2020 American Control Conference
Denver, CO, USA, July 1-3, 2020

On Data-driven Attack-resilient Gaussian Process Regression for Dynamic Systems

Hunmin Kim, Pinyao Guo, Minghui Zhu, and Peng Liu

Abstract— **This paper studies attack-resilient Gaussian process regression of partially unknown nonlinear dynamic systems subject to sensor attacks and actuator attacks. The problem is formulated as the joint estimation of states, attack vectors, and system functions of partially unknown systems. We propose a new learning algorithm by incorporating our recently developed unknown input and state estimation technique into the Gaussian process regression algorithm. Stability of the proposed algorithm is formally studied. We also show that average case learning errors of system function approximation are diminishing if the number of state estimates whose estimation errors are non-zero is bounded by a constant. We demonstrate the performance of the proposed algorithm by numerical simulations on the IEEE 68-bus test system.**

I. Introduction

Machine learning is increasingly used in cyber-physical systems (CPS) for a broad area of applications such as image recognition in self-driving vehicles, control of energy systems, and healthcare systems. These data-driven techniques are well suited for complex systems whose models are challenging to obtain. However, machine learning systems are threatened by cyberattacks [1], [2]. Protecting machine learning systems from cyberattacks is imperative.

Literature review. The problem of interest is related to machine learning in the presence of training data errors; i.e., fault tolerant learning [3], [4], and adversarial learning [5]. They focus on probably approximately correct learning in the presence of (malicious) noises; i.e., they aim to design and analyze algorithms which generate approximately correct results with high probability. Fault tolerant learning focuses on complete random errors, while adversarial learning considers adversarial errors which maximize the learning error. Unlike the papers mentioned above, this paper studies attack-resilient machine learning; i.e., obtaining correct machine learning despite that training data are potentially corrupted.

Our work is closely related to data-driven learning of unknown dynamic systems using Gaussian process regression (GPR) [6]. Gaussian process (GP) is proven as an efficient tool for system identification [7], fault detection [8] and control [9] of unknown dynamic systems. In [10], GPR is amalgamated with extended Kalman filter; i.e., the GP-EKF algorithm to achieve fault detection of completely unknown dynamic systems. To enhance identification accuracy, papers [10], [11] replace extended Kalman filter by unscented Kalman filter and apply the algorithm to sensor fault detection [10] and state estimation [11]. In [12], GPR is integrated with square root Cubature Kalman filter, which is computationally cheaper and numerically more reliable than the GP-EKF algorithm. None of the above data-driven techniques is applicable to handle attacks or attack-resilient learning. No theoretic guarantee is provided in the above.

Contribution. We present a new attack-resilient GPR algorithm for partially unknown nonlinear dynamic systems subject to sensor attacks and actuator attacks. We incorporate our recently developed unknown input and state estimation technique [13] into the Gaussian process regression algorithm to address the challenge that the system function is unknown. GPR then uses the history of the estimates to approximate the system function. Average case learning errors of system function approximation are expected to diminish. We show the performance of the proposed algorithm by numerical simulations on the IEEE 68-bus test system.

II. Problem formulation

Consider the nonlinear stochastic system

$$\begin{aligned} x_k &= f(x_{k-1}, u_{k-1} + d_{a,k-1}) + w_{k-1} \\ y_k &= C_k x_k + d_{s,k} + v_k \end{aligned} \tag{1}$$

where $x_k \in \mathbb{R}^n$, $y_k \in \mathbb{R}^m$, $u_k \in \mathbb{R}^a$, $d_{a,k} \in \mathbb{R}^a$ and $d_{s,k} \in \mathbb{R}^m$ are state, output, input, actuator attack vector, and sensor attack vector, respectively. We assume that noise vectors $w_k \in \mathbb{R}^n$ and $v_k \in \mathbb{R}^m$ are independent and identically distributed zero-mean Gaussian, with covariance matrices $Q \triangleq \mathbb{E}[w_k w_k^T]$ and $R \triangleq \mathbb{E}[v_k v_k^T]$.

Attack model. Signal injection attacks are comprised of signal magnitude attacks; i.e., the attacker injects attack signals, and signal location attacks; i.e., the attacker chooses targeted sensors and actuators. Signal injection attacks are modeled by actuator attack $d_{a,k}$ and sensor attack $d_{s,k}$ where zero value of either attack vector indicates that the corresponding actuator or sensor is free of attack, and a non-zero value indicates the magnitude of the attack.

Knowledge of the defender. System function f in (1) is unknown to the defender while output matrix C_k is known. The defender is accessible to input u_k and output y_k but is

Hunmin Kim (hunmin@illinois.edu) is with the Mechanical Science and Engineering, University of Illinois, 1206 West Green St., Urbana, IL 61801.

Pinyao Guo (pinyao.guo@airbnb.com) is with Airbnb Inc., 888 Brannan St, San Francisco, CA 94103, USA. This work was conducted when the author was affiliated to the College of Information Sciences and Technology, Pennsylvania State University.

Minghui Zhu (muz16@psu.edu) is with the School of Electrical Engineering and Computer Science, Pennsylvania State University, 201 Old Main, University Park, PA 16802.

Peng Liu (pliu@ist.psu.edu) is with the College of Information Sciences and Technology, Pennsylvania State University, 201 Old Main, University Park, PA 16802.

This work is partially supported by the National Science Foundation (CNS-1505664, ECCS-1846706) and the College of Information Sciences and Technology at the Pennsylvania State University.

978-1-5386-8266-1/$31.00 ©2020 AACC

unaware of the attack vectors $d_{a,k}$ and $d_{s,k}$, as well as which actuators/sensors are under attacks. Noise vectors w_k, v_k and autocovariance Q are unknown but R is known.

Objective. The defender aims to recursively estimate state x_k, attack vectors $d_{a,k}$, $d_{s,k}$ and learn system function f in the presence of sensor attacks and actuator attacks.

III. Preliminaries

This section summarizes the notations and notionsr. It also discusses classic GPR following the presentation in [6].

A. Notations and notions

Hat notation over a variable denotes an estimate of the variable. In particular, $\hat{x}_{k|k-1}$ is a predicted state (an estimate without the current output); $\hat{x}_k$ is a state estimate (an estimate with the current output); $\hat{d}_k$ is an estimate of attack vector of d_k; and $\hat{f}$ is an approximation of function f. Also, $\tilde{a}_k \triangleq a_k - \hat{a}_k$ denotes the estimation error and $P_k^a \triangleq \mathbb{E}[\tilde{a}_k \tilde{a}_k^T]$ denotes the error covariance of a_k. Let $dim(v)$ denote the dimension of vector v. Gaussian distribution is denoted by $\mathcal{N}(\mu, \Sigma)$, where μ is mean and Σ is covariance.

Definition 3.1: (Definition 6.1 in [6]) Hilbert space $\mathcal{H}$ of real functions f defined on $\mathcal{X}$ is called a reproducing kernel Hilbert space (RKHS) endowed with an inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ if there exists a unique function g such that for every $x \in \mathcal{X}$, $g(x, x')$ as a function of x' belongs to $\mathcal{H}$, and g has the reproducing property $\langle f(\cdot), g(\cdot, x) \rangle_{\mathcal{H}} = f(x)$. ■

In the above definition, function g is called kernel, and $\|f\|_{\mathcal{H}} \triangleq \sqrt{\langle f, f \rangle_{\mathcal{H}}}$ is a norm induced by the inner product.

B. Gaussian Process Regression

GPR is a non-parametric regression algorithm by GP implementation. Consider the model

$$\mathbf{z} = f(\mathbf{x}) + w \tag{2}$$

with input $\mathbf{x} \in \mathbb{R}^n$ and scalar output $\mathbf{z} \in \mathbb{R}$ where $w \in \mathbb{R}$ is zero-mean Gaussian noise with variance σ^2. We are going to approximate function f in (2), given a set of input-output observations. A pair $\mathbf{x}_i, \mathbf{z}_i$ of input-output observation is called training data. A set $D \triangleq \langle \mathbf{X}, \mathbf{Z} \rangle$ of training data is given where $\mathbf{X} = [\mathbf{x}_1, \cdots, \mathbf{x}_N], \mathbf{Z} = [\mathbf{z}_1, \cdots, \mathbf{z}_N]$ and N is the number of the training data pairs. GPR aims to approximate function f in (2) by utilizing the training data set D under the assumption that f is a zero-mean GP (see p.540 in [14] for the GP definition). Under the GP assumption, $[f(\mathbf{x}_1), \cdots, f(\mathbf{x}_N)]^T$ is multivariate Gaussian and we denote its covariance by (kernel matrix) $G(\mathbf{X}, \mathbf{X})$, where (i, i') element of G is denoted by kernel $g(x_i, x_{i'})$. Kernel represents a similarity between the outputs. Please refer to Table 4.1 in [6] for commonly used kernel functions.

According to p.200 in [6], given test input $\mathbf{x}_*$, the Gaussian predictive distribution over test output $\mathbf{z}_*$ has mean

$$\mu(\mathbf{x}_*, D) = g_*^T (G(\mathbf{X}, \mathbf{X}) + \sigma^2 I)^{-1} \mathbf{Z} \tag{3}$$

and variance $\Sigma(\mathbf{x}_*, D) = g(\mathbf{x}_*, \mathbf{x}_*) - g_*^T (G(\mathbf{X}, \mathbf{X}) + \sigma^2 I)^{-1} g_*$ where $g_* = G(\mathbf{X}, \mathbf{x}_*)$. We call them as GPR mean and GPR variance, respectively. If output $\mathbf{z} \in \mathbb{R}^m$ in (2) is multi-dimensional, then GPR is conducted for each output element of $\mathbf{z}$. Let $\mu(\mathbf{x}_*, D(i))$ and $\Sigma(\mathbf{x}_*, D(i))$ denote the GP for the i^{th} element of $\mathbf{z}$ where $D(i) = \langle \mathbf{X}, \mathbf{Z}(i) \rangle$ and $\mathbf{Z}(i) = [\mathbf{z}_1(i), \cdots, \mathbf{z}_N(i)]^T$. Then, we define the function GPR as

$$[\bar{\mu}(\mathbf{x}_*, D), \bar{\Sigma}(\mathbf{x}_*, D)] \triangleq GPR(\mathbf{x}_*, D) \tag{4}$$

where $\bar{\mu}(\mathbf{x}_*, D) = [\mu(\mathbf{x}_*, D(1)), \cdots, \mu(\mathbf{x}_*, D(n))]^T$ and $\bar{\Sigma}(\mathbf{x}_*, D) = \text{diag}(\Sigma(\mathbf{x}_*, D(1)), \cdots, \Sigma(\mathbf{x}_*, D(n)))$ denote the GPR mean of function f and its covariance, respectively.

This paper utilizes Gaussian kernel, where (i, i') element of G is described by $G_{ij} = g(\mathbf{x}_i, \mathbf{x}_j) = \sigma_h^2 e^{-\frac{1}{2}(\mathbf{x}_i - \mathbf{x}_j)^T W (\mathbf{x}_i - \mathbf{x}_j)}$. Diagonal matrix W represents the length scale of each input, and σ_h^2 is a variance. A set of parameters $\theta = [W, \sigma_h, \sigma]$ is called hyper-parameters and they show the user's interpretation of the regression function. They may be chosen by maximizing the log-likelihood of the training output so that the choice of hyper-parameters is optimal: $\theta_{\max} = \text{argmax}_\theta(\log(p(\mathbf{Z}|\mathbf{X}, \theta)))$ as in [6].

IV. Attack-resilient Gaussian Process Regression (ArGPR)

We, in this section, derive a data-driven attack-resilient learning algorithm to address the problem described in Section II. In Section IV-A, we discuss preliminary steps. Then, a description of training data set is given in Section IV-B. Section IV-C presents the solution, the Attack-resilient Gaussian Process Regression (ArGPR) algorithm. The proposed algorithm is derived in detail in Section IV-D.

A. Output decomposition, and system transformation

We discuss two inherent difficulties and tricks to deal with them. First, the defender is unaware of the sensor attack locations. We will first address the problem under the assumption that the locations of vulnerable sensors are known. In Section VI, we discuss how to relax this assumption. Under this assumption, output y_k in (1) is decomposed into

$$y_{1,k} = C_{1,k} x_k + d_{1,k} + v_{1,k}, \quad y_{2,k} = C_{2,k} x_k + v_{2,k} \tag{5}$$

where $y_{1,k}$ is the potentially corrupted sensor output, and $y_{2,k}$ is the sensor output that is free of attacks.

Second, state estimation errors and function approximation errors are dependent. To break the interdependency, we let actuator attack vector estimate compensate the actuator attack and the errors of the function approximation. Then, function approximation errors no longer induce errors in state estimation. In particular, we rewrite system (1) as follows:

$$x_k = f(x_{k-1}, u_{k-1}) + d'_{2,k-1} + w_{k-1} \tag{6}$$

where $d'_{2,k-1} \triangleq f(x_{k-1}, u_{k-1} + d_{a,k-1}) - f(x_{k-1}, u_{k-1})$. Given function approximation $\hat{f}_k(\cdot) = \bar{\mu}(\cdot, \hat{D}_k)$ in (4) and state estimate $\hat{x}_{k-1}$, system model (5) and (6) becomes

$$\begin{aligned} x_k &= \hat{f}_k([x_{k-1}^T, u_{k-1}^T]^T) + d_{2,k-1} + w_{k-1} \\ y_{1,k} &= C_{1,k} x_k + d_{1,k} + v_{1,k}, \quad y_{2,k} = C_{2,k} x_k + v_{2,k} \end{aligned} \tag{7}$$

where $d_{2,k-1} \triangleq d'_{2,k-1} + \tilde{f}_k(x_{k-1}, u_{k-1})$, and $\tilde{f}_k(x_{k-1}, u_{k-1}) \triangleq f(x_{k-1}, u_{k-1}) - \hat{f}_k([x_{k-1}^T, u_{k-1}^T]^T)$ is unknown approximation error. The covariance matrix of w_{k-1} is approximated as $\hat{Q}_{k-1} = \bar{\Sigma}([\hat{x}_{k-1}^T, u_{k-1}^T]^T, \hat{D}_k)$.

Since both actuator attack $d'_{2,k-1}$ and function approximation error $\tilde{f}_k(x_{k-1}, u_{k-1})$ are unknown, the defender is unable to separate them from the sum $d_{2,k-1}$. Therefore, we will estimate transformed attack vector $d_{2,k-1}$ instead of $d'_{2,k-1}$ and $d_{a,k-1}$, and its estimate is denoted by $\hat{d}_{2,k-1}$.

Our estimation algorithm will utilize linearization to track covariance. Linearization of system (7) at the estimates is

$$x_k = A_{k-1}x_{k-1} + B_{k-1}u_{k-1} + d_{2,k-1} + w_{k-1}$$
$$y_{1,k} = C_{1,k}x_k + d_{1,k} + v_{1,k}, \quad y_{2,k} = C_{2,k}x_k + v_{2,k}$$

where $[A_{k-1}^T, B_{k-1}^T]^T = \frac{\partial \hat{f}([\hat{x}_{k-1}^T, u_{k-1}^T]^T)}{\partial [\hat{x}_{k-1}^T, u_{k-1}^T]^T}$.

B. Training data set

To regress function f, it is required to know input-output observations according to Section III-B. Let us define $x_k^+ \triangleq f(x_k, u_k) + w_k$. The desired training data set available at time k is given by $D_k \triangleq \langle \mathbf{X}_k, \mathbf{X}_k^+ \rangle$ where

$$\mathbf{X}_k = \left[\begin{array}{ccc} \mathbf{x}_1 & \cdots & \mathbf{x}_{N(k)} \end{array} \right] \triangleq \left[\begin{array}{ccc} x_0 & \cdots & x_{k-2} \\ u_0 & \cdots & u_{k-2} \end{array} \right],$$

$\mathbf{X}_k^+ = [\mathbf{x}_1^+, \cdots, \mathbf{x}_{N(k)}^+] \triangleq [x_0^+, \cdots, x_{k-2}^+]$ and $N(k)$ is the number of input-output pairs in the training data set. However, unlike Section III-B, x_k and x_k^+ are unavailable. Instead, we use their estimates to perform function regression. The available training data set is $\hat{D}_k \triangleq \langle \hat{\mathbf{X}}_k, \hat{\mathbf{X}}_k^+ \rangle$ where

$$\hat{\mathbf{X}}_k = \left[\begin{array}{ccc} \hat{\mathbf{x}}_1 & \cdots & \hat{\mathbf{x}}_{N(k)} \end{array} \right] \triangleq \left[\begin{array}{ccc} \hat{x}_0 & \cdots & \hat{x}_{k-2} \\ u_0 & \cdots & u_{k-2} \end{array} \right],$$

$\hat{\mathbf{X}}_k^+ = [\hat{\mathbf{x}}_1^+, \cdots, \hat{\mathbf{x}}_{N(k)}^+] \triangleq [\hat{x}_1 - \hat{d}_0, \cdots, \hat{x}_{k-1} - \hat{d}_{k-2}]$. Although training data set $\hat{D}_k$ contains estimation errors, we will derive an algorithm as if $\hat{D}_k$ has no estimation errors (certainty equivalence principle [15]). The errors in the training data set will be considered in the analysis section.

C. Algorithm statement

ArGPR algorithm utilizes Gaussian process regression to approximate unknown dynamic systems, which is then used to estimate the current internal state by unknown input and state estimation technique [13]. In particular, ArGPR algorithm (Algorithm 1) consists of a bank of ArE algorithm 2 (line 2) and a system function estimator (training set updater, line 3). The ArE algorithm can be seen as an extension of the extended Kalman filter with two extensions; first, it incorporates attack vector estimation [13]; second, the unknown system function is replaced by GPR function approximation. The ArE algorithm recursively produces state estimate $\hat{x}_k$, attack vector estimates $\hat{d}_{2,k-1}$, and $\hat{d}_{1,k}$. The following section presents the algorithm derivation in details.

D. Derivation of the ArE algorithm

System learning. Given $\hat{D}_k$, we are able to find approximation $\hat{f}_k(\cdot) = \bar{\mu}(\cdot, \hat{D}_k)$ of system function f and covariance $\hat{Q}_k = \bar{\Sigma}(\cdot, \hat{D}_k)$ via $GPR([x^T, u^T]^T, D_k)$ in (4). They will be used as if they are the ground truth.

Actuator attack $d_{2,k-1}$ estimation. Assuming that there is no actuator attack, we predict the current state (line 2) as $\hat{x}'_{k|k-1} = \hat{f}_k([\hat{x}_{k-1}^T, u_{k-1}^T]^T)$. From this estimation, actuator attack $d_{2,k-1}$ can be estimated by $y_{2,k}^j$ in (5) as $\hat{d}_{2,k-1} = M_k(y_{2,k} - C_{2,k}\hat{x}'_{k|k-1}) \simeq M_k(C_{2,k}A_{k-1}\tilde{x}_{k-1} + C_{2,k}d_{2,k-1} + C_{2,k-1}w_{k-1} + v_{2,k})$ (line 7) where function $\hat{f}_k([\hat{x}_{k-1}^T, u_{k-1}^T]^T)$ is linearized. Assuming that $\mathbb{E}[\tilde{x}_{k-1}] = 0$, we choose gain matrix M_k by the Gauss Markov theorem in [16] as $M_k = (C_{k,2}^T \tilde{R}_{2,k}^{-1} C_{2,k})^{-1} C_{2,k}^T \tilde{R}_{2,k}^{-1}$ (line 6) where $\tilde{R}_{2,k} \triangleq C_{2,k}A_{k-1}P_{k-1}^x A_{k-1}^T C_{2,k}^T + R_{2,k} + C_{2,k}\hat{Q}_{k-1}C_{2,k}^T$. Estimation error of $d_{2,k-1}$ is given by

$$\tilde{d}_{2,k-1} = -M_k(C_{2,k}A_{k-1}\tilde{x}_{k-1} + C_{2,k}w_{k-1} + v_{2,k}) \quad (8)$$

where $M_k C_{2,k} = I$ is used. Error covariance matrix is found by $P_{k-1}^{d_2} = M_k C_{2,k} A_{k-1} P_{k-1}^x (M_k C_{2,k} A_{k-1})^T + M_k C_{2,k} \hat{Q}_{k-1} (M_k C_{2,k})^T + M_k R_{2,k} M_k^T$ (line 8).

Algorithm 1: Attack-resilient Gaussian Process Regression (ArGPR)

```
1: Input: $\hat{x}_{k-1}$, $P_{k-1}^x$, and $\hat{D}_k$;
2: $[\hat{x}_k, \hat{d}_{2,k-1}, \hat{d}_{1,k}, P_k^x, P_{k-1}^{d_2}, P_k^{d_1}] = ArE(\hat{x}_{k-1}, P_{k-1}^x, \hat{D}_k)$;
   ▷ Training data set update
3: $\hat{D}_{k+1} = \hat{D}_k \cup \langle [\hat{x}_{k-1}^T, u_{k-1}^T]^T, \hat{x}_k - \hat{d}_{2,k-1} \rangle$;
4: $\hat{f}_k([x^T, u^T]^T) = \bar{\mu}([x^T, u^T]^T, \hat{D}_k)$;
5: Return: $\hat{x}_k$, $\hat{d}_{2,k-1}$, $\hat{d}_{1,k}$, $P_k^x$, $P_{k-1}^{d_2}$, $P_k^{d_1}$, $\hat{D}_{k+1}$, and $\hat{f}_k([x^T, u^T]^T)$.
```

Algorithm 2: Attack-resilient Estimation (ArE)

```
1: Input: $\hat{x}_{k-1}$, $P_{k-1}^x$, $D_k$;
   ▷ Actuator attack $d_{2,k-1}$ estimation
2: $[\hat{x}'_{k|k-1}, \hat{Q}_{k-1}] = GPR([\hat{x}_{k-1}^T, u_{k-1}^T]^T, D_k)$;
3: $[A_{k-1}^T, B_{k-1}^T]^T = \frac{\partial \bar{\mu}([\hat{x}_{k-1}^T, u_{k-1}^T]^T, D_k)}{\partial [\hat{x}_{k-1}^T, u_{k-1}^T]^T}$;
4: $\hat{y}'_{2,k} = C_{2,k}\hat{x}'_{k|k-1}$;
5: $\tilde{R}_{2,k} = C_{2,k}A_{k-1}P_{k-1}^x A_{k-1}^T C_{2,k}^T + R_2 + C_{2,k}\hat{Q}_{k-1}C_{2,k}^T$;
6: $M_k = (C_{2,k}^T \tilde{R}_{2,k}^{-1} C_{2,k})^{-1} C_{2,k}^T \tilde{R}_{2,k}^{-1}$;
7: $\hat{d}_{2,k-1} = M_k(y_{2,k} - \hat{y}'_{2,k})$;
8: $P_{k-1}^{d_2} = M_k C_{2,k} A_{k-1} P_{k-1}^x (M_k C_{2,k} A_{k-1})^T + M_k R_2 M_k^T + M_k C_{2,k} \hat{Q}_{k-1} (M_k C_{2,k})^T$;
   ▷ State prediction
9: $\hat{x}_{k|k-1} = \hat{f}_k([\hat{x}_{k-1}^T, u_{k-1}^T]^T) + \hat{d}_{2,k-1}$;
10: $\bar{Q}_{k-1} = (I - M_k C_{2,k}) \hat{Q}_{k-1} (I - M_k C_{2,k})^T + M_k R_2 M_k^T$;
11: $\bar{A}_{k-1} = (I - M_k C_{2,k}) A_{k-1}$;
12: $P_{k|k-1}^x = \bar{A}_{k-1} P_{k-1}^x \bar{A}_{k-1}^T + \bar{Q}_{k-1}$;
   ▷ State estimation
13: $L_k = (P_{k|k-1}^x C_{2,k}^T - M_k R_2)\ (C_{2,k} P_{k|k-1}^x C_{2,k}^T + R_2 - C_{2,k} M_k R_2 - R_2 M_k^T C_{2,k}^T)^{-1}$;
14: $\hat{x}_k = \hat{x}_{k|k-1} + L_k(y_{2,k} - C_{2,k}\hat{x}_{k|k-1})$;
15: $P_k^x = (I - L_k C_{2,k}) P_{k|k-1}^x (I - L_k C_{2,k})^T + L_k R_2 L_k^T + (I - L_k C_{2,k}) M_k R_2 L_k^T + L_k R_2 M_k^T (I - L_k C_{2,k})^T$;
   ▷ Sensor attack $d_{1,k}$ estimation
16: $\hat{d}_{1,k} = y_{1,k} - C_{1,k}\hat{x}_k f$;
17: $P_k^{d_1} = C_{1,k} P_k^x C_{1,k}^T + R_1$;
18: Return: $\hat{x}_k$, $\hat{d}_{2,k-1}$, $\hat{d}_{1,k}$, $P_k^x$, $P_{k-1}^{d_2}$, and $P_k^{d_1}$.
```

State prediction. Generate state prediction $\hat{x}_{k|k-1}$ by simulating system (1) as $\hat{x}_{k|k-1} = \hat{f}_k([\hat{x}_{k-1}^T, u_{k-1}^T]^T) + \hat{d}_{2,k-1}$ (line 9). The state estimation error becomes

$$\tilde{x}_{k|k-1} \simeq A_{k-1}\tilde{x}_{k-1} + \tilde{d}_{2,k-1} + w_{k-1} \tag{9}$$

where function $\hat{f}_k$ is linearized. Substitution (8) into (9) leads to (line 12) $P^x_{k|k-1} = \bar{A}_{k-1}P^x_{k-1}\bar{A}^T_{k-1} + \bar{Q}_{k-1}$. where $\bar{Q}_{k-1} \triangleq (I - M_kC_{2,k})\hat{Q}_{k-1}(I - M_kC_{2,k})^T + M_kR_{2,k}M_k^T$, and $\bar{A}_{k-1} \triangleq (I - M_kC_{2,k})A_{k-1}$.

State estimation. Update state prediction $\hat{x}_{k|k-1}$ as (line 14) $\hat{x}_k = \hat{x}_{k|k-1} + L_k(y_{2,k} - C_{2,k}\hat{x}_{k|k-1})$. Substitution $y_{2,k}$ in (5) into this equation leads to $\tilde{x}_k \simeq (I - L_kC_{2,k})\tilde{x}_{k|k-1} - L_kv_{2,k}$. Its error covariance matrix is (line 15)

$$\begin{aligned} P_k^x &= (I - L_kC_{2,k})P^x_{k|k-1}(I - L_kC_{2,k})^T + L_kR_{2,k}L_k^T \\ &+ (I - L_kC_{2,k})M_kR_{2,k}L_k^T + L_kR_{2,k}M_k^T(I - L_kC_{2,k})^T. \end{aligned}$$

Minimizing $\mathrm{tr}(P_k^x)$ with decision variable L_k is an unconstrained optimization problem. We can find the minimizer by taking derivative of $\mathrm{tr}(P_k^x)$ and setting it equal to zero

$$\begin{aligned} \frac{\partial\,\mathrm{tr}(P_k^x)}{\partial L_k} &= 2((C_{2,k}P^x_{k|k-1}C^T_{2,k} - R_{2,k}M_k^TC^T_{2,k} \\ &- C_{2,k}M_kR_{2,k} + R_{2,k})L_k^T + R_{2,k}M_k^T - C_{2,k}P^x_{k|k-1}). \end{aligned}$$

The solution is $L_k = (P^x_{k|k-1}C^T_{2,k} - M_kR_{2,k})(R_{2,k} + C_{2,k}P^x_{k|k-1}C^T_{2,k} - C_{2,k}M_kR_{2,k} - R_{2,k}M_k^TC^T_{2,k})^{-1}$ (line 13).

Sensor attack $d_{1,k}$ estimation. Given $\hat{x}_k$, and the assumption that $\mathbb{E}[\tilde{x}_k] = 0$, sensor attack $d_{1,k}$ can be estimated by $y_{1,k}$ in (5) (line 16): $\hat{d}_{1,k} = y_{1,k} - C_{1,k}\hat{x}_k = C_{1,k}\tilde{x}_k + d_{1,k} + v_{1,k}$. Estimation error is obtained by $\tilde{d}_{1,k} = -(C_{1,k}\tilde{x}_k + v_{1,k})$ with covariance $P_k^{d_1} = C_{1,k}P_k^xC^T_{1,k} + R_{1,k}$ (line 17).

Training data set update (ArGPR). Lastly, we construct a new training data pair and add it to the training data set (line 8) $D_{k+1} = D_k \cup \langle[\hat{x}^T_{k-1}, u^T_{k-1}]^T, \hat{x}_k - \hat{d}_{2,k-1}\rangle$. Then, the algorithm returns estimates of system function and output function from the updated training data set as $\hat{f}_k([x^T, u^T]^T) = \bar{\mu}([x^T, u^T]^T, D_k)$.

V. Average case learning of GPR

This section presents an analysis of function approximation errors in terms of average case learning of GPR. The error $f(x_{k-1}, u_{k-1}) - \hat{f}_k([\hat{x}^T_{k-1}, u^T_{k-1}]^T)$ is the point of interest. This error becomes $\tilde{f}_k(x_{k-1}, u_{k-1})$ if $x_{k-1} = \hat{x}_{k-1}$.

Training data set. Let us define the errors in the training data set as follows: $\Delta\mathbf{x}_i = \mathbf{x}_i - \hat{\mathbf{x}}_i$, $\Delta\mathbf{X}_k = \mathbf{X}_k - \hat{\mathbf{X}}_k$, $\Delta\mathbf{x}_i^+ = \mathbf{x}_i^+ - \hat{\mathbf{x}}_i^+$, and $\Delta\mathbf{X}_k^+ = \mathbf{X}_k^+ - \hat{\mathbf{X}}_k^+$. Using the above notations, training data set $\hat{D}_k$ becomes

$$\begin{aligned} \hat{D}_k &\triangleq \langle\hat{\mathbf{X}}_k, \hat{\mathbf{X}}_k^+\rangle = \langle\hat{\mathbf{X}}_k, \mathbf{f}(\hat{\mathbf{X}}_k) + \mathbf{w}_k - \mathbf{f}(\hat{\mathbf{X}}_k) + \mathbf{f}(\mathbf{X}_k) - \Delta\mathbf{X}_k^+\rangle \\ &= \langle\hat{\mathbf{X}}_k, \hat{\mathbf{Z}}_k - \mathbf{f}(\hat{\mathbf{X}}_k) + \mathbf{f}(\mathbf{X}_k) - \Delta\mathbf{X}_k^+\rangle = \langle\hat{\mathbf{X}}_k, \hat{\mathbf{Z}}_k + \Delta\mathbf{Z}_k\rangle \end{aligned}$$

where $\mathbf{f}(\hat{\mathbf{X}}_k) = [f(\hat{\mathbf{x}}_1), \cdots, f(\hat{\mathbf{x}}_{N(k)})]$ and $\hat{\mathbf{Z}}_k = \mathbf{f}(\hat{\mathbf{X}}_k) + \mathbf{w}_k$. In the analysis, we use $\hat{\mathbf{X}}_k$ and $\hat{\mathbf{Z}}_k$ as the input and output to learn function f as the classic GPR in Section III-B. Correspondingly, we consider $\Delta\mathbf{Z}_k = [\Delta\mathbf{z}_1, \cdots, \Delta\mathbf{z}_{N(k)}] = -\mathbf{f}(\hat{\mathbf{X}}_k) + \mathbf{f}(\mathbf{X}_k) - \Delta\mathbf{X}_k^+$ be the output errors.

Analysis in RKHS. Most widely used kernels, including Gaussian kernel, satisfy the following assumption.

Assumption 5.1: Kernel g is continuous symmetric and positive definite. Kernel is time-invariant.

Hyper-parameter θ is chosen time-invariant to satisfy Assumption 5.1. Under Assumption 5.1, there exists a unique Reproducing Kernel Hilbert Space $\mathcal{H}$ (RKHS) by the Moore-Aronszajn theorem (Theorem 6.1 in [6]).

Now consider the minimization of functional

$$J_k[\hat{f}] = \frac{1}{2}\|\hat{f}\|^2_{\mathcal{H}} + \frac{1}{2\sigma^2}\sum_{i=1}^{N(k)}(\hat{\mathbf{x}}_i^+ - \hat{f}(\hat{\mathbf{x}}_i))^2 \tag{10}$$

where σ^2 is the variance of w in (2). The second term works for data fitting and the first term smooths the solution, called regularizer. According to Section 6.2.2 in [6], the minimizer of functional (10) is the GPR mean function (3). In particular, the minimizer of the above functional is in the form of

$$\hat{f}(\mathbf{x}_*) = \sum_{i=1}^{N(k)}\alpha_i g(\mathbf{x}_*, \hat{\mathbf{x}}_i) \tag{11}$$

by the representer theorem [17]. By (11), functional (10) becomes

$$J_k[\alpha] = \frac{1}{2}\alpha^T G(\hat{\mathbf{X}}_k, \hat{\mathbf{X}}_k)\alpha + \frac{1}{2\sigma^2}\|\hat{\mathbf{X}}_k^+ - G(\hat{\mathbf{X}}_k, \hat{\mathbf{X}}_k)\alpha\|^2$$

where $\alpha = [\alpha_1, \cdots, \alpha_{N(k)}]^T$. By taking its derivative with respect to vector α and setting it equal to zero, we can obtain the solution $\alpha = (G(\hat{\mathbf{X}}_k, \hat{\mathbf{X}}_k) + \sigma^2 I)^{-1}\hat{\mathbf{X}}_k^+$. The complete solution $\hat{f}(\mathbf{x}_*) = g_*^T(\mathbf{x}_*)(G(\hat{\mathbf{X}}_k, \hat{\mathbf{X}}_k) + \sigma^2 I)^{-1}\hat{\mathbf{X}}_k^+$ is identical to the GPR mean function in (4). Motived by this property, let us define $\tilde{f}_{k|J_k}(x_{k-1}, u_{k-1}, \hat{x}_{k-1}) \triangleq f(x_{k-1}, u_{k-1}) - \hat{f}_{k|J_k}(\hat{x}_{k-1})$ where $\hat{f}_{k|J_k} = \mathrm{argmin}_{\hat{f}}\, J_k[\hat{f}]$. Note that $\tilde{f}_{k|J_k} = \tilde{f}_k$ and $\hat{f}_{k|J_k} = \hat{f}_k$, provided that $\hat{D}_k$ is known and $\hat{x}_{k-1} = x_{k-1}$ holds. We will analyze average case GPR learning $\hat{f}_{k|\mathbb{E}[J_k]}$ under the following assumptions.

Assumption 5.2: The function f is in RKHS $\mathcal{H}$.

Since $g \in \mathcal{H}$, any linear combination of g is in RKHS $\mathcal{H}$. Thus, function f is in RKHS $\mathcal{H}$ if and only if there exist a set of $\mathbf{x}_i \in \mathbb{R}^{n+a}$, and $\beta_i \in \mathbb{R}$ such that $f(\mathbf{x}) = \sum_{i=1}^{\infty}\beta_i g(\mathbf{x}_i, \mathbf{x})$. Comparing this equation with (11), Assumption 5.2 implies that the chosen kernel can perform sufficiently well to approximate the regression function.

Assumption 5.3: Input $\hat{\mathbf{X}}_k$ and output $\hat{\mathbf{Z}}_k$ in training data are sampled from probability distributions with corresponding probability measure $\mu(\hat{\mathbf{x}}, \hat{\mathbf{z}})$. State estimation errors $\Delta\mathbf{X}_k$, $\Delta\mathbf{X}_k^+$ are independent of $\mathbf{X}_k$ and $\hat{\mathbf{Z}}_k$, respectively.

Under Assumption 5.1, according to Mercer's theorem (Theorem 4.2 in [6]), there is a set of orthonormal eigenfunctions $\{\phi_j\}$ and nonnegative eigenvalues $\{\lambda_j\}$ corresponding to the kernel such that $g(\mathbf{x}, \mathbf{x}') = \sum_{j=1}^{\infty}\lambda_j\phi_j(\mathbf{x})\phi_j(\mathbf{x}')$ where $\sum_{j=1}^{\infty}\lambda_j < \infty$. Under Assumption 5.2, there exists a set of constants $c_j \in \mathbb{R}$ such that $f(\mathbf{x}_*) = \sum_{j=1}^{\infty}c_j\phi_j(\mathbf{x}_*)$. There are infinitely many set of orthonomal eigenfunctions. Of them, we choose one such that Assumption 5.4 holds; e.g., $\phi_j(\mathbf{x}) = e^{\sqrt{-1}^2 j\mathbf{x}}$ (Fourier transform).

Assumption 5.4: Eigenfunctions satisfy $\phi_j(\mathbf{x} + \Delta\mathbf{x}) = \phi_j(\mathbf{x})\phi_j(\Delta\mathbf{x})$.

Theorem 5.1: Under Assumptions 5.1, 5.2, 5.3 and 5.4, it holds that

$\hat{f}_{k|\mathbb{E}[J_k]}(\mathbf{x}_*) = \sum_{j=1}^{\infty} \big[\frac{\lambda_j}{\lambda_j+\sigma^2/N(k)} \frac{1}{N(k)} \sum_{i=1}^{N(k)} (c_j\phi_j(-\Delta\mathbf{x}_i) + \Delta\mathbf{x}_i \int \phi_j(\hat{\mathbf{x}})d\mu(\hat{\mathbf{x}}))\phi_j(\mathbf{x}_*)\big]$.

Interpretation of Theorem 5.1. According to Theorem 5.1, identification error is described by

$$\begin{aligned}
&\tilde{f}_{k|\mathbb{E}[J_k]}(x_{k-1}, u_{k-1}, \hat{x}_{k-1}) = f(\mathbf{x}_{k-1}) - \hat{f}_{k|\mathbb{E}[J_k]}(\hat{\mathbf{x}}_{k-1}) \\
&= f(\mathbf{x}_{k-1}) - \hat{f}_{k|\mathbb{E}[J_k]}(\mathbf{x}_{k-1}) + \hat{f}_{k|\mathbb{E}[J_k]}(\mathbf{x}_{k-1}) - \hat{f}_{k|\mathbb{E}[J_k]}(\hat{\mathbf{x}}_{k-1}) \\
&= \sum_{j=1}^{\infty} (c_j - \frac{\lambda_j}{\lambda_j + \sigma^2/N(k)} c_j)\phi_j(\mathbf{x}_{k-1}) \\
&+ \sum_{j=1}^{\infty} \frac{\lambda_j}{\lambda_j + \sigma^2/N(k)} (c_j - c_j \frac{1}{N(k)} \sum_{i=1}^{N(k)} \phi_j(\Delta\mathbf{x}_i))\phi_j(\mathbf{x}_{k-1}) \\
&- \sum_{j=1}^{\infty} \frac{\lambda_j}{\lambda_j + \sigma^2/N(k)} \int \frac{\phi_j(\hat{\mathbf{x}})d\mu(\hat{\mathbf{x}})}{N(k)} \sum_{i=1}^{N(k)} \Delta\mathbf{x}_i^+ \phi_j(\hat{\mathbf{x}}_{k-1}) \\
&+ \sum_{j=1}^{\infty} \frac{\lambda_j}{\lambda_j + \sigma^2/N(k)} \frac{1}{N(k)} \sum_{i=1}^{N(k)} (c_j\phi_j(\Delta\mathbf{x}_i) \\
&- \Delta\mathbf{x}_i^+ \int \phi_j(\hat{\mathbf{x}})d\mu(\hat{\mathbf{x}}))(\phi_j(\mathbf{x}_{k-1}) - \phi_j(\hat{\mathbf{x}}_{k-1})).
\end{aligned}$$

The first term shows the function identification error given $\Delta\mathbf{X}_k = 0$ and $\Delta\mathbf{X}_k^+ = 0$. This error decreases as $N(k) \to \infty$, where $\lim_{k\to\infty} N(k) = \lim_{k\to\infty} k + N(0) = \infty$. The second term is the error induced by $\Delta\mathbf{X}_k$. Since $\phi_i(0) = 1$ under Assumption 5.4, this error is zero if $\Delta\mathbf{X}_k = 0$. If the total approximation error induced by $\Delta\mathbf{X}_k$ grows slower than a linear rate; i.e., there exist $\alpha > 0$, $0 < \beta < 1$, and k_τ such that $\|N(k) - \sum_{i=1}^{N(k)} \phi_j(\Delta\mathbf{x}_i)\| \le \alpha(N(k))^\beta$ for $\forall k \ge k_\tau$, then $\lim_{N(k)\to\infty} c_j - c_j \frac{1}{N(k)} \sum_{i=1}^{N(k)} \phi_j(\Delta\mathbf{x}_i) = 0$. Similarly, the third term is the error induced by $\Delta\mathbf{X}^+$, and vanishes if $\Delta\mathbf{x}_i^+ = 0$. Also, $\lim_{N(k)\to\infty} \frac{1}{N(k)} \sum_{i=1}^{N(k)} \Delta\mathbf{x}_i^+ = 0$ if there is the finite number of indices such that $\Delta\mathbf{x}_i^+ \neq 0$. The last term is the error induced by the current input. This term decreases as the estimation error $\|\hat{\mathbf{x}}_{k-1} - \mathbf{x}_{k-1}\|$ decreases.

VI. Discussions: Unknown compromised sensors

If the defender is unaware of vulnerable sensors, the defender needs to consider all possible combinations of sensor attack locations. Let $\mathcal{J}$ denote the set of hypothetical modes, and each mode $j \in \mathcal{J}$ assumes that a particular subset of sensors may be corrupted by sensor attacks, and the others are free of sensor attacks. Due to different assumptions on the attack locations, each mode $j \in \mathcal{J}$ has different output $y_{1,k}^j$ and $y_{2,k}^j$ in (5). We conduct ArE algorithm (Algorithm 2) for each mode $j \in \mathcal{J}$ and find a prior probability and a posterior probability (see below). Then, the training data set update in ArGPR algorithm (Algorithm 1) is conducted using the values associated with the most likely mode $\hat{j}_k$.

We quantify the difference between the predicted output and the measured output as follows $\nu_k^j = y_{2,k}^j - C_{2,k}^j \hat{x}_{k|k-1}^j$. The output error ν_k^j is a multivariate Gaussian random variable. Therefore, the likelihood function is given by $\mathcal{N}_k^j \triangleq p(y_k|j = \text{true}) = \mathcal{N}(\nu_k^j; 0, \bar{P}_{k|k-1}^j) = \frac{\exp(-(\nu_k^j)^T (\bar{P}_{k|k-1}^j)^{-1} \nu_k^j/2)}{(2\pi)^{dim(y_{2,k}^j)/2} |\bar{P}_{k|k-1}^j|^{\frac{1}{2}}}$ where $\bar{P}_{k|k-1}^j = C_{2,k}^j P_{k|k-1}^j (C_{2,k}^j)^T + R_{2,k}^j$ is the error covariance matrix of ν_k^j. By the Bayes' theorem, the posterior probability is $\rho_k^j = \frac{\bar{\rho}_k^j}{\sum_{i=1}^{|\mathcal{J}|} \bar{\rho}_k^i}$, where $\bar{\rho}_k^j = \max\{\eta_k^j \rho_{k-1}^j, \epsilon\}$ and $0 < \epsilon < \frac{1}{|\mathcal{J}|}$ is a pre-selected small constant preventing the vanishment of the mode probability. The most likely mode is chosen as the current mode $\hat{j}_k = \text{argmax}_j(\rho_k^j)$.

TABLE I: System variables and parameters

System variables			
f	angular frequency	θ	phase angle
P_M	mechanical power	P_{ij}	power flow
P_C	controllable load	P_L	net load
P_{elec}	electrical power output		
System parameters			
D	damping constant	m	angular momentum
t_{ij}	tie-line stiffness		

VII. Numerical simulation

We present the simulations on the IEEE 68-bus test system (Figure in [18]) where the results of the ArGPR algorithm are compared with those of the GP-EKF algorithm in [12].

System model. We consider a power network represented by an undirected graph $(\mathcal{V}, \mathcal{E})$ where $\mathcal{V} \triangleq \{1, \cdots, 68\}$ and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ are the set of buses and the set of transmission lines, respectively. Let us denote $\mathcal{S}_i \triangleq \{l \in \mathcal{V}\backslash\{i\} | (i, l) \in \mathcal{E}\}$ the set of neighboring buses of $i \in \mathcal{V}$. Each bus is either a generator bus $i \in \mathcal{G}$ or a load bus $i \in \mathcal{L}$ and $\mathcal{V} = \mathcal{G} \cup \mathcal{L}$. The dynamic system of a generator bus $i \in \mathcal{G}$ with attacks is described as the following nonlinear system [19]:

$$\begin{aligned}
\Delta\dot{\theta}_i(t) &= \Delta f_i(t) + w_{1,i}(t) \\
\Delta\dot{f}_i(t) &= -\frac{1}{m_i}\big(D_i\Delta f_i(t) + \sum_{j\in\mathcal{S}_i} \Delta P_{ij}(t) - \Delta P_{M_i}(t) \\
&\quad + d_{a,i}(t) + \Delta P_{L_i}(t)\big) + w_{2,i}(t) \\
y_{i,k} &= [\Delta\theta_{i,k}, \Delta f_{i,k}, \Delta P_{elec_i,k}]^T + [0, d_{s,i,k}^T]^T + v_{i,k} \quad (12)
\end{aligned}$$

where $\Delta P_{ij}(t) = t_{ij}\sin(\Delta\theta_i(t) - \Delta\theta_j(t))$ and $\Delta P_{elec_i,k} = \Delta P_{L_i}(t) + D_i\Delta f_i(t)$. Table I summarizes the system variables and parameters and Δ denotes the distance from nominal value. Vectors $d_{a,i}(t) \in \mathbb{R}$ and $d_{s,i,k} \in \mathbb{R}^2$ denote actuator attack and sensor attack, respectively. The dynamic system for $i \in \mathcal{L}$ has the same dynamic model except $\Delta P_{M_i}(t) = -\Delta P_{C_i}(t)$. We assume that power demand $\Delta P_{L_i}(t)$ is known because it can be predicted by load forecasting methods [20]. Mechanical power $\Delta P_{M_i}(t)$ and controllable load $\Delta P_{C_i}(t)$ are considered known inputs of each bus, and we implement backstepping based stabilizing distributed controllers [21] for frequency control. Subscript k of a continuous-time variable stands for k^{th} discrete time value; e.g., $d_{a,i,k} = d_{a,i}(t_k)$.

Simulation settings. Noises $w_i(t)$ and $v_{i,k}$ are zero-mean Gaussian with covariance $Q_{i,k} = 0.01^2 I$, and $R_{i,k} = 0.01^2 I$. Sampling period is $\epsilon = 0.1s$. The system parameters are adopted from page 598 in [22], where $D_i = 1$, $t_{ij} = 1.5$, and $m_i = 10$ for $\forall i \in \mathcal{V}$. The systems (12) for $\forall i$ are subject

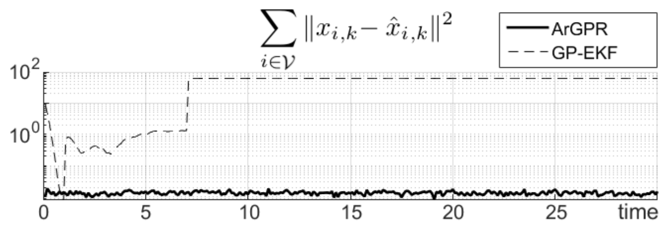

Fig. 1: State estimation errors $\sum_{i\in\mathcal{V}} \|x_{i,k} - \hat{x}_{i,k}\|^2$ in log-scale, where $x_{i,k} = [\Delta\theta_{i,k}, \Delta f_{i,k}]^T$.

to both actuator attacks $d_{a,i}(t) = 30\sin(\frac{i\cdot t_k}{10\pi}) + \frac{i}{100}$ for $t > 1$ and sensor attacks $d_{s,i,k} = [2+0.3\sin(t_k - i), 0]^T$ for $t > 7$.

Distributed implementation. We implement the ArGPR algorithm in a distributed way. Each bus is associated with a local defender. At time k, each local defender i measures $y_{i,k}$ and receives $\Delta\hat{\theta}_{j,k-1}$ from $j \in \mathcal{S}_i$. For the ArGPR algorithm, mechanical power $\Delta P_{M_i}(t)$ (controllable load $\Delta P_{C_i}(t)$, resp.) as well as the estimate of neighboring states $\hat{\theta}_{j,k}$ are treated as inputs for $i \in \mathcal{G}$ ($i \in \mathcal{L}$); i.e., $u_{i,k} = [\Delta P_{M_i,k}, \{\Delta\hat{\theta}_{j,k}\}_{j\in\mathcal{S}_i}]^T$ ($u_{i,k} = [\Delta P_{C_i,k}, \{\Delta\hat{\theta}_{j,k}\}_{j\in\mathcal{S}_i}]^T$). Each local defender is unaware of system (12) but is aware of the true hypothetical mode. We use 1 randomly chosen training pair for initialization. The GP-EKF algorithm in [12] is implemented with the same settings for comparison.

Simulation results. Figures 1 and 2 summarize the simulation results. In Figure 1, the bold line represents the ArGPR algorithm, the dashed line represents the GP-EKF algorithm. It shows that state estimation errors of the ArGPR algorithm diminish, while those of the GP-EKF algorithm remain large. On the other hand, the GP-EKF algorithm fails to estimate states attack-resiliently. Figure 2 presents attack vector estimation errors where $d'_{2,i,k} = [0, -d_{a,i,k}/m_i]^T = [0, -3\sin(\frac{i\cdot t}{10\pi}) - \frac{i}{1000}]^T$, and function approximation errors $\tilde{f}_{i,k}$. The both errors remain low in ArGPR, but function approximation errors are subject to attacks in the GP-EKF.

VIII. Conclusion

We study attack-resilient GPR of unknown nonlinear dynamic systems against both sensor attacks and actuator attacks. We propose a new learning algorithm by incorporating our recently developed unknown input and state estimation technique into the Gaussian process regression algorithm. We empirically demonstrate that the proposed algorithm estimates internal state attack-resiliently, outperforming the GP-EKF algorithm. Unlike existing attack detectors, the proposed algorithm does not require prior system models.

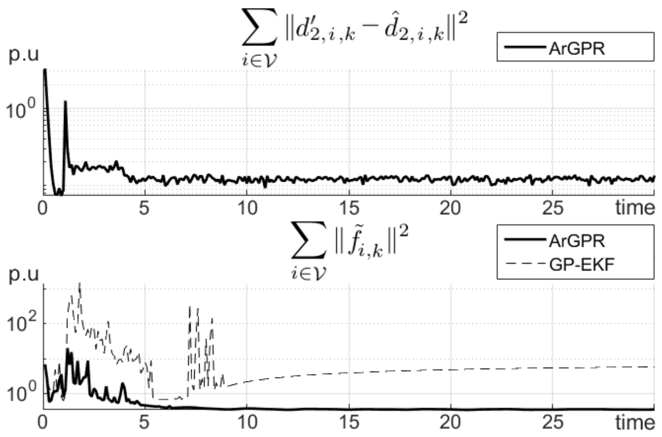

Fig. 2: Actuator attack vector estimation errors in log-scale $\sum_{i\in\mathcal{V}} \|d'_{2,i,k} - \hat{d}_{2,i,k}\|^2$; function approximation errors in log-scale $\sum_{i\in\mathcal{V}} \|\tilde{f}_{i,k}\|^2$.

References

[1] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[2] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, pages 506–519. ACM, 2017.

[3] D. Angluin and P. Laird. Learning from noisy examples. *Machine Learning*, 2(4):343–370, 1988.

[4] S. A. Goldman and R. H. Sloan. Can PAC learning algorithms tolerate random attribute noise? *Algorithmica*, 14(1):70–84, 1995.

[5] M. Kearns and M. Li. Learning in the presence of malicious errors. *SIAM Journal on Computing*, 22(4):807–837, 1993.

[6] C. E. Rasmussen and C. K. I. Williams. *Gaussian processes for machine learning*, volume 1. MIT press Cambridge, 2006.

[7] J. Wang, A. Hertzmann, and D. M. Blei. Gaussian process dynamical models. In *Advances in neural information processing systems*, pages 1441–1448, 2006.

[8] M. A. Osborne, R. Garnett, K. Swersky, and N. De Freitas. Prediction and fault detection of environmental signals with uncharacterised faults. In *AAAI*, 2012.

[9] R. Murray-Smith and D. Sbarbaro. Nonlinear adaptive control using nonparametric Gaussian process prior models. *IFAC Proceedings Volumes*, 35(1):325–330, 2002.

[10] B. Safarinejadian and E. Kowsari. Fault detection in non-linear systems based on GP-EKF and GP-UKF algorithms. *Systems Science & Control Engineering: An Open Access Journal*, 2(1):610–620, 2014.

[11] J. Ko, D. J. Kleint, D. Fox, and D. Haehnelt. GP-UKF: Unscented Kalman filters with Gaussian process prediction and observation models. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pages 1901–1907. IEEE, 2007.

[12] E. Kowsari, B. Safarinejadian, and J. Zarei. Non-parametric fault detection methods in non-linear systems. *IET Science, Measurement & Technology*, 10(3):167–176, 2016.

[13] H. Kim, P. Guo, M. Zhu, and P. Liu. On attack-resilient estimation of switched nonlinear cyber-physical systems. In *2017 American Control Conference*, pages 4328–4333, 2017.

[14] D. J. C. MacKay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

[15] P. Whittle. The risk-sensitive certainty equivalence principle. *Journal of Applied Probability*, pages 383–388, 1986.

[16] T. Kailath, A. H. Sayed, and B. Hassibi. *Linear estimation*, volume 1. Prentice Hall Upper Saddle River, NJ, 2000.

[17] B. Schölkopf, R. Herbrich, and A. J. Smola. A generalized representer theorem. In *International Conference on Computational Learning Theory*, pages 416–426. Springer, 2001.

[18] X. Zhang, C. Rehtanz, and B. Pal. *Flexible AC transmission systems: Modelling and control*. Springer Science & Business Media, 2012.

[19] A. J. Wood and B. F. Wollenberg. *Power Generation Operation and Control*. New York: Wiley, 1996.

[20] H. S. Hippert, C. E. Pedreira, and R. C. Souza. Neural networks for short-term load forecasting: A review and evaluation. *IEEE Transactions on power systems*, 16(1):44–55, 2001.

[21] H. Kim, M. Zhu, and J. Lian. Distributed robust adaptive frequency control of power systems with dynamic loads. *IEEE Transactions on Automatic Control*, 2019. To appear.

[22] P. Kundur, N. J. Balu, and M. G. Lauby. *Power system stability and control*. McGraw-Hill, 1994.