them with their actual values and the results of the centralized Prony. It can be seen that all these algorithms yield accurate estimates of the slow eigenvalues with relative error less than 1%.

The drawbacks of these architectures compared to the centralized case are obvious. S-ADMM, unlike centralized Prony, needs both uplink and downlink communications. D-ADMM suffers from loss of data privacy. H-ADMM involves higher communication delays at the cost of lesser computations.

### B. IEEE 145-Bus Model

We next consider the IEEE 145-bus model as a larger case study [24]. We assume the system to be divided into eight areas. Each area contains one local PDC and ten PMUs. The identity of PMU buses are listed in Table III. The simulated measurements are obtained using PST nonlinear dynamics simulation routine `s_simu` and the data file `data50m.m` [23]. A three-phase fault is applied on the line connecting buses 6 and 7. The fault starts at $t = 0.1$ s, clears at bus 6 at $t = 0.15$ s and at bus 7 at $t = 0.20$ s. The measurements are downsampled using $T = 0.1$ s. The initial ten samples (1 s) of the measurements are gathered before starting the iterations. We set $2n = 120$ and $\rho = 10^{-5}$. Fig. 9 shows how the estimates of $\sigma$ and $\Omega$ per iteration (the solid lines) converge to the actual values (the dashed lines) for four selected slow modes of the system after 200 iterations.

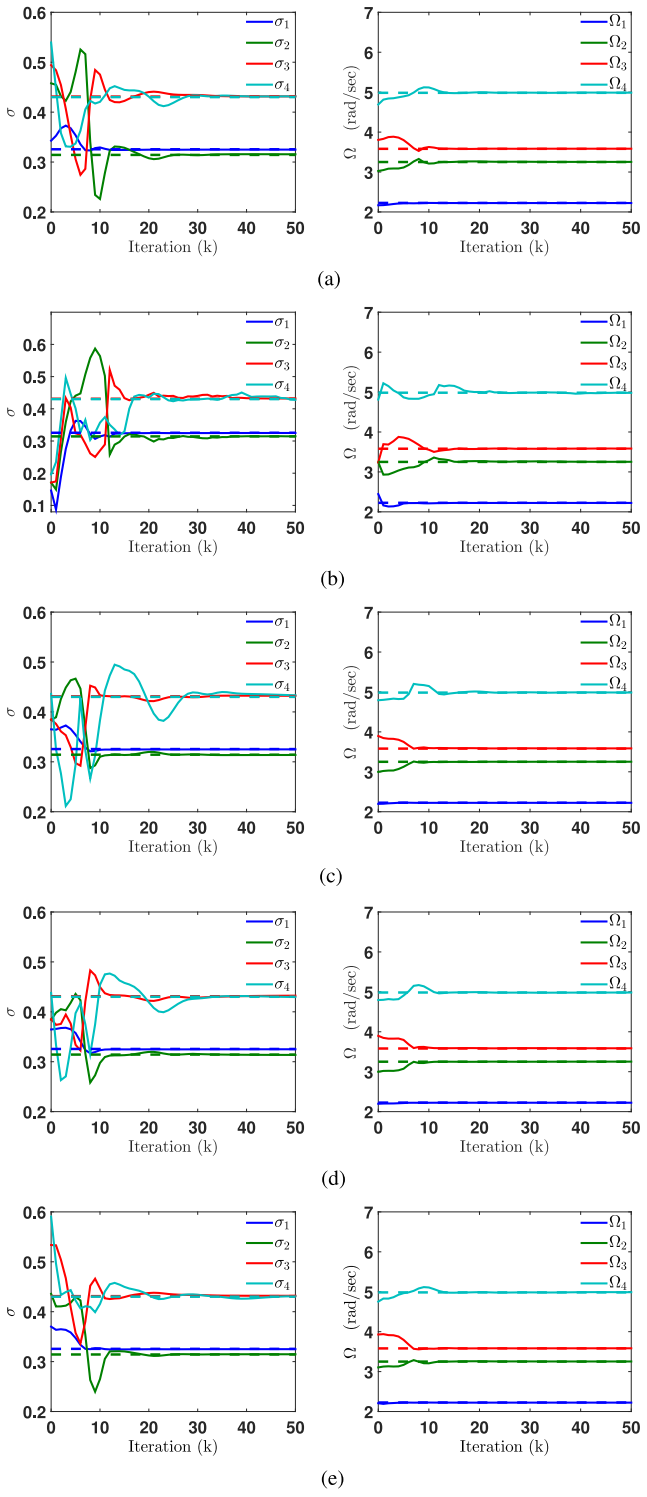


Fig. 6. Estimation of $\sigma$ and $\Omega$ per iteration $k$ (solid lines) versus their actual values obtained from PST (dashed lines). (a) S-ADMM algorithm. (The values are calculated at the ISO from $\tilde{\boldsymbol{a}}^k$ based on Step 2 of the Prony algorithm.) (b) A-ADMM. (c) and (d) D-ADMM with $\mathcal{G}_1$ and $\mathcal{G}_2$, respectively. (The values are calculated using $\boldsymbol{a}_1^k$.) (e) H-ADMM.

The average computation time per iteration per PDC in this case is 4.7 ms. For the same number of PMUs $p$, if we reduce the number of PMUs per PDC to 4, and hence, increase the number of PDCs up to 20, the average run-time per iteration reduces down to 1.9 ms. This clearly shows that the proposed

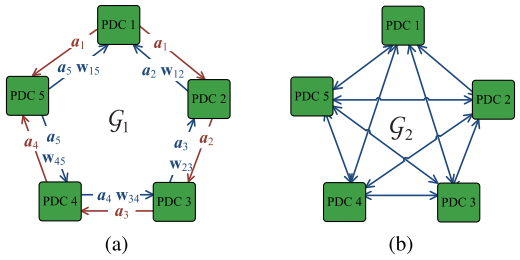


Fig. 7. Communication graphs used in Architecture 2. (a) D-ADMM ($\mathcal{G}_1$). (b) D-ADMM ($\mathcal{G}_2$).

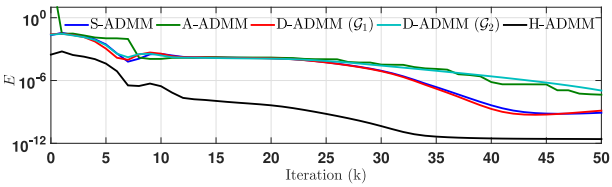


Fig. 8. Computation of the error term $E(k) = \sum_{j=1}^{N}(1/2)\|\hat{H}_j^k \boldsymbol{a}_j^k - \hat{\mathbf{c}}_j^k\|_2^2$ for the four proposed algorithms per iteration.

distributed algorithms are more beneficial in terms of computation time as the network becomes large since larger the network, easier it is to control the ratio of the number of PMUs and PDCs in each area.

## VI. Real-Time Simulations Using ExoGENI

Besides the conventional approach of estimating oscillation modes using offline software programs such as MATLAB, we also implement our distributed algorithms using C/C++ in a realistic U.S.-wide network testbed called ExoGENI [25]. This testbed is designed to support research and innovation in networking, operating systems, future Internet architectures, and networked data-intensive cloud computing. We have recently integrated our real-time digital simulator (RTDS)-based hardware-in-the-loop PMU testbed at NC State with ExoGENI, thereby forming a federated testbed called ExoGENI-WAMS. Details about this testbed can be found in [26]. An advantage of using ExoGENI is that we can run parallel applications with performance isolation. Each application can run in its own virtual infrastructure that consists of virtual machine (VM) and storage connected by virtual networking channels. Therefore, in our proposed architecture, the oscillation monitoring algorithms can run in perfect isolation. Even the PMU data resolutions between these application layers can be varied without any interference by parallel downsampling.

We use this federated testbed to serve as a platform for evaluating the end-to-end delays in implementing the S-ADMM algorithm using real-time data streaming from multiple PMUs to multiple PDCs that are realized using VMs or computers. ExoGENI allows users to create custom topologies using resources from multiple federated providers via a control and management software called the open resource control architecture to orchestrate the networked cloud resource provisioning. We implement the Architecture 1 in the flowchart shown in Fig. 10, where four VMs serve as local PDCs

TABLE II
ESTIMATED SLOW EIGENVALUES OF THE IEEE 68-BUS MODEL USING CENTRALIZED PRONY (STEPS 1 AND 2) AND THE DISTRIBUTED PRONY ALGORITHMS AT $k = 50$

| Actual ($-\sigma_i \pm j\Omega_i$) | Centralized Prony | Distributed Prony | | | | |
|---|---|---|---|---|---|---|
| | | S-ADMM | A-ADMM | D-ADMM ($\mathcal{G}_1$) | D-ADMM ($\mathcal{G}_2$) | H-ADMM |
| $-0.3256 \pm j2.2262$ | $-0.3250 \pm j2.2230$ | $-0.3249 \pm j2.2231$ | $-0.3249 \pm j2.2231$ | $-0.3250 \pm j2.2231$ | $-0.3250 \pm j2.2231$ | $-0.3249 \pm j2.2231$ |
| $-0.3143 \pm j3.2505$ | $-0.3146 \pm j3.2531$ | $-0.3155 \pm j3.2535$ | $-0.3147 \pm j3.2533$ | $-0.3137 \pm j3.2525$ | $-0.3136 \pm j3.2525$ | $-0.3147 \pm j3.2533$ |
| $-0.4312 \pm j3.5809$ | $-0.4318 \pm j3.5849$ | $-0.4316 \pm j3.5838$ | $-0.4321 \pm j3.5849$ | $-0.4323 \pm j3.5865$ | $-0.4323 \pm j3.5864$ | $-0.4319 \pm j3.5838$ |
| $-0.4301 \pm j4.9836$ | $-0.4308 \pm j4.9865$ | $-0.4315 \pm j4.9871$ | $-0.4280 \pm j4.9801$ | $-0.4333 \pm j4.9857$ | $-0.4296 \pm j4.9834$ | $-0.4310 \pm j4.9883$ |

TABLE III
COMPUTATIONAL AREA PARTITIONING OF THE IEEE 145-BUS MODEL. THE BUSES WRITTEN IN BOLDFACE ARE EQUIPPED WITH PMUS (10 PMUS PER AREA)

| Area | Buses |
|---|---|
| 1 | **1, 2, 33, 37, 49, 50, 93, 99, 102, 110**, 3, 4, 5, 34, 35, 36, 38, 39, 40, 41, 42, 43, 44, 45, 47, 48, 53, 55, 84, 87, 88, 113, 114 |
| 2 | **6, 7, 8, 57, 63, 66, 67, 101, 104, 111**, 9, 10, 11, 32, 51, 52, 56, 69 |
| 3 | **13, 14, 20, 58, 59, 72, 98, 100, 103, 112**,12, 15, 16, 17, 18, 19, 21, 54, 70, 71 |
| 4 | **115, 116, 117, 118, 140, 141, 142, 143, 144, 145**, 46, 85 |
| 5 | **130, 131, 132, 133, 134, 135, 136, 137, 138, 139** |
| 6 | **61, 82, 95, 119, 120, 121, 122, 126, 128, 129**, 62, 86, 123, 127 |
| 7 | **60, 64, 65, 79, 80, 90, 94, 97, 107, 124**, 68, 92, 125 |
| 8 | **22, 77, 81, 89, 91, 96, 105, 106, 108, 109**, 23, 24, 25, 26, 27, 28, 29, 30, 31, 73, 74, 75, 76, 78, 83 |

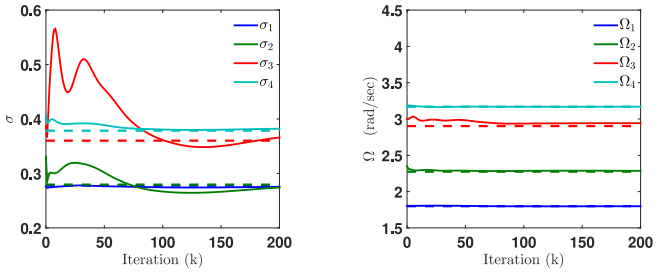


Fig. 9. Estimation of $\sigma$ and $\Omega$ using S-ADMM for the 145-bus model.

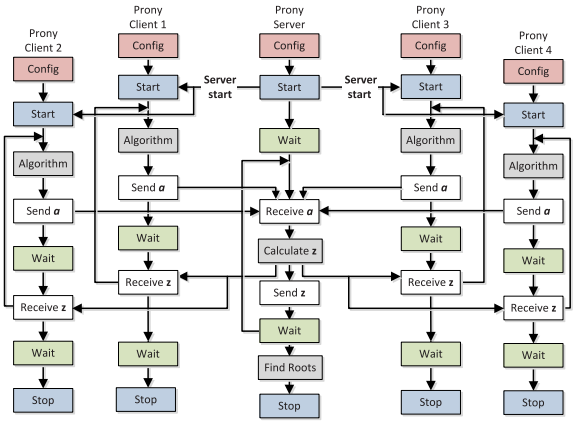


Fig. 10. Execution flowchart of the distributed Prony using S-ADMM.

(clients) and the fifth VM serves as the central server. The IEEE 39-bus power system model introduced in [17] is used here. The clients accept local PMU data coming from the RTDS running this model, execute (11), and send the estimates to the server. The server averages the estimates and transmits the average back to the four clients to proceed to the next step. The component-wise end-to-end delays for every iteration are shown in Table IV. It can be seen that although S-ADMM has a significantly large delay $T_3$, its end-to-end time is only 22% of that for the centralized Prony. This is because both $T_1$ and $T_2$ for the distributed case are much smaller than their centralized counterparts.

TABLE IV
$T_1$: MEASUREMENT EXCHANGE STREAMING DELAY, $T_2$: COMPUTATION DELAY, AND $T_3$: AVERAGING PLUS PARAMETER EXCHANGING DELAY (ALL TIME DELAYS ARE MEASURED FOR A SINGLE ITERATION, UNITS ARE IN MILLISECONDS)

| Algorithm | $T_1$(ms) | $T_2$(ms) | $T_3$(ms) | Total Time |
|---|---|---|---|---|
| Centralized Prony | 42.8 | 14.6 | N/A | 57.4 |
| Distributed Prony | 0.1 | 0.0932 | 12.7081 | 12.9 |

## VII. CONCLUSION

In this paper, we presented four cyber-physical estimation algorithms for wide-area oscillation monitoring using synchrophasors. Our algorithms demonstrate how multitudes of geographically dispersed PMUs and PDCs can communicate with each other, and how the various binding factors in the network protocols can pose bottlenecks for their communication. The results, thereby provide valuable insights and guidance in deploying future PMU and PDC infrastructures, not only for power systems but for any generic cyber-physical sensor network where monitoring and control decisions need to be made under critical time-constraints. Our future work will be to evaluate the reliability of the proposed architectures under different cyber-attack scenarios.

## REFERENCES

[1] A. G. Phadke and J. S. Thorp, *Synchronized Phasor Measurements and Their Applications*. New York, NY, USA: Springer, 2008.
[2] A. Bose, "Smart transmission grid applications and their supporting infrastructure," *IEEE Trans. Smart Grid*, vol. 1, no. 1, pp. 11–19, Jun. 2010.
[3] P. T. Myrda, J. Taft, and P. Donner, "Recommended approach to a NASPInet architecture," in *Proc. 45th Hawaii Int. Conf. Syst. Sci. (HICSS)*, Maui, HI, USA, 2012, pp. 2072–2081.
[4] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1989.
[5] C. W. Taylor, D. C. Erickson, K. E. Martin, R. W. Wilson, and V. Venkatasubramanian, "WACS—Wide-area stability and voltage control system: R & D and online demonstration," *Proc. IEEE*, vol. 93, no. 5, pp. 892–906, May 2005.
[6] R. Hasan, R. Bobba, and H. Khurana, "Analyzing NASPInet data flows," in *Proc. IEEE Power Syst. Conf. Expo. (PSCE)*, Seattle, WA, USA, 2009, pp. 1–6.
[7] J. J. Sanchez-Gasca and J. H. Chow, "Performance comparison of three identification methods for the analysis of electromechanical oscillations," *IEEE Trans. Power Syst.*, vol. 14, no. 3, pp. 995–1002, Aug. 1999.
[8] J. F. Hauer, C. J. Demeure, and L. L. Scharf, "Initial results in Prony analysis of power system response signals," *IEEE Trans. Power Syst.*, vol. 5, no. 1, pp. 80–89, Feb. 1990.

[9] N. Zhou, D. J. Trudnowski, J. W. Pierre, and W. A. Mittelstadt, "Electromechanical mode online estimation using regularized robust RLS methods," *IEEE Trans. Power Syst.*, vol. 23, no. 4, pp. 1670–1680, Nov. 2008.
[10] A. R. Messina and V. Vittal, "Nonlinear, non-stationary analysis of inter-area oscillations via Hilbert spectral analysis," *IEEE Trans. Power Syst.*, vol. 21, no. 3, pp. 1234–1241, Aug. 2006.
[11] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2011.
[12] S. Kar and G. Hug, "Distributed robust economic dispatch in power systems: A consensus + innovations approach," in *Proc. IEEE Power Energy Soc. Gen. Meeting*, San Diego, CA, USA, 2012, pp. 1–8.
[13] Z. Zhang and M.-Y. Chow, "Convergence analysis of the incremental cost consensus algorithm under different communication network topologies in a smart grid," *IEEE Trans. Power Syst.*, vol. 27, no. 4, pp. 1761–1768, Nov. 2012.
[14] E. Dall'Anese, H. Zhu, and G. B. Giannakis, "Distributed optimal power flow for smart microgrids," *IEEE Trans. Smart Grid*, vol. 4, no. 3, pp. 1464–1475, Sep. 2013.
[15] A. D. Dominguez-Garcia and C. N. Hadjicostis, "Coordination of distributed energy resources for provision of ancillary services: Architectures and algorithms," in *Encyclopedia of Systems and Control*, J. Baillieul and T. Samad, Eds. Berlin, Germany: Springer-Verlag, 2014.
[16] T. Erseghe, "Distributed optimal power flow using ADMM," *IEEE Trans. Power Syst.*, vol. 29, no. 5, pp. 2370–2380, Sep. 2014.
[17] S. Nabavi and A. Chakrabortty, "A real-time distributed Prony-based algorithm for modal estimation of power system oscillations," in *Proc. Amer. Control Conf. (ACC)*, Portland, OR, USA, 2014, pp. 729–734.
[18] S. Nabavi and A. Chakrabortty, "Distributed estimation of inter-area oscillation modes in large power systems using alternating direction multiplier method," in *Proc. IEEE Power Energy Soc. Gen. Meeting*, National Harbor, MD, USA, 2014, pp. 1–5.
[19] A. A. Fouad and P. M. Anderson, *Power System Control and Stability*. Piscataway, NJ, USA: IEEE Press, 2003.
[20] E. Wei and A. Ozdaglar. (2013). *On the $O(1/k)$ Convergence of Asynchronous Distributed Alternating Direction Method of Multipliers*. [Online]. Available: http://arxiv.org/abs/1307.8254
[21] R. Zhang and J. T. Kwok, "Asynchronous distributed ADMM for consensus optimization," in *Proc. 31st Int. Conf. Mach. Learn.*, Beijing, China, 2014, pp. 1701–1709.
[22] E. Wei and A. Ozdaglar, "Distributed alternating direction method of multipliers," in *Proc. IEEE Conf. Decis. Control*, Maui, HI, USA, 2012, pp. 5445–5450.
[23] J. H. Chow and K. W. Cheung, "A toolbox for power system dynamics and control engineering education and research," *IEEE Trans. Power Syst.*, vol. 7, no. 4, pp. 1559–1564, Nov. 1992.
[24] V. Vittal, "Transient stability test systems for direct stability methods," *IEEE Trans. Power Syst.*, vol. 7, no. 1, pp. 37–43, Feb. 1992.
[25] I. Baldine *et al.*, "ExoGENI: A multi-domain infrastructure-as-a-service testbed." in *Testbeds and Research Infrastructure. Development of Networks and Communities*. Berlin, Germany: Springer-Verlag, 2012, pp. 97–113.
[26] J. Zhang, A. Chakrabortty, and X. Yufeng, "Distributed implementation of wide-area monitoring algorithms for power systems using a US-wide ExoGENI-WAMS testbed," in *Proc. 44th Annu. IEEE/IFIP Int. Conf. Depend. Syst. Netw. (DSN)*, Atlanta, GA, USA, Jun. 2014, pp. 762–767.

**Seyedbehzad Nabavi** (S'12) received the B.S. degree from the Amirkabir University of Technology, Tehran, Iran, in 2009, and the M.S. degree from North Carolina State University, Raleigh, NC, USA, in 2011, where he is currently pursuing the Ph.D. degree, all in electrical engineering.

His current research interests include wide-area monitoring of power systems.

**Jianhua Zhang** (S'12) received the B.E. degree from Jimei University, Xiamen, China, in 2002, and the M.E. degrees from Xiamen University, Xiamen, and the New Mexico Institute of Mining and Technology, Socorro, NM, USA, in 2005 and 2010, respectively, all in electrical engineering. She is currently pursuing the Ph.D. degree in electrical engineering with North Carolina State University, Raleigh, NC, USA.

Her current research interests include wide-area monitoring, security, and control of power systems.

**Aranya Chakrabortty** (S'02–M'06–SM'15) received the Ph.D. degree in electrical engineering from Rensselaer Polytechnic Institute, Troy, NY, USA, in 2008.

Since 2010, he has been an Assistant Professor in the Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, NC, USA, where he is also affiliated with the FREEDM Systems Center. His current research interests include all branches of control theory with applications to power systems, especially in wide-area monitoring and control using synchrophasors.

Dr. Chakrabortty was the recipient of the National Science Foundation CAREER Award in 2011.