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

\[ x_k = f(x_{k-1}, u_{k-1} + d_{a,k-1}) + w_{k-1} \]
\[ y_k = C_k x_k + d_{s,k} + v_k \]

(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
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 notions. 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, $\hat{a}_k \triangleq a_k - \hat{a}_k$ 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 $\mu$ is mean and $\Sigma$ is covariance.

**Definition 3.1:** (Definition 6.1 in [6]) Hilbert space $\mathcal{H}$ of real functions $f$ defined on $\mathcal{X}$ is 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

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

with input $x \in \mathbb{R}^n$ and scalar output $z \in \mathbb{R}$ where $w \in \mathbb{R}$ is zero-mean Gaussian noise with variance $\sigma^2$. We are going to approximate function $f$ in (2), given a set of input-output observations. A pair $x_i, z_i$ of input-output observation is called training data. A set $D \triangleq \{X, Z\}$ of training data is given where $X = [x_1, \cdots, x_N], \ Z = [z_1, \cdots, z_N]$ 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, $[f(x_1), \cdots, f(x_N)]^T$ is multivariate Gaussian and we denote its covariance by (kernel matrix) $G(X, X)$, where $(i, i')$ element of $G$ is denoted by kernel $g(x_i, x_{i'})$.

Kernel represents a similarity between the inputs. Please refer to Table 4.1 in [6] for commonly used kernel functions.

According to p.200 in [6], given test input $x_i$, the Gaussian predictive distribution over test output $z_i$ has mean

$$\mu(x_i, D) = g_i^T (G(X, X) + \sigma^2 \mathbb{I})^{-1} z$$

and variance $\Sigma(x_i, D) = g_i g_i^T (G(X, X) + \sigma^2 \mathbb{I})^{-1} g_i$, where $g_i = G(X, x_i)$. We call them as GPR mean and GPR variance, respectively. If output $z \in \mathbb{R}^m$ in (2) is multi-dimensional, then GPR is conducted for each output element of $z$. Let $\mu(x_i, D)$ and $\Sigma(x_i, D)$ denote the GP for the $i^{th}$ element of $z$ where $D(i) = (X, Z(i))$ and $Z(i) = [z_1(i), \cdots, z_N(i)]^T$. Then, we define the function GPR as

$$\mu(x, D), \Sigma(x, D) \triangleq \text{GPR}(x, D) \tag{4}$$

where $\mu(x, D) = [\mu(x, D(1)), \cdots, \mu(x, D(n))]^T$ and $\Sigma(x, D) = \text{diag}(\Sigma(x, D(1)), \cdots, \Sigma(x, D(n)))$ 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 $G_{ii} = g(x_i, x_{i'}) = \sigma_f^2 e^{-\frac{1}{2}(x_i-x_{i'})^T W (x_i-x_{i'})}$. Diagonal matrix $W$ represents the length scale of each input, and $\sigma_f^2$ is a variance. A set of parameters $\theta = \{W, \sigma_f, \sigma\}$ 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: $\theta_{\text{max}} = \operatorname{arg\,max}_{\theta} (\log(p(Z|X, \theta)))$ as in [6].

### IV. Attack-resistant 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-resistant 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 $y_{ik}$ 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 $\hat{f}_k(\cdot) = \hat{\mu}_k(\cdot, D_k)$ in (4) and state estimate $\hat{x}_{k-1}$, system model (5) and (6) becomes

$$x_k = \hat{f}_k([x_{k-1}^T, u_{k-1}^T]^T) + d_{2,k-1} + w_{k-1} \tag{7}$$

$$y_{1,k} = C_{1,k} x_k + d_{1,k} + v_{1,k}, \quad y_{2,k} = C_{2,k} x_k + v_{2,k} \tag{7}$$

where $d_{2,k-1} \triangleq d_{2,k-1} + \hat{f}_k((x_{k-1} - u_{k-1}), u_{k-1})$, and $\hat{f}_k((x_{k-1} - u_{k-1}), u_{k-1}) \triangleq f(x_{k-1}, u_{k-1}) - \hat{f}_k([x_{k-1}^T, u_{k-1}^T]^T)$ is unknown approximation error. The covariance matrix of $d_{2,k-1}$ is approximated as $\hat{Q}_{k-1} = \Sigma([x_{k-1}^T, u_{k-1}^T]^T, D_k)$.
Since both actuator attack $d_{2,k-1}$ and function approximation error $f_k(x_{k-1}, u_{k-1})$ 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

$$
x_k = A_k x_{k-1} + B_{k-1} u_{k-1} + d_{2,k-1} + w_{k-1}
$$

$$
y_{1,k} = C_{1,k} x_k + d_1 k + v_{1,k}, 
\hat{y}_{2,k} = C_2,k x_k + v_{2,k}
$$

where $[A^T_{k-1}, B^T_{k-1}] = \frac{\partial f([x_{k-1}, u_{k-1}]^T)}{\partial[x_{k-1}, u_{k-1}]^T}$.

**B. Training data set**

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

$$
X_k = \left[ \begin{array}{c} x_1 \cdots x_N(k) \end{array} \right] \triangleq \left[ \begin{array}{c} x_0 \cdots x_{k-2} \\ u_0 \cdots u_{k-2} \end{array} \right],
$$

$X_k^+ = \left[ \begin{array}{c} x_1^+ \cdots x_N^+(k) \end{array} \right] \triangleq \left[ \begin{array}{c} x_1^+ \cdots x_{k-2}^+ \end{array} \right] \text{ and } N(k) \text{ is the number of input-output pairs in the training data set. However, unlike Section III-B, } x_k \text{ and } x_k^- \text{ are unavailable. Instead, we use their estimates to perform function regression. The available training data set is } D_k \triangleq \langle \hat{X}_k, \hat{X}_k^- \rangle \text{ where}

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

$\hat{X}_k^+ = \left[ \begin{array}{c} \hat{x}_1^+ \cdots \hat{x}_N^+(k) \end{array} \right] \triangleq \left[ \begin{array}{c} \hat{x}_1^+ \cdots \hat{x}_{k-2}^+ \end{array} \right] \text{. Although training data set } \hat{D}_k \text{ contains estimation errors, we will derive an algorithm as if } \hat{D}_k \text{ 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 updatr, line 3). The ArE algorithm can be seen as an extension of the updated 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 $\hat{f}_k(\cdot) = \hat{\mu}(\cdot, \hat{D}_k)$ of system function $f$ and covariance $\hat{Q}_k = \hat{\Sigma}(\cdot, \hat{D}_k)$ via $\text{GPR}([x^T_k, u^T_k]^T, \hat{D}_k)$ 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 $\hat{x}^\prime_{k|k-1} = \hat{f}_k([x_{k-1}^T, u_{k-1}^T]^T)$.

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

<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Input: $\hat{x}<em>{k-1}, P^x</em>{k-1}$, and $\hat{D}_k$;</td>
</tr>
<tr>
<td>2</td>
<td>$\hat{x}<em>k, \hat{d}</em>{2,k-1}, d_{1,k}, F_k, P^d_{k-1}, P^d_{k}$ = $\text{ArE}(\hat{x}<em>{k-1}, P^x</em>{k-1}, \hat{D}_k)$;</td>
</tr>
<tr>
<td>3</td>
<td>$\text{Training data set update}$</td>
</tr>
<tr>
<td>4</td>
<td>$\hat{D}<em>{k+1} = \hat{D}<em>k \cup ([\hat{x}</em>{k-1}^T, u</em>{k-1}^T]^T), \hat{x}<em>k - \hat{d}</em>{2,k-1}$;</td>
</tr>
<tr>
<td>5</td>
<td>$\hat{f}_k([x^T, u^T]^T) = \hat{\mu}([x^T, u^T]^T, \hat{D}_k)$;</td>
</tr>
<tr>
<td>6</td>
<td>Return: $\hat{x}<em>k, \hat{d}</em>{2,k-1}, \hat{d}<em>{1,k}, P^x_k, P^d</em>{k-1}, P^d_k, \hat{D}_{k+1}$, and $\hat{f}_k([x^T, u^T]^T)$;</td>
</tr>
</tbody>
</table>

**Algorithm 2: Attack-resilient Estimation (ArE)**

<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Input: $\hat{x}<em>{k-1}, P^x</em>{k-1}, D_k$;</td>
</tr>
<tr>
<td>2</td>
<td>$\text{Actuator attack } d_{2,k-1}$ estimation</td>
</tr>
<tr>
<td>3</td>
<td>$\hat{x}<em>{\text{est}}</em>{k-1}, \hat{Q}<em>{k-1} = \text{GPR}([\hat{x}</em>{k-1}^T, u_{k-1}^T]^T, D_k)$;</td>
</tr>
<tr>
<td>4</td>
<td>$[A^T_{k-1}, B^T_{k-1}] = \frac{\partial \hat{f}([\hat{x}<em>{k-1}^T, u</em>{k-1}^T]^T)}{\partial[\hat{x}<em>{k-1}^T, u</em>{k-1}^T]^T}$;</td>
</tr>
<tr>
<td>5</td>
<td>$\hat{y}<em>{2,k}^+ = C_2,k \hat{x}</em>{\text{est}}_{k-1}$;</td>
</tr>
<tr>
<td>6</td>
<td>$\hat{R}<em>{2,k} = C_2,k \hat{A}</em>{k-1} P^d_{k-1} A_{k-1}^T C_2,k + R_2 + C_2,k \hat{Q}_{k-1} C_2,k^T$;</td>
</tr>
<tr>
<td>7</td>
<td>$M_k = (C_2,k R_2^{-1} C_2,k)^{-1} C_2,k R_2^{-1}$;</td>
</tr>
<tr>
<td>8</td>
<td>$\hat{P}^d_{k-1} = M_k (y_{2,k} - \hat{y}_{2,k}^+)$;</td>
</tr>
<tr>
<td>9</td>
<td>$\text{State prediction}$</td>
</tr>
<tr>
<td>10</td>
<td>$\hat{x}_{k</td>
</tr>
<tr>
<td>11</td>
<td>$\hat{Q}_{k</td>
</tr>
<tr>
<td>12</td>
<td>$\hat{A}<em>{k-1} = (I - M_k C_2,k) A</em>{k-1}$;</td>
</tr>
<tr>
<td>13</td>
<td>$\hat{P}^d_{k</td>
</tr>
<tr>
<td>14</td>
<td>$\text{State estimation}$</td>
</tr>
<tr>
<td>15</td>
<td>$L_k = (\hat{P}^d_{k</td>
</tr>
<tr>
<td>16</td>
<td>$(C_2,k\hat{P}^d_{k</td>
</tr>
<tr>
<td>17</td>
<td>$\hat{x}<em>{k} = \hat{x}</em>{k</td>
</tr>
<tr>
<td>18</td>
<td>$\hat{P}^d_{k} = (I - L_k C_2,k) F_k_{k</td>
</tr>
<tr>
<td>19</td>
<td>$\text{Sensor attack } d_{1,k}$ estimation</td>
</tr>
<tr>
<td>20</td>
<td>$\hat{d}<em>{1,k} = y</em>{1,k} - C_1,k \hat{x}_{k}$;</td>
</tr>
<tr>
<td>21</td>
<td>$P^d_{k} = C_1,k P^d_{k} C_1,k^T + R_1$;</td>
</tr>
<tr>
<td>22</td>
<td>$\text{Return: } \hat{x}<em>k, \hat{d}</em>{2,k-1}, \hat{d}<em>{1,k}, P^x_k, P^d</em>{k-1}, P^d_k$;</td>
</tr>
</tbody>
</table>
State prediction. Generate state prediction $\hat{x}_{k|k-1}$ by simulating system (1) as $\hat{x}_{k|k-1} = \hat{f}_k([\hat{x}_{k-1}^T, u_{k-1}^T]^T) + \hat{d}_{k-1}$ (line 9). The state estimation error becomes

$$\hat{x}_{k|k-1} \simeq A_{k-1}\hat{x}_{k-1} + \hat{d}_{k-1} + w_{k-1}$$

(9)

where function $\hat{f}_k$ is linearized. Substitution (8) into (9) leads to (line 12)

$$P_{k|k-1} = A_{k-1}P_{k-1}A_{k-1}^T + Q_{k-1} \text{ where } Q_{k-1} \triangleq (I - M_k C_{k,2}) Q_{k-1} (I - M_k C_{k,2})^T + M_k R_k M_k^T,$n

and $A_{k-1} \triangleq (I - M_k C_{k,2}) A_{k-2}$. State estimation. Update state prediction $\hat{x}_{k|k-1}$ as (line 14)

$$\hat{x}_k = \hat{x}_{k|k-1} + L_k(y_{k,2} - C_k \hat{x}_{k|k-1}).$$

Substitution $y_{k,2}$ in (5) into this equation leads to $\hat{x}_k \simeq (I - L_k C_{k,2}) \hat{x}_{k|k-1} - L_k y_{2,k}$. Its error covariance matrix is (line 15)

$$P_k = (I - L_k C_{k,2}) P_{k|k-1}(I - L_k C_{k,2})^T + L_k R_k L_k^T$$

Minimizing $tr(P_k^2)$ with decision variable $L_k$ is an unconstrained optimization problem. We can find the minimizer by taking derivative of $tr(P_k^2)$ and setting it equal to zero

$$\frac{\partial tr(P_k^2)}{\partial L_k} = 2((C_{k,2} P_{k|k-1} C_{k,2}^T - R_k M_k^T M_k C_{k,2}) - C_{k,2} M_k R_k^T + R_k M_k^T - C_{k,2} P_{k|k-1})$$

The solution is $L_k = (P_{k|k-1} C_{k,2}^T - M_k M_k^T)/(R_k + C_{k,2} P_{k|k-1} C_{k,2})$ (line 13).

Sensor attack $d_{k,1}$ estimation. Given $\hat{x}_k$, and the assumption that $E[\hat{x}_k] = 0$, sensor attack $d_{k,1}$ can be estimated by $y_{k,1}$ (in (5) (line 16)) $d_{k,1} = y_{k,1} - C_{k,1,1}\hat{x}_k = C_{k,1,1}\hat{x}_k + d_{k,1} + v_{k,1}$.

To obtain $d_{k,1}$, we solve (12) with $\alpha$ and $\beta$ as independent of $\hat{x}_k$. Comparing this equation with (11), we obtain $\hat{x}_k = \hat{x}_{k|k-1} - \hat{d}_{k-1}$. Minimizing $tr(P_k^2)$ with decision variable $L_k$ and $R_k$ (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) $D_{k+1} = D_k \cup ([\hat{x}_{k-1}^T, u_{k-1}^T]^T, \hat{x}_k - \hat{d}_{k-1})$.

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 $\epsilon(x_{k-1}, u_{k-1}) - \hat{f}_k([\hat{x}_{k-1}^T, u_{k-1}^T]^T)$ is the point of interest. This error becomes $\epsilon_k(x_{k-1}, u_{k-1})$ if $x_{k-1} = \hat{x}_{k-1}$.

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

$$\hat{D}_k \triangleq (\hat{X}_k, \hat{x}_k^+, \hat{X}_k^+, \hat{x}_k^+) = (\hat{X}_k, \hat{f}(\hat{x}_k) + w_k - f(\hat{x}_k) + f(\hat{x}_k) - \Delta X_k^+)$$

$$= (\hat{X}_k, \hat{Z}_k - f(\hat{x}_k) + f(\hat{x}_k) - \Delta X_k^+) = (\hat{X}_k, \hat{Z}_k + \Delta Z_k)$$

where $f(\hat{x}_k) = [f(\hat{x}_1), \ldots, f(\hat{x}_{N(k)})]$ and $\hat{Z}_k = f(\hat{x}_k) + w_k$. In the analysis, we use $X_k$ and $Z_k$ as the input and output to learn function $f$ as the classic GPR in Section III-B. Correspondingly, we consider $\Delta Z_k = [\Delta z_1, \ldots, \Delta z_{N(k)}] = -f(\hat{x}_k) + f(\hat{x}_k) - \Delta X_k^+$ 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 $\theta$ is chosen time-invariant to satisfy Assumption 5.1. Under Assumption 5.1, there exists a unique Reproducing Kernel Hilbert Space $H$ (RKHS) by the Moore-Aronszajn theorem (Theorem 6.1 in [6]).

Now consider the minimization of functional

$$J_k[\hat{f}] = \frac{1}{2}\|\hat{f}\|^2_H + \frac{1}{2\sigma^2} \sum_{i=1}^{N(k)} (\hat{X}_i^T - \hat{f}(\hat{x}_i))^2$$

(10)

where $\sigma^2$ is the variance of $w$ in (2). The second term works for data fitting and the first term smoothes 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

$$\hat{f}(x_k) = \sum_{i=1}^{N(k)} \alpha_i g(x_k, \hat{x}_i)$$

(11)

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

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

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

Assumption 5.2: The function $f$ is in RKHS $H$.

Since $g \in H$, any linear combination of $g$ is in RKHS $H$. Thus, function $f$ is in RKHS $H$ if and only if there exist a set of $x_i \in \mathbb{R}^{n+d}$, and $\beta_i \in \mathbb{R}$ such that $f(x) = \sum_{i=1}^{\infty}\beta_i g(x_i, x)$. 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 $X_k$ and output $Z_k$ in training data are sampled from probability distributions with corresponding probability measure $\mu(\hat{x}, \hat{z})$. State estimation errors $\Delta X_k$, $\Delta X_k^+$ are independent of $X_k$ and $Z_k$, 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 that such that $g(x, x') = \sum_{j=1}^{\infty} \lambda_j \phi_j(x) \phi_j(x')$ 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 $f(x) = \sum_{j=1}^{\infty} c_j \phi_j(x)$. There are infinitely many set of orthonormal eigenfunctions. Of them, we choose one such that Assumption 5.4 holds; e.g., $\phi_j(x) = e^{\sqrt{-1}\pi jx}$ (Fourier transform).

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

Theorem 5.1: Under Assumptions 5.1, 5.2, 5.3 and 5.4, it holds that
\[ \hat{f}_{k|E}[J_k|x^*_k] = \sum_{n=1}^{\infty} \frac{\lambda_j}{\lambda_j + \sigma^2/N(k)} \frac{1}{N(k)} \sum_{i=1}^{N(k)} \phi_j(\Delta x_i) + \ldots \]

Interpretation of Theorem 5.1. According to Theorem 5.1, identification error is described by
\[ \hat{f}_{k|E}[J_k|x^*_k] = f(x_{k-1}) - \hat{f}_{k|E}[J_k|x_{k-1}] = f(x_{k-1}) - \hat{f}_{k|E}[J_k|x_{k-1}] + \hat{f}_{k|E}[J_k|x_{k-1}] - \hat{f}_{k|E}[J_k|x_{k-1}] \]
\[ = \sum_{j=1}^{\infty} \left( c_j - \frac{\lambda_j}{\lambda_j + \sigma^2/N(k)} \right) \phi_j(x_{k-1}) \]
\[ + \sum_{j=1}^{\infty} \frac{\lambda_j}{\lambda_j + \sigma^2/N(k)} \phi_j(\Delta x_i) \]
\[ - \sum_{j=1}^{\infty} \lambda_j \phi_j(\Delta x_i) \]
\[ - \sum_{j=1}^{\infty} \frac{\lambda_j}{\lambda_j + \sigma^2/N(k)} \phi_j(\Delta x_i) \]
\[ - \sum_{j=1}^{\infty} \frac{\lambda_j}{\lambda_j + \sigma^2/N(k)} \phi_j(\Delta x_i) \]
\[ - \sum_{j=1}^{\infty} \frac{\lambda_j}{\lambda_j + \sigma^2/N(k)} \phi_j(\Delta x_i) \]
\[ - \sum_{j=1}^{\infty} \frac{\lambda_j}{\lambda_j + \sigma^2/N(k)} \phi_j(\Delta x_i) \]

The first term shows the function identification error given \( \Delta X_k = 0 \) and \( \Delta X_k^+ = 0 \). This error decreases as \( N(k) \rightarrow \infty \), where \( \lim_{k \rightarrow N} N(k) = \lim_{k \rightarrow \infty} k + (N(0) = \infty). \) The second term is the error induced by \( \Delta X_k. \) Since \( \phi_j(0) = 1 \) under Assumption 5.4, this error is zero if \( \Delta X_k = 0. \) If the total approximation error induced by \( \Delta X_k \) grows slower than a linear rate; i.e., there exist \( a > 0, 0 < \beta < 1, \) \( k_\gamma, \) such that \( \| N(k) - \sum_{i=1}^{N(k)} \phi_j(\Delta x_i) \| \leq \alpha \| N(k) \|^\beta \) for \( k \geq k_\gamma, \) then \( \lim_{N(k) \rightarrow \infty} c_j - c_j \leq \frac{1}{N(k)} \sum_{i=1}^{N(k)} \phi_j(\Delta x_i) = 0. \) Similarly, the third term is the error induced by \( \Delta x_k^+, \) and vanishes if \( \Delta x_k^+ = 0. \) Also, \( \lim_{N(k) \rightarrow \infty} \frac{1}{N(k)} \sum_{i=1}^{N(k)} \Delta X_i = 0 \) if there is the finite number of indices such that \( \Delta X_i^+ \neq 0. \) The last term is the error induced by the current input. This term decreases as the estimation error \( \| \hat{x}_{k-1} - x_{k-1} \| \) 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 \( J \) denote the set of hypothetical modes, and each mode \( j \in J \) 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 \( j \in J \) has different output \( y_{k,j}^1 \) and \( y_{k,j}^2 \) in (5). We conduct ArGPr algorithm (Algorithm 2) for each mode \( j \in J \) and find a priori 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 \( j_k. \)

We quantify the difference between the predicted output and the measured output as follows \( \nu_k = y_{k,j_k}^2 - C_{2,j_k} x_{k,j_k}^{-1}. \) The output error \( \nu_k \) is a multivariate Gaussian random variable. Therefore, the likelihood function is given by
\[ N(\nu_k; 0, \hat{P}_{k|k-1}) = \frac{1}{(2\pi)^{\frac{1}{2}} \| \hat{P}_{k|k-1}^{1/2} \|} \exp \left( -\frac{1}{2} \nu_k^T \hat{P}_{k|k-1}^{-1} \nu_k \right) \]
\[ = \frac{1}{(2\pi)^{\frac{1}{2}} \| \hat{P}_{k|k-1}^{1/2} \|} \exp \left( -\frac{1}{2} \nu_k^T \hat{P}_{k|k-1}^{-1} \nu_k \right) \]

\begin{table}[h]
\centering
\caption{System variables and parameters}
\begin{tabular}{|c|c|c|}
\hline
\textbf{System variables} & \textbf{\( f \)} & \textbf{\( \theta \)} \textbf{phase angle} \\
\hline
\textbf{PM} & \textbf{mechanical power} & \textbf{\( P_{ij} \)} & \textbf{power flow} \\
\hline
\textbf{PC} & \textbf{controllable load} & \textbf{\( P_L \)} & \textbf{net load} \\
\hline
\end{tabular}
\end{table}

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 \( G = (V, \mathcal{E}) \) where \( V \triangleq \{ 1, \ldots, 68 \} \) and \( \mathcal{E} \subseteq V \times V \) are the sets of buses and the set of transmission lines, respectively. Let us denote \( S_i \triangleq \{ i \in V \} \) the set of neighboring buses of \( i \in V. \) Each bus is either a generator bus \( i \in G \) or a load bus \( i \in L \) and \( \mathcal{V} = G \cup L. \) The dynamic system of a generator bus \( i \in G \) with attacks is described as the following nonlinear system [19]:
\[ \Delta \dot{\theta}_i(t) = \Delta f_i(t) + w_{1,i}(t) \]
\[ \Delta \dot{f}_i(t) = -\frac{1}{m_i} \left( D_i \Delta f_i(t) + \sum_{j \in S_i} \Delta P_{ij}(t) - \Delta P_{ji}(t) + a_{i,j}(t) + \Delta P_{L_i}(t) \right) + w_{2,i}(t) \]
\[ y_{i,k} = \left[ \Delta \theta_{i,k}, \Delta f_{i,k}, \Delta P_{elec,k}, T_0^2, [0, d_{s,i,k}^T] + v_{i,k} \right] \]
\[ \text{TABLE I: System variables and parameters} \]

VII. NUMERICAL SIMULATION

Authorized licensed use limited to: Penn State University. Downloaded on May 30, 2023 at 02:32:27 UTC from IEEE Xplore. Restrictions apply.
to both actuator attacks $d_{a,i}(t) = 30 \sin(\frac{i t}{105}) + \frac{t}{100}$ 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 $\Delta \theta_{j,k-1}$ from $j \in S_i$. 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 $i \in G$ ($i \in L$); i.e., $u_{i,k} = [\Delta P_{M_i,k}, \{\Delta \theta_{j,k}\}_{j \in S_i}]$ ($u_{i,k} = [\Delta P_{C_i,k}, \{\Delta \theta_{j,k}\}_{j \in S_i}]$). 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 $d_{2,i,k} = [0, -d_{a,i,k}/m_i]^T = [0, -3\sin(i t - i)/1000]^T$, 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.

**REFERENCES**


