illiad: InteLLigent Invariant and Anomaly Detection in Cyber-Physical Systems

NIKHIL MURALIDHAR*, Discovery Analytics Center, Virginia Tech
CHEN WANG*, Department of Electrical and Computer Engineering, Virginia Tech
NATHAN SELF, Discovery Analytics Center, Virginia Tech
MARJAN MOMTAZPOUR, Microsoft Inc.
KIYOSHI NAKAYAMA and RATNESH SHARMA, NEC Laboratories America, Inc.
NAREN RAMAKRISHNAN, Discovery Analytics Center, Virginia Tech

Cyber-physical systems (CPSs) are today ubiquitous in urban environments. Such systems now serve as the backbone to numerous critical infrastructure applications, from smart grids to IoT installations. Scalable and seamless operation of such CPSs requires sophisticated tools for monitoring the time series progression of the system, dynamically tracking relationships, and issuing alerts about anomalies to operators. We present an online monitoring system (*illiad*) that models the state of the CPS as a function of its relationships between constituent components, using a combination of model-based and data-driven strategies. In addition to accurate inference for state estimation and anomaly tracking, *illiad* also exploits the underlying network structure of the CPS (wired or wireless) for state estimation purposes. We demonstrate the application of *illiad* to two diverse settings: a wireless sensor motes application and an IEEE 33-bus microgrid.

CCS Concepts: • **Mathematics of computing** → *Time series analysis*; • **Information systems** → **Sensor networks**; **Data mining**; • **Computing methodologies** → **Anomaly detection**; • **Hardware** → *Power and energy*; *Smart grid*;

Additional Key Words and Phrases: Urban computing, urban informatics, IoT, big-data, state-estimation

ACM Reference format:
Nikhil Muralidhar, Chen Wang, Nathan Self, Marjan Momtazpour, Kiyoshi Nakayama, Ratnesh Sharma, and Naren Ramakrishnan. 2018. illiad: InteLLigent Invariant and Anomaly Detection in Cyber Physical Systems. *ACM Trans. Intell. Syst. Technol.* 9, 3, Article 35 (January 2018), 20 pages.
https://doi.org/10.1145/3066167

1 INTRODUCTION

It has been projected that by the year 2030, cities will grow by 590,000 square miles and add an additional 1.47 billion people, so six of every ten people will live in a city. One of the most

This research has been supported in part by U.S. National Science Foundation, Grant No. DGE-1545362, and the Institute for Critical Technology and Applied Science, Virginia Tech. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.
Authors' addresses: N. Muralidhar, N. Self, and M. Momtazpour, 900 North Glebe Road, Arlington, VA 22203; emails: {nik90, nwself}@vt.edu, marjan@cs.vt.edu; C. Wang, The Bradley Department of Electrical Engineering, Perry St, Blacksburg, VA 24060; email: chenwang@vt.edu; K. Nakayama and R. Sharma, NEC Laboratories America, Inc., 10080 North Wolfe Road, Suite SW3-350, Cupertino, CA 95014; emails: {knakayama, ratnesh}@nec-labs.com; N. Ramakrishnan, VTRC-Arlington, Room 5-026, 900 North Glebe Road, Arlington, VA 22203; email: naren@cs.vt.edu.

https://doi.org/10.1145/3066167

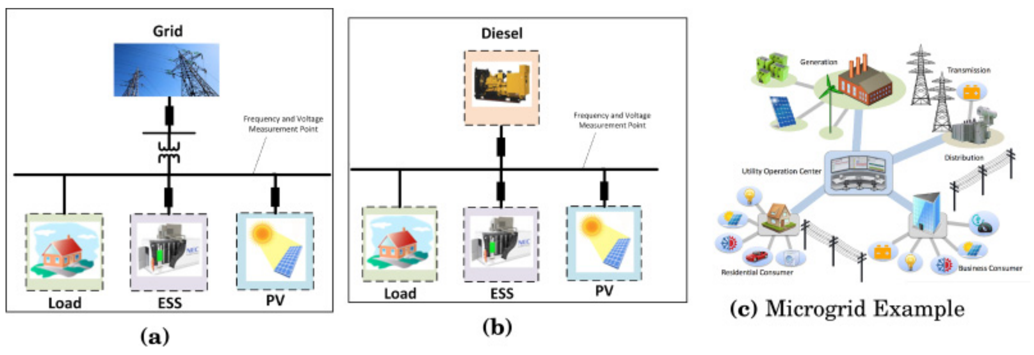


Fig. 1. Urban microgrid examples and design and deployment scenarios. **(a)** Represents an AC grid connected to main grid through a point of common coupling (PCC). **(b)** Represents a Diesel generator along with other Distributed Generation sources without the main grid.

consequential changes with this global influx of citizens will be the stress placed on cyber-physical systems across the urban landscape, from smart grids to massively distributed IoT installations to support net-zero energy objectives. At the same time, as cities become seen as urban-scale cyber-physical systems, a vast amount of data about system management and operation is continuously being harvested and analyzed through sensor networks. Scalable and seamless operation of such cyber-infrastructure requires sophisticated tools for monitoring the progression of the system, dynamically tracking relationships, and issuing anomaly alerts to operators.

Leveraging our prior work (Momtazpour et al. 2015), we develop a system dynamics approach (*illiad*) to invariant and anomaly detection in cyber-physical systems. Our key contributions are:

— A state estimation and anomaly detection algorithm (KASE) that combines model-based (Recursive Bayesian Filtering) and data-driven (Autoregression with Exogenous Inputs and Exploratory Factor Analysis) approaches. The integration of model-based and data-driven strategies leverages the selective superiorities of both into a comprehensive system.
— An approach to incorporate the underlying network structure of the cyber-physical system (wired or wireless) into the state estimation process. We demonstrate how this idea significantly improves the computational complexity of inference and renders the approach tractable to large urban settings.
— A visual dashboard application for real-time anomaly detection and alerting in urban-scale cyber-physical systems. We demonstrate the application of *illiad* to two diverse settings: a wireless sensor motes application and an IEEE 33-bus microgrid.

2 BACKGROUND AND RELATED WORK

Availing energy from renewable sources to meet increased energy demands due to increasing electrification in urban environments is a key challenge we are faced with today as alluded to by Zheng et al. (2014). An integration of microgrids into the existing power grid has been widely acknowledged as a potential solution. Resilient operation of power and microgrid systems requires constant monitoring of data to extract or predict anomalies and to support rapid system recovery. Figure 1 represents typical urban microgrids and deployment scenarios. The microgrid consists of several distributed generation units (DGs), such as a photovoltaic (PV) array and diesel generator, as well as energy storage systems and loads. These components are connected together using power lines, transformers, and feeders. Microgrids can operate in both islanded and grid-tied modes. In the islanded mode, the microgrid is as shown in Figure 1(b).

Energy systems (e.g., PV, battery, load, diesel generator) are typically installed and connected to the microgrid controller via TCP/IP with systems that feature Supervisory Control and Data Acquisition (SCADA), exploiting open communication standards, such as OLE for Process Control (OPC) and Modbus RTU/TCP, which is a master-slave protocol for use with its programmable logic controllers (PLCs). Measurement data and its collected features vary depending on the type of energy device and its communications protocol, such as the format of XML. The measurement data contains active and reactive output power values of each device, voltage and frequency values as microgrid data, and state of charge (SOC) for batteries for every second.

Microgrids, including those studied here, are typically based on traditional power distribution system models, while also incorporating relatively modern urban components, such as electric vehicles (EVs) and energy storage (ES) devices along with renewable power generation components, such as wind turbines and PV cells (Chowdhury and Crossley 2009). However, such new additions make the operation of the distribution system more complex (Tsikalakis and Hatziargyriou 2011). One factor, for example, would be the uncertain and intermittent power output of the renewable distributed generation components, specifically PV panels and wind turbines. Such intermittent power output makes system state estimation a challenging task.

State estimation of the power system is typically accomplished by learning invariant relationships between system components, and such invariants are then used in an energy management system (EMS) to construct a real-time network model (Monticelli 2000). In Cobelo et al. (2007), the authors proposed a method aimed at providing the distribution management system controller with real-time information from the microgrid to increase the penetration of the renewable distributed generation. Rana and Li (2015) have proposed a Kalman filter-based microgrid state estimation method using an IoT network to acquire information about the distributed generation grid. Wang et al. (2014) also proposed a linear filtering-based state estimation method that mainly focused on small signal models. Hu et al. (2011) developed a probabilistic model to conduct real-time linear state estimation through belief propagation. Instead of focusing on a single microgrid, Korres et al. (2011) proposed a multi-microgrid state estimator with limited real-time measurement and investigated the impact of distributed generation in both grid-connected and islanded scenarios. However, most existing research has focused on accuracy of estimation rather than anomaly detection using invariant relationships between components. Even fewer approaches exist that successfully combine model-based and data-driven approaches with the goal of anomaly detection through state estimation. Anomaly detection in microgrids using sensor location and connection information can be considered a spatio-temporal anomaly detection problem, and some relevant work has been carried out by Chawla et al. (2012) in the transportation domain. In Zheng et al. (2015), the authors propose a spatio-temporal anomaly detection system for detecting collective anomalies by leveraging multi-domain datastreams.

3 PROBLEM FORMULATION

As described earlier, our work builds upon the preliminary framework described in Momtazpour et al. (2015), and we begin by introducing this approach in context.

We are given a set of n timeseries $\mathcal{D} = x_1(t) \ldots x_n(t)$ in a single or across multiple cyber-physical systems, each timeseries $x_i(t)$ is modeled as a vector. The values of the vector for a time window $t_k \ldots t_{k+w}$ are represented as

$$\mathcal{X}_i^{k:k+w} = [x_i(t_k), x_i(t_{k+1}), \ldots x_i(t_{k+w})]^T. \tag{1}$$

Each timeseries $x_i(t)$ is represented by a random variable X_i and is assumed to be drawn from a distribution represented by X_i. Every CPS has many direct and latent interactions amongst the components and having sufficient insight into these interactions and relationships is crucial to effectively manage the system.

Definition 3.1 (Approximate Dependency). At time step t_m, time series $x_j(t) \in \mathcal{D}$ approximately depends on $x_i(t) \in \mathcal{D}$ if and only if there exists a function $f : \mathbb{R} \rightarrow \mathbb{R}$ that for appropriately small $\epsilon > 0$:

$$\hat{x}_j(t_m) = f\left(\mathcal{X}_j^{1:m-1}, \mathcal{X}_i^{1:m}\right) \tag{2}$$

and

$$|x_j(t_m) - \hat{x}_j(t_m)| < \epsilon. \tag{3}$$

This dependency is depicted by $x_j(t) \xrightarrow{\epsilon} x_i(t)|_{t_m}$.

Definition 3.2 (System Invariants). Two timeseries, $x_j(t) \in \mathcal{D}$ and $x_i(t) \in \mathcal{D}$, are system-invariant up to time T within range of ϵ if and only if at least one of the following is satisfied:

$$\exists f : \mathbb{R} \rightarrow \mathbb{R} \quad \textit{and} \quad \forall \quad t \mid 0 \leq t \leq T : x_j(t) \xrightarrow{\epsilon} x_i(t)|_{0 \leq t \leq T}$$

or

$$\exists f : \mathbb{R} \rightarrow \mathbb{R} \quad \textit{and} \quad \forall \quad t \mid 0 \leq t \leq T : x_i(t) \xrightarrow{\epsilon} x_j(t)|_{0 \leq t \leq T}.$$

We denote an invariant time series by $x_i(t) \overset{\epsilon}{\rightleftharpoons} x_j(t)$.

Based on the nature of the system, dependencies between time series can be linear or nonlinear and this is modeled by the function f. In complex cyber-physical systems, when we have a large number of time series, it is appropriate to represent the invariants in the form of a graph.

Definition 3.3 (Invariant Graph). Graph $G = (V, E)$ with the set of vertices $V = \{v_1, \ldots, v_n\}$ and the set of edges $E = \{e_1, \ldots, e_n\}$ is called an invariant graph of a system with observed timeseries $\mathcal{D} = \{x_1, \ldots x_n\}$, where $e = (v_i, v_j) \in E$ if and only if $x_i(t) \overset{\epsilon}{\rightleftharpoons} x_j(t)$.

From Definition 3.3, we can surmise that vertex v_i is equivalent to a timeseries $x_i(t)$. System invariants and invariant graphs represent features and system-function under normal conditions. In the presence of anomalies, when the function of the system deviates from the norm, these dependencies might disappear. While two timeseries $x_i(t)$ and $x_j(t)$ might be invariant under normal conditions, the invariant relationship might disappear in the case of an anomaly.

Definition 3.4 Broken Invariants. We say that system invariant $x_i(t) \overset{\epsilon}{\rightleftharpoons} x_j(t)$ is broken at time $T = t_m$ if and only if timeseries $x_i(t)$ and $x_j(t)$ satisfy the following conditions:

$$\exists f : \mathbb{R} \rightarrow \mathbb{R}$$

and

$$\forall \quad t \mid 0 \leq t < T = t_m :$$

$$\left(x_j(t) \xrightarrow{\epsilon} x_i(t)|_{t<T} \quad \wedge \quad |x_j(t_m) - f(\mathcal{X}_j^{1:m-1}, \mathcal{X}_i^{1:m})| \geq \epsilon\right)$$

or

$$\left(x_i(t) \xrightarrow{\epsilon} x_j(t)|_{t<T} \quad \wedge \quad |x_i(t_m) - f(\mathcal{X}_i^{1:m-1}, \mathcal{X}_j^{1:m})| \geq \epsilon\right).$$

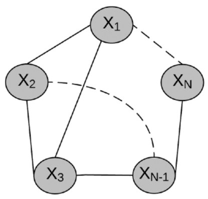


Fig. 2. Original System Topology.

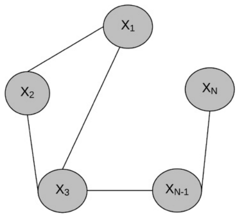


Fig. 3. ARX Invariant Graph.

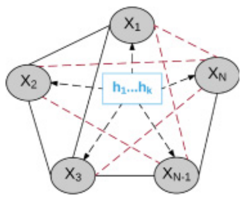


Fig. 4. Original Method (ARX + ARXL) Invariant Graph.

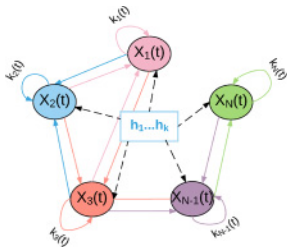


Fig. 5. KASE Method Invariant Graph.

In a CPS, the existence of certain unobserved factors has an effect on overall system behavior. But modeling these factors and their effects on all the electro-mechanical devices in a CPS is a non-trivial task. Characterizing these latent effects can aid in state estimation.

Figure 2 shows the network topology of an example CPS with N components $X_1, \ldots, X_N$. Here, the solid lines indicate direct relationships between sensors through physical connections while the dashed lines indicate indirect relationships between sensors. We see five direct and two indirect relationships depicted here. Figures 3 and 4 represent invariant graphs that can be inferred using current state-of-the-art methods (ARX and ARX+ARXL; introduced in Momtazpour et al. (2015)). ARX is a classical autoregression method (with exogenous inputs) and thus captures only direct relationships, as shown in Figure 3. ARX+ARXL, as shown in Figure 4 does learn indirect relationships but these relationships are linear and rather simplistic in nature. Our proposed approach (KASE) aims to identify more complex hidden relationships through hidden factors labeled $h_1, \ldots h_k$ and Kalman state estimates in Figure 5. These support the modeling of the system at higher orders.

3.1 State Estimation with Kalman Filters

Anomaly detection in sensor networks can be addressed using a dynamic state estimation technique (Nishiya et al. 1982), wherein at any time point t, we wish to estimate the state of each component in the system. As a function of these state estimates and corresponding actual state measurements, we can then infer whether the system is stable or if there is an anomaly in one or more components in the system. One of the most robust and widely used methods for state estimation is the Kalman filter (Kalman 1960), which is a special case of recursive Bayesian estimation wherein the data is considered to obey the Gaussian distribution (Chen 2003). In this article, we consider the effects of state estimation in CPS using the linear Kalman filter. We model the Kalman filter as described by Welch and Bishop (2001). It attempts to estimate the state $x \in \mathbb{R}^n$ at time step k of a discrete-time controlled process with a measurement $z \in \mathbb{R}^n$, The Kalman filtering process assumes that the matrices A,H,Q, and R, which represent the process transition matrix, the measurement transition matrix, the process covariance matrix, and measurement covariance matrix, respectively, and the initial mean and covariance of the data are known at the outset. In our work, we estimate A, Q, and R using expectation maximization (Borman 2004). The matrix H is the identity matrix. The initial state mean and covariance (μ_0 and Σ_0) are estimated from historical data.

3.2 Leveraging the Neighborhood Assumption

Section 3 outlines the basic problem framework and provides definitions about invariants, and anomalies. Although the original (ARX + ARXL) method attempts to find all direct and indirect dependencies in addition to learning complex latent variables, we hypothesize that considering the network topology of the CPS to learn only a set of direct relationships for each component yields equal or stronger state estimation and anomaly detection capabilities. We term this approach SAIL for *Structure Aware Invariant Learning*.

Let us first consider the case when the topology of the CPS is known, for example, consider the sample CPS depicted in Figure 2. Here, the system consists of N components $X_1, \ldots, X_N$. If we consider X_1, then it has two direct connections, one to X_3 and the other to X_2 along with an indirect relationship with X_N. Our hypothesis in the context of X_1 claims that the state of $X_1(t)$ can be estimated simply using only the values $X_1^{1:t-1}$, $X_3^{1:t}$, $X_2^{1:t}$ and the Kalman filter *a posteriori* state estimate $k_1(t)$. Hence, set $\{X_2, X_3\}$ forms the SAIL neighborhood of X_1 denoted S_n^1. In the context of wireless sensor networks that lack a connected network topological structure, we exploit the physical locations of sensors to obtain a proximal set of nearest neighbors for each component in the wireless network of CPS components.

The SAIL approach requires us to augment our previously stated definition 3.1 as follows:

Definition 3.5 (Structure Aware Approximate Dependency). At time step t_m, time series $x_j(t) \in \mathcal{D}$ with SAIL neighborhood S_n^j approximately depends on $x_i(t) \in \mathcal{D}$ if and only if, $x_i(t) \in S_n^j$ and there exists a function $f : \mathbb{R} \to \mathbb{R}$ that for appropriately small $\epsilon > 0$:

$$\hat{x}_j(t_m) = f\left(\mathcal{X}_j^{1:m-1}, \mathcal{X}_i^{1:m}\right) \tag{4}$$

and

$$|x_j(t_m) - \hat{x}_j(t_m)| < \epsilon \tag{5}$$

This dependency is depicted by $x_j(t) \xrightarrow{\epsilon} x_i(t)|_{t_m}$

The aforementioned structure informed invariant learning methods help scale the invariant learning procedure to large systems as they reduce the quadratic complexity of the algorithm. This is because, although the complexity of the algorithm technically is still quadratic in the worst case, the complexity of the newly proposed KASE (Kalman Autoregressive State Estimation with Latent Factors and Exogenous Inputs) invariant learning algorithm is a function of the average cardinality of the SAIL neighborhood for all the components. Since it is highly unlikely that all or even most of the components of a CPS will be directly connected, the SAIL neighborhood of each component will be relatively sparse hence achieving the scalability and speeding up the algorithm relative to benchmarks that do not exploit the neighborhood assumption.

We now outline the proposed KASE approach and explain the algorithm in detail in Section 4. In Figure 5, we observe different colored nodes and edges. If we consider one node, say $X_2(t)$, then we see that the SAIL neighborhood of $X_2(t)$ denoted S_n^2 contains $\{X_1(t), X_3(t)\}$. The blue-colored incoming edges from $X_1(t)$ and $X_3(t)$ to $X_2(t)$ denote the invariants calculated for state estimation of $X_2(t)$. The self-loop labelled $k_2(t)$ indicates the Kalman filter state estimate for $X_2(t)$ at time t. The latent factors $h_1, \ldots h_k$ are the same as stated previously. Hence, the KASE algorithm performs state estimation as a function of the current Kalman filter state estimate, the current measurements of proximal components as well as the system-wide relationships learned through the latent factors. The system hence seamlessly combines both model-based (Kalman filters) as well as data-driven approaches (latent factor based autoregression) for state estimation.

4 METHODS

In this section, we describe the KASE algorithm for invariant learning and anomaly detection. We also outline the working of the entire **illiad** application.

4.1 Factor Analysis

Let $X_1, \ldots X_n$ denote random variables and $H_1, \ldots, H_k$ denote k hidden factors, and assume that the latent variables are generated using *factor analysis* with the assumption that they can be expressed as linear combinations of the observed variables. Factor analysis attempts to model the variation of the data and hence models the latent sources that cause said variation (De Winter and Dodou 2016). Factor analysis although similar to PCA is more generalizable due to consistency of factor loadings for different feature subsets (Suhr 2005). Details of the exact factor analysis procedure we have used can be obtained from Momtazpour et al. (2015). Other resources for factor analysis are also available (Jöreskog 1967; Harman 1976).

4.2 *KASE*—Kalman Autoregressive State Estimation with Latent Factors and Exogenous Inputs

If we were to once again consider the set $\mathcal{D} = \{x_1(t), x_2(t), \ldots x_n(t)\}$, then according to the ARX model in Jiang et al. (2006), the state estimate for a timeseries at time t is given by

$$\hat{x}_j(t) = \sum_{p=1}^{u} a_p x_j(t-p) + \sum_{p=0}^{v} b_p x_i(t-l-p). \tag{6}$$

Here, $\hat{x}_j(t)$ represents the state estimate of the timeseries x_j at time t and this is calculated as a function of the previous values of timeseries x_j and the values of the exogenous timeseries x_i. The parameters u, v, l represent the order of the model and control the number of previous timesteps that affect the state estimate at the current timestep. a_p and b_p represent weight parameters that control the effect of each of the historical values on the current timestep. In our model, u = v and the values are estimated using cross-validation. In our case, we assume that $l = 0$ as there is no lag.

Incorporating the previous assumptions, the corresponding equation for the Latent Factor ARX model (ARX + ARXL) as outlined in Momtazpour et al. (2015) is

$$\hat{x}_j(t) = \sum_{p=1}^{u} a_p x_j(t-p) + \sum_{p=0}^{u} b_p x_i(t-p) + \sum_{p=0}^{u} \sum_{q=1}^{k} c_{pq} h_k(t-p). \tag{7}$$

Here, k indicates the number of latent factors and u indicates the size of the sliding window we use (the number of previous values we consider). For our experiments, we have set $u = 10$.

We modify Equation (7) to incorporate the Kalman filter state estimates step as follows:

$$\hat{x}_j(t) = \sum_{p=1}^{u} a_p x_j(t-p) + \sum_{p=0}^{u} b_p x_i(t-p) + \sum_{p=0}^{u} \sum_{q=1}^{k} c_{pq} h_k(t-p) + \sum_{p=0}^{u} d_p k_{ji}(t-p). \tag{8}$$

Equation (8) incorporates the Kalman state estimates for timeseries x_j, represented by the symbol $k_{ji}(t)$. The d_p's represent the corresponding weights of each historical state estimate. The symbol $k_{ji}(t)$ indicates the Kalman filter state estimate for $x_j(t)$ calculated using the measurements of timeseries x_j and x_i at time t.

4.2.1 Learning Invariant Based State Estimates. In the context of two timeseries, x_i and x_j, wherein x_j is the output timeseries and x_i is the input timeseries, we calculate the state estimate of $x_j(t)$ as follows:

Let $\bar{\mathbf{x}}_k \in \mathbb{R}^{2X1}$ indicate the Kalman prior state mean of the state at time k, then the transition equations of the Kalman filter is given by

$$\bar{\mathbf{x}}_k = \mathbf{A}\mathbf{x}_{k-1} + \mathbf{B}\mathbf{u}_k + \mathbf{w}_{k-1}. \tag{9}$$

Here, $\mathbf{A} \in \mathbb{R}^{2X2}$ is the transition matrix. $\mathbf{x}_{k-1} \in \mathbb{R}^{2X1}$ indicates the *a posteriori* state estimate at time $k-1$. We ignore the optional control input term $\mathbf{Bu}_k$ and make the assumption as stated previously that $\mathbf{w}_{k-1} \in \mathbb{R}^{2X1}$ is a white-noise process that indicates the error at time $k-1$:

$$\mathbf{y}_k = \mathbf{z}_k - \mathbf{H}\bar{\mathbf{x}}_k, \tag{10}$$

where $\mathbf{H} \in \mathbb{R}^{2X2}$ is the measurement matrix. This, in our case, is the identity matrix. $\mathbf{z}_k \in \mathbb{R}^{2X1}$ are the actual measurements at time k. In the case of the two timeseries x_i (input) and x_j (output),

$$\mathbf{z}_k = \begin{bmatrix} x_j(k) \\ x_i(k) \end{bmatrix}.$$

$\mathbf{y}_k \in \mathbb{R}^{2X1}$ indicates the residual at time k:

$$\mathbf{x}_k = \bar{\mathbf{x}}_k + \mathbf{K}\mathbf{y}_k. \tag{11}$$

$\mathbf{K} \in \mathbb{R}^{2X2}$ represents the Kalman gain and is calculated as a function of the process and measurement covariance matrices $\mathbf{P}$ and $\mathbf{Q}$, and the measurement matrix $\mathbf{H}$[1]
$\mathbf{x}_k \in \mathbb{R}^{2X1}$ represents the posterior state estimate. In the context of the time series x_i and x_j, where x_j is the output timeseries whose state at time t is to be estimated,

$$\mathbf{x}_t = \begin{bmatrix} k_{ji}(t) \\ k_{ij}(t) \end{bmatrix}.$$

Here, $k_{ji}(t) \in \mathbb{R}$ indicates the *a posteriori* state estimate (mean) for timeseries x_j using the exogenous input x_i for time t. If we now refer back to Figure 5, and consider the node $X_2(t)$ in blue, then the self-loop labelled $k_2(t) \in \mathbb{R}^{CX1}$ represents a vector of *a posteriori* state means, where $C = |S_n^2|$ and where the SAIL set S_n^2 of timeseries $X_2 = \{X_1, X_3\}$:

$$k_2(t) = \begin{bmatrix} k_{21}(t) \\ k_{23}(t) \end{bmatrix}.$$

$\mathbf{A}$, $\mathbf{P}$, $\mathbf{Q}$ are estimated (and periodically re-estimated) from historical data using the expectation maximization algorithm (Borman 2004). The initial state mean and covariance μ_0 and Σ_0 are estimated from the data.

As acknowledged by Jiang et al. (2006), Ge et al. (2013), Chen et al. (2010), Chen et al. (2008), and Sharma et al. (2013), ARX only takes into account direct linear relationships. Hence, we retain the latent factor model to take into account the complex underlying relationships between the components in the system. We also consider the model-based Kalman filtering approach owing to the advantages and improvement of system performance in fault detection and health monitoring of hybrid-approaches (model-based + data-driven) discussed in Tidriri et al. (2016).

Algorithm 1 represents the newly designed KASE algorithm that combines both model-based and data-driven approaches in learning the invariant graph. The algorithm defined in Momtazpour et al. (2015) has been augmented to incorporate topological structural assumptions (line 6). At each timestep t, the **UpdateKalman** function (line 16) takes in the measurements of a pair of timeseries $x_i(t)$, $x_j(t)$ and updates the model using Equations (9), (10), and (11). The posterior state estimate of the updated model is then used to learn θ_{ji}^{KASE} . The F_{ji}^*'s are calculated and *superiority threshold* (Δ), *minimum acceptable score* (τ) are enforced similar to Momtazpour et al. (2015). We also adopt the same alerting and anomaly ranking and localization procedure with the *RMSE* evaluation metric. In Section 5, we provide a comparison of the anomaly detection ranking scores of both the original algorithm (ARX + ARXL) proposed in Momtazpour et al. (2015) and our newly proposed KASE algorithm.

[1]The notation for Equations (9), (10), and (11) has been taken from Labbe Jr (2014) and Welch and Bishop (2001).

ALGORITHM 1: KASE Invariant Search Algorithm

```
Input: $x_i, i \in \{1,\dots,n\}$: set of timeseries, $\Delta$: ARX superiority threshold, $\tau$: minimum acceptable score,
       $t_s$ and $t_e$: start and end time of training dataset.
Output: G: Invariant Graph
1  $S_{ARX} = \{\}$;
2  $S_{ARXL} = \{\}$;
3  $S_{KASE} = \{\}$;
4  for $i$ = 1 to $n$ do
5      for $j$ = 1 to $n$ do
6          if $((i == j)$ or $(x_i \notin S_n^j))$ then
7              continue;
8          end
9          foreach $t_s \le t \le t_e$ do
10             Learn an ARX model, $\theta_{ji}^{ARX}$, using Equation (6);
11             Calculate $\hat{x}_j^{ARX}(t)$ using $\theta_{ji}^{ARX}$;
12             Compute $\mathcal{F}_{ji}^{ARX}(t)$ ;
13             Learn an ARXL model, $\theta_{ji}^{ARXL}$, using Equation (7);
14             Calculate $\hat{x}_j^{ARXL}(t)$ using $\theta_{ji}^{ARXL}$;
15             Compute $\mathcal{F}_{ji}^{ARXL}(t)$ ;
16             UpdateKalman($x_i(t), x_j(t)$)     /*Add Measurements*/
17             Learn a KASE model, $\theta_{ji}^{KASE}$, using Equation (8);
18             Calculate $\hat{x}_j^{KASE}(t)$ using $\theta_{ji}^{KASE}$;
19             Compute $\mathcal{F}_{ji}^{KASE}(t)$ ;
20         end
21         if $\left(\sum_{t=t_s}^{t_e} \mathcal{F}_{ji}^{ARX}(t) \ge max(\sum_{t=t_s}^{t_e} \mathcal{F}_{ji}^{ARXL}\ and\ \sum_{t=t_s}^{t_e} \mathcal{F}_{ji}^{KASE}) - \Delta\right)$ and
           $(min_t(F_{ji}^{ARX}(t)) \ge \tau)$ then
22             $S_{ARX} = S_{ARX} \cup \{x_i \rightleftharpoons x_j\}$;
23         else if $\left(\sum_{t=t_s}^{t_e} \mathcal{F}_{ji}^{ARXL} \ge (\sum_{t=t_s}^{t_e} \mathcal{F}_{ji}^{KASE} - \Delta)\right)$ and $(min_t(F_{ji}^{ARXL}(t)) \ge \tau)$ then
24             $S_{ARXL} = S_{ARXL} \cup \{x_i \rightleftharpoons x_j\}$;
25         else
26             /*Indicates that $\mathcal{F}_{ji}^{KASE}$ has highest score*/
27             $S_{KASE} = S_{KASE} \cup \{x_i \rightleftharpoons x_j\}$;
28     end
29 end
```

4.3 *illiad* System Architecture

We outline the functioning of the *illiad* anomaly detection system in this section. Figure 6 depicts the process flow/architecture diagram of the *illiad* system. The system periodically re-trains the Kalman filter and the factor analysis model using historical data while also continuously updating the state estimate of the Kalman filter using new data instances. The system also has a front-end dashboard with the invariant graph depicted on screen aiding the system maintainer to easily glean whether the system has anomalous components. Figures 7 and 8 represent the dashboard of the wireless sensor motes temperature sensor network and the IEEE 33 bus microgrid network described later in Section 5.1. The dashboard is interactive and allows for the user to click on nodes in the network to view various historical data and metrics about them. In case of an anomaly,

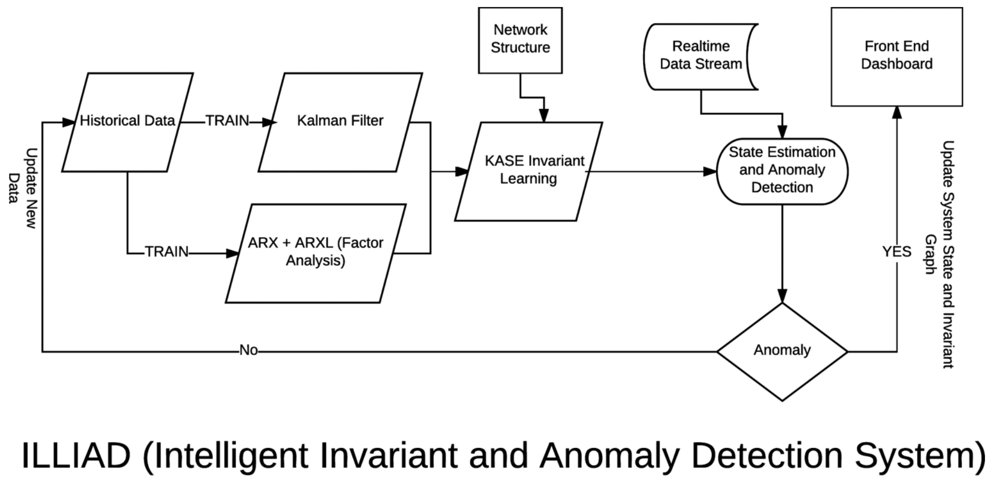
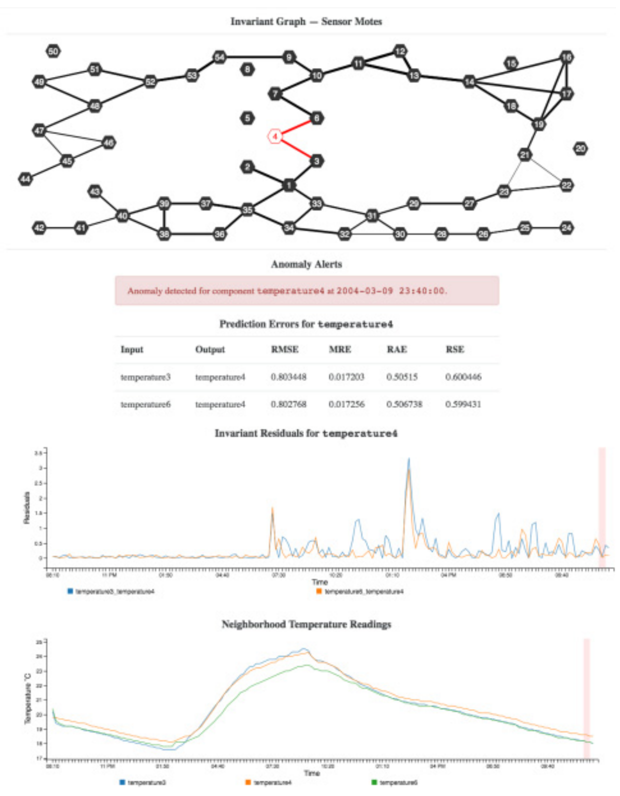

Network Structure
Realtime Data Stream
Front End Dashboard
Historical Data
TRAIN
Kalman Filter
KASE Invariant Learning
State Estimation and Anomaly Detection
Update New Data
TRAIN
ARX + ARXL (Factor Analysis)
YES
Update System State and Invariant Graph
No
Anomaly
ILLIAD (Intelligent Invariant and Anomaly Detection System)

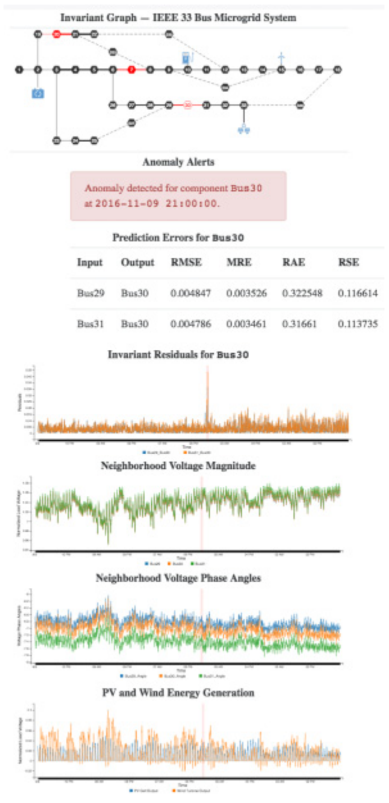
Fig. 6. illiad System Architecture.


Invariant Graph — Sensor Motes
Anomaly Alerts
Invariant Residuals for temperature4
Residuals
Time
temperature3_temperature4
temperature6_temperature4
Neighborhood Temperature Readings
Temperature °C
Time
temperature3
temperature4
temperature6

Anomaly detected for component temperature4 at 2004-03-09 23:40:00.

Prediction Errors for temperature4

Input	Output	RMSE	MRE	RAE	RSE
temperature3	temperature4	0.803448	0.017203	0.50515	0.600446
temperature6	temperature4	0.802768	0.017256	0.506738	0.599431


Invariant Graph — IEEE 33 Bus Microgrid System
Anomaly Alerts
Invariant Residuals for Bus30
Neighborhood Voltage Magnitude
Neighborhood Voltage Phase Angles
PV and Wind Energy Generation

Anomaly detected for component Bus30 at 2016-11-09 21:00:00.

Prediction Errors for Bus30

Input	Output	RMSE	MRE	RAE	RSE
Bus29	Bus30	0.004847	0.003526	0.322548	0.116614
Bus31	Bus30	0.004786	0.003461	0.31661	0.113735

Fig. 7. illiad Dashboard - Sensormotes.

Fig. 8. illiad Dashboard - IEEE 33 Bus Microgrid.

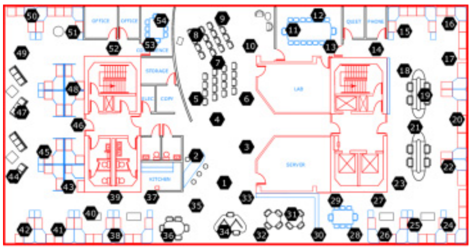


Fig. 9. Lab Layout Sensor Motes.

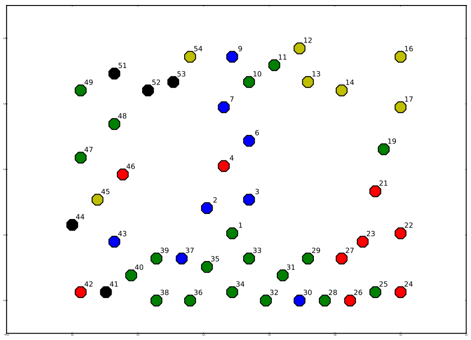


Fig. 10. Clustering of Sensors Based on Original Temperature Readings. A fair degree of spatial correlation can be observed.

an alert is issued (indicated by a red box under the invariant graph) on the dashboard and the corresponding broken invariants are highlighted in red as shown in the figures. It is apparent from the broken invariants in Figure 7 that Sensor 4 has an anomaly. The timeseries graphs at the bottom of the dashboard represent the residuals of each of the invariants of the anomalous sensor (sensor 4 in Figure 7 and Bus 30 in Figure 8). In addition to residuals, the actual temperature readings and the voltage magnitudes, phase angles, and PV, Wind Generation statistics are also included in their respective dashboards. The *illiad* system can thus be used for system health monitoring and fault detection both for wired and wireless cyber-physical systems.

5 EXPERIMENTAL RESULTS

We present the application of our methodology to two diverse datasets in urban computing. Through these experiments, we seek to showcase that our system is applicable to both wireless and wired sensor networks in real-world settings. We will showcase how structural and proximal relationships are inferred automatically in the case of wireless sensor networks. Most importantly, we wish to demonstrate how our system leads to an improvement in the quality of anomaly and invariant detection over existing benchmarks. To this end, we conduct experiments to showcase the strength of invariant graphs learned by *illiad* as well as its ability to accurately identify anomalies even with a relatively sparse invariant graph. We also discuss the benefits of the *SAIL* invariant learning approach over the combinatorial invariant learning approach used in prior work by analyzing the runtime behavior of the two systems.

We first describe the application of *illiad* to a wireless sensor network and subsequently to a microgrid system.

5.1 Sensor Motes

Dataset Description: The sensor motes dataset contains measurements from wireless sensors at Intel Berkeley Research lab.[2] There are a total of 54 sensors located at a lab measuring temperature, humidity, light, and voltage between February 28 and April 5, 2004. Each sensor was able to record different variables every 31s. Figure 9 shows the location of each sensor in different parts of the lab. We focus on temperature recordings between February 28 and March 10 for our study.

[2]http://db.csail.mit.edu/labdata/labdata.html.

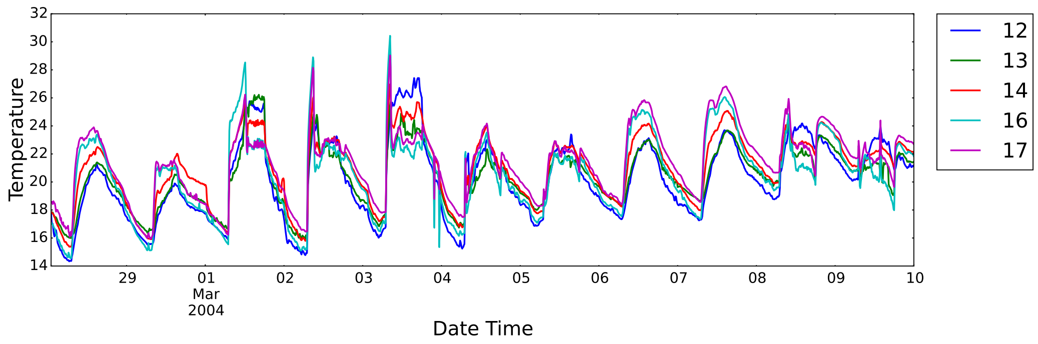


Fig. 11. Temperature Trends of Sensors in the top right yellow cluster in Figure 10.

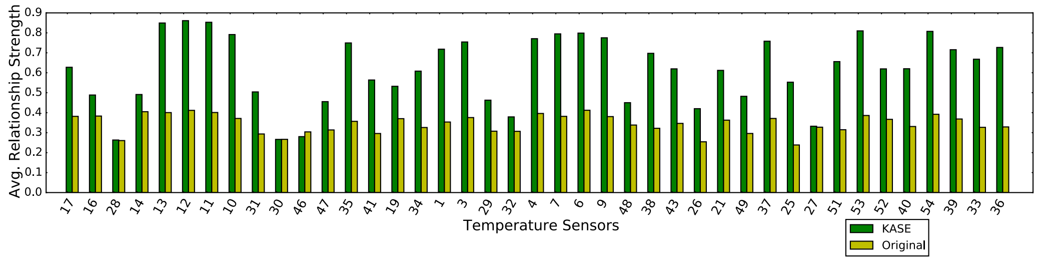


Fig. 12. Average Relationship Strength Per Temperature Sensor.

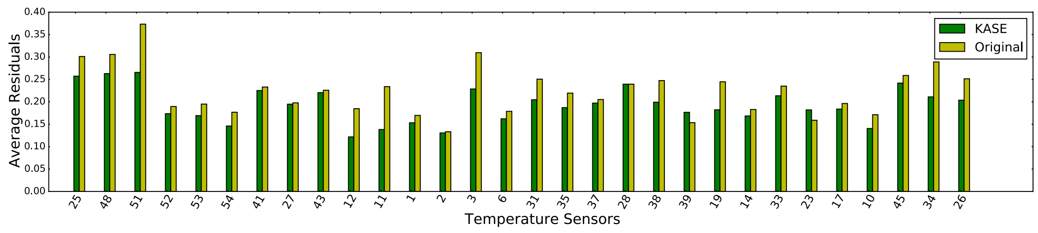


Fig. 13. Average Residual Per Temperature Sensor.

Data Processing: We cleanse the data by eliminating sensors with a large number of missing entries. The missing temperature readings in the remaining set of sensors were addressed using linear interpolation (Meijering 2002). The dataset was then down-sampled to 10min intervals with the mean as the sampling rule. Figure 10 depicts a representation of all the filtered set of sensors used in our analysis, clustered according to their temperature readings. Figure 11 shows the temperature values of a group of sensors that are part of the same cluster. We can see that all the sensors in the figure showcase similar patterns of temperature variation.

Results and Discussion: In Figure 12, we plot the average normalized relationship strength ($1 - avg\_prediction\_error$) per temperature sensor. The relationship strength is indicative of how well the invariants that a sensor is involved in are able to estimate the current state of the sensor. We observe that the relationship strengths of sensors obtained using KASE is higher than that obtained using the original (ARX + ARXL) method indicating that KASE is able to infer a higher number of invariant relationships that are stronger. This theory is further corroborated by Figure 13, which depicts the average residuals per temperature sensor; in this case as well, we

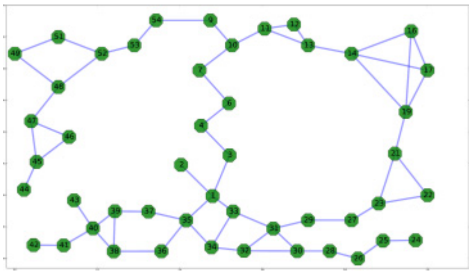


Fig. 14. Original Nearest-Neighbor Graph (based on sensor location information).

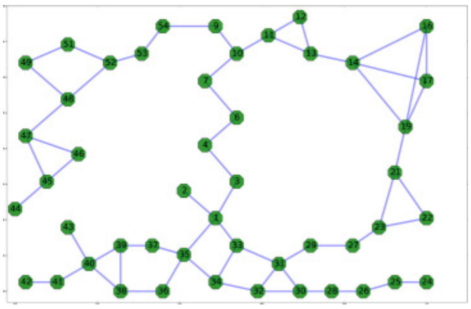


Fig. 15. Invariant Graph Learned from KASE algorithm.

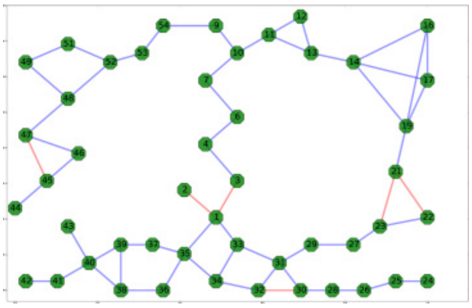


Fig. 16. Invariant Graph Learned from ARXL + ARX method (with SSAIL assumption).

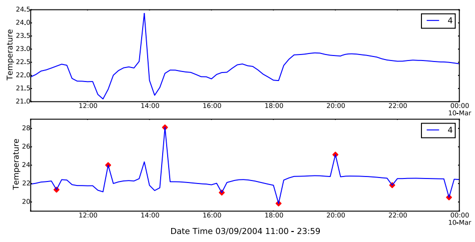


Fig. 17. Temperature Sensor Number 4, Original Readings vs. Anomaly Injection Readings Snapshot.

observe that KASE is able to generalize better on the test set and produce more accurate temperature state estimates. The network structure of the wireless sensor network is derived using the *SAIL* method (using sensor locations) described in Section 3 and is shown in Figure 14. The invariant graph learned using this network from the KASE algorithm is depicted in Figure 15. We observe that the invariant graph is identical to the original network structure shown in Figure 14. Figure 16 indicates a similar invariant graph obtained by executing the original (ARX + ARXL) algorithm with the *SAIL* neighborhood constraint on the same temperature sensormotes dataset. We observe that the number of edges learned in the invariant graph is lower than in the case of KASE. The missing edges are depicted in red in Figure 16.

Anomaly Detection: Apart from the invariant discovery and state estimation accuracy, we also conducted experiments to test the anomaly detection capabilities of the KASE algorithm in comparison to the original (ARX + ARXL) algorithm. The experiment was carried out by first injecting anomalies into temperature sensor number 4 between 11 a.m. and 23:59 p.m. on March 9, 2004, and then running the two algorithms (original ARX + ARXL and the KASE algorithm) to understand the behavior of the two systems when faced with an anomaly. The anomalous snippet of the time series of sensor 4 and the original timeseries have been shown in Figure 17. The anomalies were generated by adding Gaussian noise with 0 mean and 2.0 standard deviation to the sensor 4 temperature values at certain pre-determined timepoints between 11 a.m. and 23:59 p.m. on March 9, 2004. These anomalies have been indicated in red in Figure 17. We use the anomaly ranking algorithm used in Momtazpour et al. (2015) to quantify and compare the anomaly detection quality

Table 1. Anomaly Detection Score Sensormotes

Component	Method Name	Remaining Links	Broken Links	Ranking Score
Sensor 4	Original (ARX + ARXL)	14	15	0.517
Sensor 4	KASE	0	2	1.0

of both methods. The results of the anomaly detection are indicated in Table 1. Here, we observe that although a significant number of invariants concerning the anomalous sensor (sensor 4) are broken, there are almost an equivalent number of invariants still existing in the system indicating that these invariants do not register the anomaly. The KASE algorithm, however, recognizes the anomaly by breaking all the existing invariants (in this case sensor 4 initially has two invariants, which are both broken) and hence obtaining a higher anomaly score and also providing for a more interpretable detection mechanism.

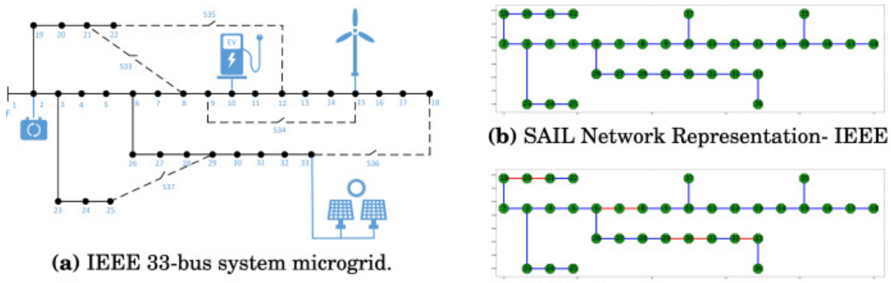
Discussion: State estimation in the current *illiad* system could suffer if the sensor distribution is significantly skewed spatially. Since we consider the spatially closest neighbors to a particular sensor as candidates for invariant learning, a significant skew in that distribution could make estimation challenging due to sparse data in some locations. This in turn could lead to fewer invariants inferred and thus diminished capacity for anomaly detection. The use of Gaussian processes for dynamic sensor placement (Ramakrishnan et al. 2005; Krause et al. 2008) can aid in judicious placement of sensors. A second issue in generalization has to do with potential mobility of sensors in practical settings (in which case, some violations of invariants could be normal). In IoT settings, we need to inherently model sensor mobility into the analytics engine.

5.2 Electric Power Microgrid

Dataset Description: This simulated dataset is based on the (real) IEEE 33-bus standard distribution power system, a commonly used example distribution network. It is composed of 33 electric buses or nodes. Bus 1 is a transformer connecting the distribution level system and bulk transmission power system. It is the feeder to the whole system, through which all the electric power flows into the network. It serves as the regular energy source. The other 32 buses are all load buses initially, which means that they only consume energy. We model all energy consumers and not the energy source in Bus 1 during our experiments as the voltage and power of the transformer is assumed constant throughout the dataset generation process. To imitate a typical mircrogrid, we have integrated multiple non-traditional elements into the system, including two types of renewable distributed energy sources: PV and wind turbines, electric vehicles, and an energy storage device.

Data Processing: The hourly operating conditions of this distribution system are simulated for a year with a 1h sampling rate, which is 8,760 cases in total, using the MatPower package (Zimmerman and Gan 1997). We conducted 8,760 power flow calculations based on hourly scales of the regular and irregular load and the hourly outputs of the renewable distributed generation. The load consumption, solar radiation and wind intensity values required as input for data generation were all considered for the city of Richmond, Virginia USA, to ensure consistency between the electric load, and the energy produced using distributed generation. The 8,760h load data has been taken from a dataset containing commercial and residential hourly load profiles for all TMY3 locations in the United States.[3] For different load buses, we use 5% uniformly distributed deviation to generate different load curves that follow the same general pattern. The load at Bus 33 has a

[3]https://en.openei.org/datasets/dataset/commercial-and-residential-hourly-load-profiles-for-all-tmy3-locations-in-the-united-states.



(a) IEEE 33-bus system microgrid.

(b) SAIL Network Representation- IEEE 33 Bus

(c) KASE Algorithm Broken Invariants Representation

Fig. 18. Original Network IEEE 33 Bus with Invariants and Anomalies.

solar panel attached to it. The solar radiation data used to generate the synthetic PV output is available from the National Solar Radiation Data Base.[4] One wind turbine is attached to Bus 15, and the power output of a wind farm located close to Richmond the data for which was obtained from the National Renewable Energy Laboratory Wind Prospector Data Set[5] was used as a reference for the simulation. We have also attached an EV charging station to Bus 10. A one-year charging load for this station was obtained from Idaho National Laboratory EV Project Quarterly and Annual Report Data.[6] An energy storage battery is attached to Bus 2. For simplicity, we adopt a charging/discharging strategy that assumes that the battery is charged during the load valley from 00:00 a.m. to 8:00 a.m. and that it is discharged from 9:00 a.m. to 11:00 p.m. The physical connections among the various components of the microgrid system have been specified in Figure 18(a) . It must be noted that the energy storage device depicted at Bus 2 has not been modeled as an explicit separate component; instead the load at Bus 2 has been modified appropriately to depict behavior of having an energy storage device attached to it. Hence, we do not model the battery as a separate component in our experiments.

Results and Discussion: Figure 18(a) represents the circuit diagram of the IEEE 33 Bus network used in our experiments. We use the network structure depicted in the figure (excluding the dashed lines that depict open switches) to learn the SAIL neighborhoods of each of the 35 components. The Wind turbine, PV Cell, and the Electric Vehicle Load have been renamed Bus 35, Bus 36, and Bus 37, respectively. The SAIL neighborhood learned by the KASE algorithm has been depicted in Figure 18(b) and the broken links due to anomalies have been depicted in Figure 18(c). We see that the invariants learned by the KASE algorithm correspond to the original circuit diagram. The KASE has three sub-components—ARX, ARXL, Kalman State Estimation. These three algorithms each try to learn relationships between each pair of invariants. From Algorithm 1 (lines 21 to 29), it is clear that the invariant learning process attempts to select the strongest relationship between a pair of components learned by ARX, ARXL and Kalman state estimation. Hence, we can conclude that the algorithm that contributes the most number of invariants to the graph has the property of learning stronger relationships between components. Table 2 depicts the number of invariants learned as a function of Maximum Acceptable Error (MAE). The MAE is the maximum prediction error (per invariant) below which invariant-relationships learned by any of the aforementioned procedures qualify for selection (this can also be considered to be $1 - \tau$, where τ indicates the minimum acceptable invariant-relationship strength) during the invariant learning process.

[4] http://rredc.nrel.gov/solar/old_data/nsrdb/1991-2005/tmy3/by_state_and_city.html.
[5] https://mapsbeta.nrel.gov/wind-prospector/?visible=wind_3tier_site_metadata.
[6] https://avt.inl.gov/project-type/downloads.

Table 2. Number of Invariants Learned as a Function of MAE

Max. Acceptable Error (MAE)	Original_SAIL	KASE	KASE_ARX	KASE_ARXL	KASE_KSE
0.0001	0	0	0	0	0
0.001	11	12	0	3	9
0.002	20	21	0	6	15
0.003	22	27	0	6	21
0.004	34	38	0	8	30
0.005	46	48	0	15	33
0.006	66	67	0	17	50
0.0851	68	68	0	18	50

Table 3. IEEE 33 Bus Avg. Residual, Avg. Errors, and Percentage Improvement of KASE over Original

Name	Original	KASE	% Improvement
RMSE	0.004153	**0.003964**	4.5639
RSE	0.148742	**0.134409**	9.6363
RAE	0.365744	**0.347662**	4.944
Residual	0.002909	**0.0027889**	4.14

The original_SAIL column represents the (ARX + ARXL) original algorithm run using the SAIL neighborhood assumption. The *KASE* column represents Algorithm 1 and KASE_ARX, KASE_ARXL, and KASE_KSE represent the number of invariants learned by each of ARX, ARXL, and Kalman state estimation sub-components of KASE, respectively. We observe that for any particular value of Maximum Acceptable Error, KASE learns a greater subset of the true set of invariants (the IEEE 33 Bus network has a total of 68 invariants) than the Original_SAIL algorithm. Even amongst the KASE sub-components, we can see that a majority of edges are contributed by the KSE procedure indicating that the Kalman filtering procedure plays a major role leading to the KASE algorithm learning a more diverse set of edges and stronger invariant relationships as compared to the Original (ARX + ARXL) algorithm. We provide quantitative proof of our claim that the KASE algorithm is indeed a better state-estimator by recording the average values for residual, RMSE (root mean squared error), RSE (relative squared error), RAE (relative absolute error) on the test set in Table 3 and Figures 19, 20, and 21. We find that the KASE algorithm achieves a significant percentage improvement over the Original (ARX + ARXL) algorithm in all cases. We must note that we have left out the componenents (Bus20, Bus30, Bus7) in which anomalies were injected in the test set as they will skew the results with high error values.[7] Since this table specifically has been included to showcase the state-estimation capability of the KASE algorithm, we feel excluding the anomalous nodes will have no effect as we have already demonstrated the anomaly detection capabilities in Table 4.

Anomaly Detection: We introduce anomalies in the data to simulate a load surge at certain time steps by decreasing the voltage at three different buses. We decrease the voltage magnitude at Bus 7 between time steps 7,500 and 7,565. We also decrease the voltage at Bus 20 from time step 8,500 to 8,550 and at Bus 30 from time step 8,500 to 8,545 and showcase the effect of these anomalies on the anomaly detection mechanism. These anomalies test the ability of the system not only to detect and report the anomalies, but also the localization capability of the system in the presence

[7]If there is an anomaly, the system under/overestimates the state significantly, which is what is indicative of the anomaly.

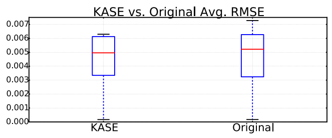


Fig. 19. KASE vs. Original (ARX+ARXL) Average RMSE. IEEE 33 Bus Microgrid System.

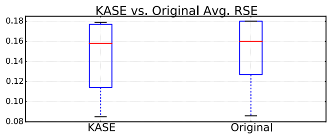


Fig. 20. KASE vs. Original (ARX+ARXL) Average RSE. IEEE 33 Bus Microgrid System.

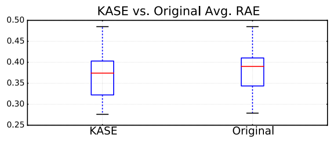


Fig. 21. KASE vs. Original (ARX+ARXL) Average RAE. IEEE 33 Bus Microgrid System.

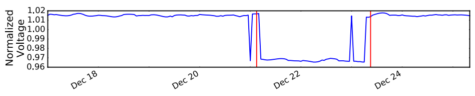


Fig. 22. Anomaly Snapshot Bus 20.

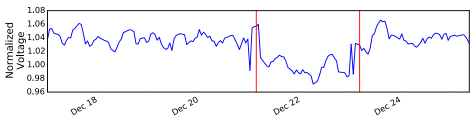


Fig. 23. Anomaly Snapshot Bus 30.

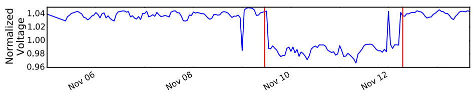


Fig. 24. Anomaly Snapshot Bus 7.

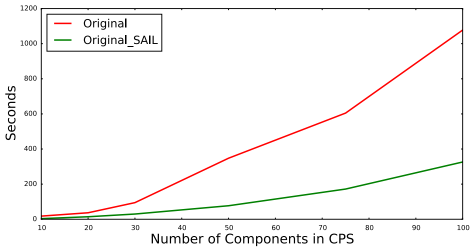


Fig. 25. Running Time of Original (ARX + ARXL) vs. Original (ARX+ARXL) + SAIL.

Table 4. Ranking and Anomaly Localization

Component	Method	Remaining Links	Broken Links	Score
Bus7	Original(ARX+ARXL)	6	14	0.7
	KASE	0	2	1.0
Bus20	Original(ARX+ARXL)	11	21	0.66
	KASE	0	2	1.0
Bus30	Original(ARX+ARXL)	4	13	0.76
	KASE	0	2	1.0

of a single as well as multiple anomalous components. The voltages of the three buses around the time steps of their respective anomalies have been depicted in Figures 22, 23, and 24. In each figure the anomalous region has been bound by vertical red lines. Since the invariant relationships learned are stronger, the anomaly detection procedure is also able to perform in a more robust manner when it encounters an anomaly. This can be seen in Figure 18(c) where all the invariants of Buses 7, 20, and 30 are broken in response to the injected anomalies. We also show that the KASE algorithm is better at ranking anomalies; on average a higher score is learned using the ranking procedure from Momtazpour et al. (2015). Table 4 shows that for each of the three buses, all invariants involving the anomalous bus are broken for the KASE algorithm yielding a higher anomaly score relative to the Original (ARX + ARXL) algorithm, further enforcing our belief that the KASE algorithm proposed in this article learns stronger and more robust invariants most or all of which are violated in case an anomaly occurs in the system. Finally, we depict the advantage of the SAIL methods toward increasing the scalability of the invariant learning algorithms discussed in this article, thereby enabling the algorithms to be run with larger datasets. To conduct this experiment, we generated a random matrix $X \in \mathbb{R}^{mXn}$, where we varied n between 10 and 100. Here, n represents the number of unique timeseries (or unique components in the CPS). We then ran both the Original and the Original method with the SAIL assumption. We simulated the SAIL neighborhood by assuming that the underlying network of the CPS had a 0.3 graph density (each component is connected to 30% of the other components in the network), which is much denser than either of our examples depicted in this article and is much denser than most connected CPSs. Figure 25 shows the significant advantage in terms of running time of incorporating the network information in the invariant learning. The figure clearly shows that the Original (ARX + ARXL) algorithm with the SAIL assumption scales much better than its non SAIL counterpart.

Discussion: Although we have taken great care to model our microgrid design using actual load profiles, and real solar and wind intensity data to simulate the generation of the PV and wind turbines, the lack of open datasets for microgrids is a serious bottleneck in this area of research. Similarly, our injection of anomalies can be generalized into a greater taxonomy of faults, for example, component failure, cyber-attacks, and changing environmental situations. We aim to build upon the work here to develop a broader framework for microgrid analytics.

5.3 System Function—Wireless versus Wired Networks

It is instructive to compare and contrast our wired and wireless network studies. While we have shown that our method works in both settings, wireless networks exhibit broader profiles of fault-tolerance not witnessed in wired networks (e.g., a sensor malfunction in a wireless network might not affect the overall system dynamics as much as a component failure in a microgrid might, for example, potentially leading to cascading failures). The illiad system has a built-in rule to raise an alert when a a threshold number of samples from some component of the CPS have failed to register but more elaborate fault models can be explored within our framework.

5.4 System Characteristics and Minimum Requirements

The proposed system has been shown to work with both wireless and wired sensor networks and has been tuned to work with power systems like microgrids as well as sensors that measure physical indicators such as temperature. Further, the system is able to perform with either static or streaming data, in both wired and wireless settings. The system requires either an absolute physical network diagram of the sensor network or a logical connectivity diagram so the invariant learning can leverage this prior information as domain knowledge and further contribute to interpretability.

6 CONCLUSIONS AND FUTURE WORK

We have presented a system for anomaly detection and state estimation for wireless and wired sensor networks. We have tested our models on multiple datasets (both real-world and synthetic) and demonstrated the improvement in performance in comparison to the baseline method. Our application successfully combines model-based (Kalman filter) and data-driven (auto-regression and latent factor-based methods) approaches to learn a better state representation of the system under surveillance. This higher accuracy in state estimation is achieved through learning stronger relationships between various components in the networks while also being sensitive to potential violations in these relationships resulting in anomalies. The sparse network structure of invariant relationships learnt through the network-structure aware invariant learning procedures makes for a scalable system, wherein the state of the system is easily interpretable by human experts tasked with overseeing system maintenance. The real-time dashboard and the alerting system aid further in this regard. The field of invariant discovery is vast as there are many varieties of latent, direct, indirect, simple, and complex relationships amongst components in cyber-physical systems. Power systems are rife with non-linearities, like voltage phase-angles. Incorporating these components into the invariant learning is a logical next step to improve the state estimation capabilities of the system. Further, each component in a power system interacts with other components in compliance with certain well-known laws of physics and electricity; thus, augmenting the anomaly detection procedure to be cognizant of these relationships would be a useful direction of future work.

REFERENCES

Sean Borman. 2004. The expectation maximization algorithm: A short tutorial. Retrieved from http://bit.ly/2hAed7x.

Sanjay Chawla, Yu Zheng, and Jiafeng Hu. 2012. Inferring the root cause in road traffic anomalies. In *Proceedings of ICDM (2012).*

Haifeng Chen, Haibin Cheng, Guofei Jiang, and Kenji Yoshihira. 2008. Exploiting local and global invariants for the management of large scale information systems. In *Proceedings of ICDM (2008).*

Haifeng Chen, Guofei Jiang, Kenji Yoshihira, and Akhilesh Saxena. 2010. Invariants based failure diagnosis in distributed computing systems. In *Proceedings of the IEEE SRDS*. 160–166.

Zhe Chen. 2003. Bayesian filtering: From kalman filters to particle filters, and beyond. *Statistics* 182, 1 (2003).

S. Chowdhury and P. Crossley. 2009. *Microgrids and Active Distribution Networks.* The Institution of Engineering and Technology.

Inigo Cobelo, Ahmed Shafiu, Nick Jenkins, and Goran Strbac. 2007. State estimation of networks with distributed generation. *Eur. Trans. Electric. Power* 17, 1 (2007), 21–36.

Joost De Winter and Dimitra Dodou. 2016. Common factor analysis versus principal component analysis: A comparison of loadings by means of simulations. *Commun. Stat.-Simul. Comput.* 10 (2016), 299–321.

Yong Ge, Guofei Jiang, and Yuan Ge. 2013. Efficient invariant search for distributed information systems. In *Proceedings of ICDM (2013).*

Harry H. Harman. 1976. *Modern Factor Analysis* (3rd ed.). University of Chicago Press.

Ying Hu, Anthony Kuh, Aleksandar Kavcic, and Dora Nakafuji. 2011. Real-time state estimation on micro-grids. In *Proceedings of IEEE IJCNN.* 1378–1385.

Guofei Jiang, Haifeng Chen, and Kenji Yoshihira. 2006. Discovering likely invariants of distributed transaction systems for autonomic system management. In *Proceedings of ICAC.* 199–208.

Karl G. Jöreskog. 1967. Some contributions to maximum likelihood factor analysis. *Psychometrika* 32, 4 (1967), 443–482.

Rudolph Emil Kalman. 1960. A new approach to linear filtering and prediction problems. *Trans. ASME J. Basic Eng.* 82 (1960), 35–45.

George N. Korres, Nikos D. Hatziargyriou, and Petros J. Katsikas. 2011. State estimation in multi-microgrids. *Eur. Trans. Electric. Power* 21, 2 (2011), 1178–1199.

Andreas Krause, Ajit Singh, and Carlos Guestrin. 2008. Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *J. Mach. Learn. Res.* 9, Feb (2008), 235–284.

Roger R. Labbe Jr. 2014. *Kalman and Bayesian Filters in Python* (1st ed.). Github.com.

Erik Meijering. 2002. A chronology of interpolation: From ancient astronomy to modern signal and image processing. *Proc. IEEE* 90, 3 (2002), 319–342.

Marjan Momtazpour, Jinghe Zhang, Saifur Rahman, Ratnesh Sharma, and Naren Ramakrishnan. 2015. Analyzing invariants in cyber-physical systems using latent factor regression. In *Proceedings of SIGKDD.* ACM.

A. Monticelli. 2000. Electric power system state estimation. *Proc. IEEE* 88, 2 (2000), 262–282.

K. Nishiya, J. Hasegawa, and T. Koike. 1982. Dynamic state estimation including anomaly detection and identification for power systems. *IEEE Proc. Gen. Trans. Distr.* 129, 1 IET (1982), 192–198.

Naren Ramakrishnan, Chris Bailey-Kellogg, Satish Tadepalli, and Varun N. Pandey. 2005. Gaussian processes for active data mining of spatial aggregates. In *Proceedings of SDM.* 427–438.

Md Masud Rana and Li Li. 2015. Kalman filter based microgrid state estimation using the internet of things communication network. In *Proceedings of ITNG.* IEEE.

Abhishek B. Sharma, Haifeng Chen, Min Ding, Kenji Yoshihira, and Guofei Jiang. 2013. Fault detection and localization in distributed systems using invariant relationships. In *Proceedings of IEEE DSN.*

Diana D. Suhr. 2005. Principal component analysis vs. exploratory factor analysis. *Proceedings of SUGI 30* 203 (2005).

Khaoula Tidriri, Nizar Chatti, Sylvain Verron, and Téodor Tiplica. 2016. Bridging data-driven and model-based approaches for process fault diagnosis and health monitoring: A review of researches and future challenges. *Annu. Rev. Control* (2016), 63–81.

Antonis G. Tsikalakis and Nikos D. Hatziargyriou. 2011. Centralized control for optimizing microgrids operation. In *Proceedings of the IEEE Power and Energy Society General Meeting*.

Yanbo Wang, Yanjun Tian, Xiongfei Wang, Zhe Chen, and Yongdong Tan. 2014. Kalman-filter based state estimation for system information exchange in a multi-bus islanded microgrid. In *Power Electronics, Machines and Drives*.

Greg Welch and Gary Bishop. 2001. An introduction to the kalman filter. *In SIGGRAPH Course Notes.* 1–80.

Yu Zheng, Licia Capra, Ouri Wolfson, and Hai Yang. 2014. Urban computing: Concepts, methodologies, and applications. *ACM TIST* 5, 3 (2014), 38.

Yu Zheng, Huichu Zhang, and Yong Yu. 2015. Detecting collective anomalies from multiple spatio-temporal datasets across different domains. In *Proceedings of ACM SIGSPATIAL*.

R. Zimmerman and D. Gan. 1997. Matpower. Power Systems Engineering Research Center, School of Electrical Engineering, Cornell University, Ithaca, NY.

Received December 2016; revised February 2017; accepted March 2017