Contents lists available at ScienceDirect

International Journal of Critical Infrastructure Protection

journal homepage: www.elsevier.com/locate/ijcip

A scalable specification-agnostic multi-sensor anomaly detection system for IIoT environments

Wissam Aoudi*, Magnus Almgren

Department of Computer Science and Engineering, Chalmers University of Technology, Gothenburg 412 96, Sweden

ARTICLE INFO

Article history:
Received 20 November 2019
Revised 30 June 2020
Accepted 3 September 2020
Available online 20 September 2020

Keywords:
IIoT
PASAD
Departure-based detection
Critical infrastructure

ABSTRACT

Advanced sensing is a key ingredient for intelligent control in Industrial Internet of Things (IIoT) environments. Coupled with enhanced communication capabilities, sensors are becoming increasingly vulnerable to cyberattacks, thereby jeopardizing the often safety-critical underlying cyber-physical system. One prominent approach to sensor-level attack detection in modern industrial environments, named PASAD, has recently been proposed in the literature. PASAD is a process-aware stealthy-attack detection mechanism that has shown promising capabilities in detecting anomalous, potentially malicious behavior through real-time monitoring of sensor measurements. Although fast and lightweight, a major limitation of PASAD is that it is univariate, meaning that only a single sensor can be monitored by one instance of the algorithm. This impediment poses serious concerns on its scalability, especially in modernized industrial environments, which typically employ a plethora of sensors. This paper generalizes PASAD to the multivariate case, where a plurality of sensors can be monitored concurrently with little added complexity. This generalization has the evident advantage of offering scalability potential for deployment in future-focused industrial environments, which are undergoing growing integration between the digital and physical worlds.

1. Introduction

The new generation of cyber-physical systems underpinned by the Industrial Internet of Things (IIoT) paradigm often employ sensors as a means to gather data from the physical world for controllers to decide on proper actuation. There is an implicit trust in sensors in the sense that the received measurements are assumed to be authentic and reflect the actual physical state of the sensed environment, which may potentially lead to malicious control decisions if compromised by motivated adversaries. In environments that involve controlling safety-critical processes (e.g., nuclear plants, power and gas distribution, automated transport systems, etc.), such an unsubstantiated assumption may wreak consequential havoc on society at large.

Recently, a process-aware stealthy-attack detection mechanism (PASAD) has been proposed as a novel approach to detecting cyberattacks on cyber-physical systems by detecting implausible sensor behavior [1]. The method, which is partly based on a new time-series analysis technique known as *singular spectrum analysis* [2], consists of a training phase and a detection phase. In the training phase, an initial part of the time series collected under normal operating conditions is used to define a baseline for the underlying system behavior. Afterwards, in the detection phase, a departure-detection mechanism continuously verifies whether or not current sensor observations conform to the established baseline.

The enabler of data-driven anomaly-detection techniques such as PASAD is the *regularity* of the process-level traffic in industrial environments, which stems from the static nature of the communication between field devices in automation systems. The method has demonstrated promising capabilities in detecting potential attack-induced *departure* of the physical process from normal dynamics through real-time monitoring of sensor measurements, and has been shown to be sufficiently lightweight to run on limited-resource hardware [3].

However, one fundamental limitation of PASAD is that it is univariate; that is to say, a single instance of the algorithm can only process one sensor at a time. With the proliferation of sensors in modern industrial environments, this aspect of the method is likely to be problematic, especially that the “smartness” of future-focused industrial environments is tightly related to employing more sensors for the task of collecting data to gain more insight into the environment and increase operational efficiency.

Although lightweight, fast, and suitable for distributed environments, the canonical way of monitoring n sensors simultaneously with PASAD at choke points is to awkwardly train and run n in-

* Corresponding author.
E-mail address: wissam.aoudi@chalmers.se (W. Aoudi).

https://doi.org/10.1016/j.ijcip.2020.100377
1874-5482/© 2020 Elsevier B.V. All rights reserved.

stances of the algorithm. Evidently, as the number of sensors grows large, the total time-to-train and allocated memory for deployment can quickly become overwhelming. The inevitable complexity and overhead involved in this approach is likely to hinder large-scale deployment of PASAD in industrial environments. Yet, with the monotonically increasing utilization of sensors, the scalability property is, at any rate, highly desirable.

In this paper, we introduce M-PASAD, a multivariate extension of PASAD that can handle a plurality of sensors efficiently. Rather than employing a plurality of PASAD instances, our proposed approach adapts the underlying theory to accommodate multiple sensors with little added complexity both in terms of running time and memory footprint. As such, M-PASAD inherits key features from PASAD, such as its noise-reduction potential, its capability to detect subtle structural changes in the monitored signal, and its efficient evaluation of the departure score during the detection phase.

We evaluate our approach using the popular Tennessee-Eastman (TE) process control simulation model and perform benchmarking using publicly available data and attack scenarios. Experimental results demonstrate that M-PASAD trains an *order of magnitude* faster than PASAD, requires constant memory independently of the number of sensors, and is consistently faster at computing the departure score during the detection phase. This boost in performance enables the deployment of M-PASAD on limited-resource hardware (e.g., microcontrollers) even when monitoring a relatively large number of sensors, which is hardly possible for the univariate version of the algorithm.

The remainder of this paper is laid out as follows: We start off by presenting PASAD in Section 2, then Section 3 introduces the multivariate extension M-PASAD, which we evaluate in Section 5. In Section 6, we review related literature and we conclude this work in Section 7.

2. PASAD: the univariate case

This section presents an overview of PASAD. A comprehensive account of both the theoretical and the practical aspects of the method can be found in Aoudi et al. [1].

PASAD is a data-driven anomaly detection technique that takes as input a continuous time series of sensor measurements and outputs an alert upon detection of unexpected behavior.

Consider a continuous real-valued time series

$$\mathcal{T} = x_1, x_2, \ldots, x_N, x_{N+1}, \ldots \tag{1}$$

of process measurements corresponding to a single sensor, PASAD initially embeds $\mathcal{T}$ into a *trajectory space* and then determines a basis for a low-dimensional subspace in which the deterministic behavior of the underlying signal is presumably more pronounced. The embedding is performed during an offline training phase by unfolding a subseries of $\mathcal{T}$ of length N into a *trajectory Hankel matrix*

$$\mathbf{X} = \begin{bmatrix} x_1 & x_2 & \ldots & x_K \\ x_2 & x_3 & \ldots & x_{K+1} \\ \vdots & \vdots & \ddots & \vdots \\ x_L & x_{L+1} & \ldots & x_N \end{bmatrix}, \tag{2}$$

whose $K = N - L + 1$ columns are the L-dimensional *lagged* vectors

$$\mathbf{x}_i = (x_i, x_{i+1}, \ldots, x_{i+L-1})^T. \tag{3}$$

The next step in the training phase is to determine a subset of eigenvectors of the covariance matrix $\mathcal{C} = \mathbf{X}\mathbf{X}^T$ that can sufficiently describe the underlying signal. To compute the eigenvectors of the covariance matrix, the singular value decomposition (SVD) of the trajectory matrix is performed to obtain the L left-singular vectors of $\mathbf{X}$, or equivalently, the eigenvectors of $\mathcal{C}$. The r leading orthonormal eigenvectors $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_r$ that correspond to the r largest eigenvalues form the basis of a so-called *signal subspace*, where r is referred to as the *statistical dimension* of the time series. The parameters N, L, and, r are determined during the offline training phase.

Now that the signal subspace has been identified by means of an orthonormal basis, PASAD proceeds by constructing a linear transformation that projects the lagged vectors (Eq. (3)) in the trajectory space onto the signal subspace, where the detection of anomalies is supposed to take place. This linear transformation is given by $\mathbf{U}^T$ such that

$$\mathbf{U} = [\mathbf{u_1} : \mathbf{u_2} : \ldots : \mathbf{u_r}]. \tag{4}$$

In fact, by virtue of being a *partial isometry*, the linear transformation $\mathbf{U}^T$ transforms the lagged vectors from the trajectory space into the r-dimensional Euclidean space $\mathbb{R}^r$, which has been shown to be *isomorphic* to the signal subspace [1], without the need for the more expensive explicit projection onto the signal subspace. The authors argued that by obviating the need to project the lagged vectors onto the signal subspace in order to compute vector norms, the detection procedure gained a significant performance uplift, and coined this property as the *isometry trick*. The isometry trick states that for an arbitrary vector $\mathbf{x}$ in the trajectory space, computing the norm of the vector $\mathbf{U}^T\mathbf{x}$ has the effect of implicitly projecting $\mathbf{x}$ onto the signal subspace and computing its norm there. In mathematical terms, for an arbitrary vector $\mathbf{x}$, it holds that $||\mathbf{U}^T\mathbf{x}|| = ||\mathbf{U}\mathbf{U}^T\mathbf{x}||$. The isometry trick is applicable in the multivariate case since $\mathbf{U}^T$ is still a partial isometry (row vectors are orthonormal).

On account of the noise-reduced representation in the signal subspace, the transformed lagged vectors follow a pattern under normal operating conditions. The task in the detection phase is to detect a potential *departure* from this pattern that correlates with a structural change in the monitored sensor signal. To this end, every lagged vector of sensor measurements is used to compute a *departure score* at every iteration in the detection phase.

One way of evaluating the departure score is by computing the Euclidean distance between the most recent test vector and the centroid of the cluster formed by the vectors projected during the training phase. Consequently, an anomaly—or rather a departure—would drive the distance from the cluster to higher values, and accordingly, PASAD raises an alert whenever this distance crosses a predetermined threshold.

3. M-PASAD: the multivariate extension

We now introduce M-PASAD, the multivariate extension of PASAD, whereby multiple sensors can be processed simultaneously to detect anomalous behavior in the underlying system. A succinct description of the workflow of M-PASAD is presented in the schematic in Fig. 1.

As argued in Section 1, a theoretical solution to the multivariate case of PASAD is motivated by the impracticality of running a large number of instances of the algorithm to monitor a plurality of sensors. Our central argument in this work is that M-PASAD eliminates the need to run multiple instances while still inheriting the key characteristics and prominent features from the univariate version.

Essentially, the objective is to identify a signal subspace that describes the *common* underlying signal during the training phase and to construct a mechanism by which the n most recent sensor measurements are incorporated in the computation of the departure score during the detection phase.

To make progress, consider n time series of measurements

$$\mathcal{T}^{(j)} = x_1^{(j)}, x_2^{(j)}, \ldots, x_N^{(j)}, x_{N+1}^{(j)}, \ldots \tag{5}$$

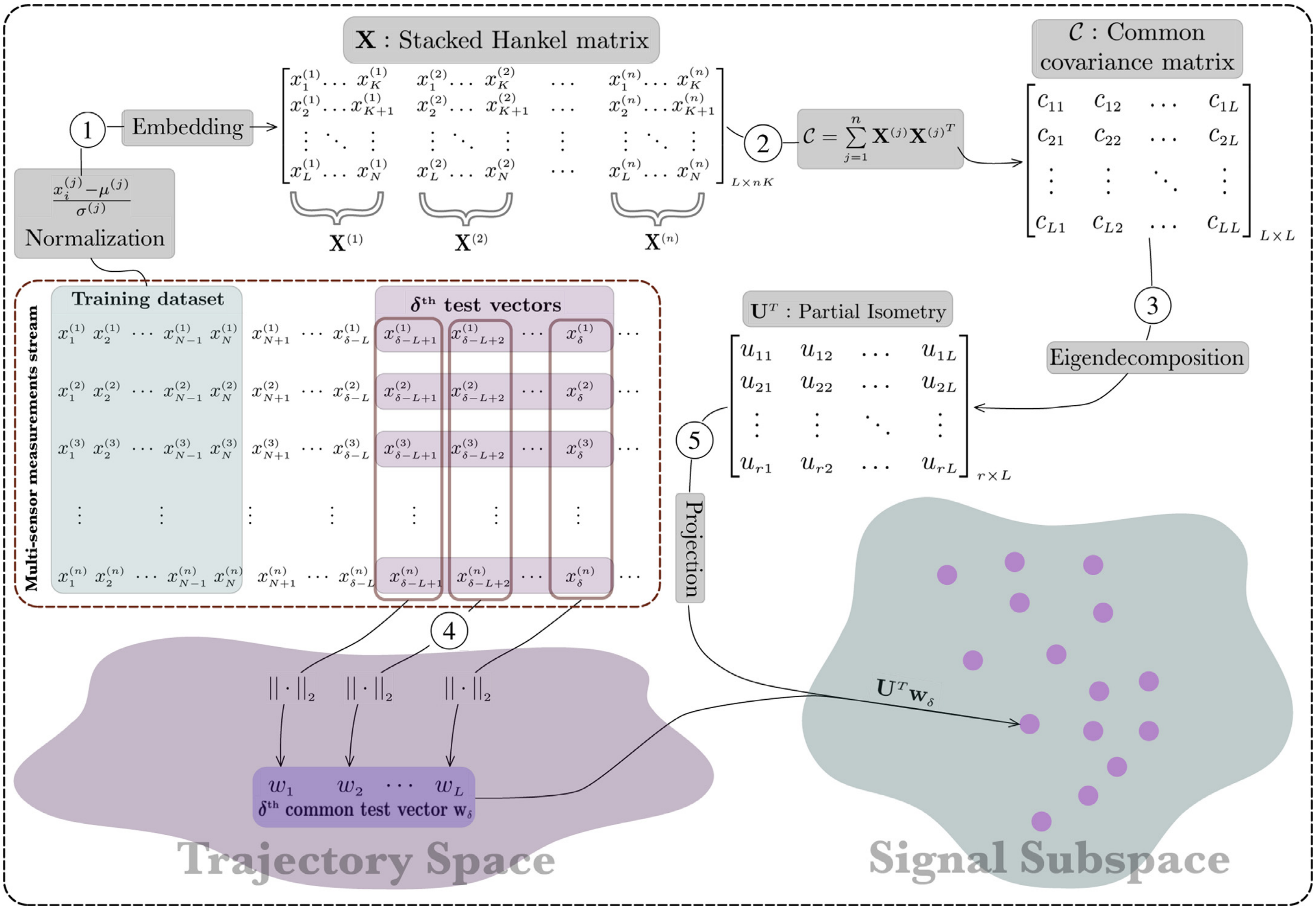

X : Stacked Hankel matrix
C : Common covariance matrix
1
Embedding
$\frac{x_i^{(j)}-\mu^{(j)}}{\sigma^{(j)}}$
Normalization
$\mathbf{X}^{(1)}$
$\mathbf{X}^{(2)}$
$\mathbf{X}^{(n)}$
$L\times nK$
2
$\mathcal{C}=\sum_{j=1}^{n}\mathbf{X}^{(j)}\mathbf{X}^{(j)T}$
$L\times L$
3
Eigendecomposition
U^T : Partial Isometry
$r\times L$
5
Projection
Training dataset
δ^{th} test vectors
Multi-sensor measurements stream
4
$||\cdot||_2$
w_1 w_2 $\cdots$ w_L
δ^{th} common test vector $\mathbf{w}_\delta$
$\mathbf{U}^T\mathbf{w}_\delta$
Trajectory Space
Signal Subspace

Fig. 1. A schematic depicting the workflow of M-PASAD.

where $j = 1, 2, \ldots, n$, corresponding to n sensors. Before commencing the training phase, as a pre-processing step, the time series of measurements are standardized to prevent some sensors from overweighing other sensors due to difference in scale. This is performed by subtracting the mean and dividing by the standard deviation after having inferred the parameters from the training subseries of the respective sensors. Specifically, for the time series $\mathcal{T}^{(j)}$, the normalized measurements are evaluated as

$$\frac{x_i^{(j)} - \mu^{(j)}}{\sigma^{(j)}}, \tag{6}$$

where $\mu^{(j)}$ and $\sigma^{(j)}$ stand for the mean and standard deviation computed over the training subseries $\mathcal{T}^{(j)} = x_1^{(j)}, x_2^{(j)}, \ldots, x_N^{(j)}$ of the j^{th} sensor.

3.1. Training phase: the stacked hankel matrix

The main idea in the training phase is to embed all the time-series data in a *common* trajectory space. Our construction for identifying a common trajectory space is a *stacked Hankel matrix*, where the $L \times K$ trajectory matrices $\mathbf{X}^{(j)}$ of the respective sensors are stacked into one $L \times nK$ common trajectory matrix $\mathbf{X}$ as follows (see (1) in Fig. 1)

$$\mathbf{X} = \left[\mathbf{X}^{(1)} : \mathbf{X}^{(2)} : \ldots : \mathbf{X}^{(n)}\right]. \tag{7}$$

As in the univariate case, a spectral decomposition of the trajectory matrix is performed to obtain a basis for the signal subspace. Solving the SVD of $\mathbf{X}$, however, can quickly become impractical as the number of sensors n grows sufficiently large. A more sensible approach is to solve the eigenvalue decomposition of the covariance matrix $\mathcal{C} = \mathbf{X}\mathbf{X}^T$, which proves to be an efficient alternative since $\mathcal{C}$ is of dimension $L \times L$, similar to the univariate case (see (2) in Fig. 1). Another advantage of this approach is that the covariance matrix can be evaluated efficiently by leveraging the additive property

$$\mathcal{C} = \mathbf{X}\mathbf{X}^T = \sum_{j=1}^{n} \mathbf{X}^{(j)}\mathbf{X}^{(j)T}, \tag{8}$$

allowing the covariance matrix to be constructed sequentially without the need to store the individual trajectory matrices of all n sensors in memory. As the covariance matrix $\mathcal{C}$ is of dimension $L \times L$, similar to the covariance matrix in the univariate case, Eq. (8) can be regarded as the *only* added complexity during the training phase.

Finally, after obtaining the eigenvectors by spectral decomposition, which describe the common structure of the sensor signals and form a basis for a common signal subspace, the partial isometry $\mathbf{U}^T$ is constructed as in the univariate version (see (3) in Fig. 1).

Unlike the case with PASAD, however, the linear transformation cannot be applied directly on the lagged vectors to map them to the signal subspace. Next, we introduce a methodical procedure for constructing a unified vector, namely an *aggregate test vector*, out of every n lagged vectors, to enable this mapping.

3.2. Detection phase: the aggregate test vector

The principal challenge in the detection phase—and indeed to the success of M-PASAD—is to construct a single test vector that meaningfully represents the n individual test vectors for every new set of sensor measurements, in order to be able to project onto the signal subspace and perform the intended analysis there.

What follows in this section is based on the key idea of examining *the change in the covariance matrix $\mathcal{C}$ incurred by adding a single set of lagged vectors to the common trajectory matrix* $\mathbf{X}$, or equivalently, by incorporating an additional set of new measurements into the training subseries.

For the sake of illustration, we will first consider the case of two sensors with time series $\mathcal{T}^{(1)}$ and $\mathcal{T}^{(2)}$ having $L \times K$ trajectory matrices $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$ respectively. We shall subsequently show that the generalization to the n-dimensional case is straightforward.

Originally, the common trajectory matrix in the 2D case is the $L \times 2K$ stacked Hankel matrix $\mathbf{X} = \left[\mathbf{X}^{(1)} : \mathbf{X}^{(2)}\right]$, or more explicitly

$$\mathbf{X} = \begin{bmatrix} x_1^{(1)} & x_2^{(1)} & \dots & x_K^{(1)} & x_1^{(2)} & x_2^{(2)} & \dots & x_K^{(2)} \\ x_2^{(1)} & x_3^{(1)} & \dots & x_{K+1}^{(1)} & x_2^{(2)} & x_3^{(2)} & \dots & x_{K+1}^{(2)} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ x_L^{(1)} & x_{L+1}^{(1)} & \dots & x_N^{(1)} & x_L^{(2)} & x_{L+1}^{(2)} & \dots & x_N^{(2)} \end{bmatrix}$$

which, according to Eq. (8), results in the *common covariance matrix*

$$\begin{aligned} \mathcal{C} &= \mathbf{X}\mathbf{X}^T \\ &= \mathbf{X}^{(1)}\mathbf{X}^{(1T)} + \mathbf{X}^{(2)}\mathbf{X}^{(2T)}. \end{aligned} \tag{9}$$

Now, in order to analyze the change incurred by incorporating an additional set of (in this case two) individual lagged vectors into the covariance matrix, let $\tilde{\mathbf{X}}^{(1)} = \left[\mathbf{X}^{(1)} : \mathbf{x}_{K+1}^{(1)}\right]$ and $\tilde{\mathbf{X}}^{(2)} = \left[\mathbf{X}^{(2)} : \mathbf{x}_{K+1}^{(2)}\right]$ be $L \times (K+1)$ augmented trajectory matrices, then the modified common trajectory matrix is given by

$$\tilde{\mathbf{X}} = \begin{bmatrix} x_1^{(1)} & \dots & x_K^{(1)} & x_{K+1}^{(1)} & x_1^{(2)} & \dots & x_K^{(2)} & x_{K+1}^{(2)} \\ x_2^{(1)} & \dots & x_{K+1}^{(1)} & x_{K+2}^{(1)} & x_2^{(2)} & \dots & x_{K+1}^{(2)} & x_{K+2}^{(2)} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ x_L^{(1)} & \dots & x_N^{(1)} & x_{N+1}^{(1)} & x_L^{(2)} & \dots & x_N^{(2)} & x_{N+1}^{(2)} \end{bmatrix},$$

or more compactly,

$$\begin{aligned} \tilde{\mathbf{X}} &= \left[\tilde{\mathbf{X}}^{(1)} : \tilde{\mathbf{X}}^{(2)}\right] \\ &= \left[\mathbf{X}^{(1)} : \mathbf{x}_{K+1}^{(1)} : \mathbf{X}^{(2)} : \mathbf{x}_{K+1}^{(2)}\right], \end{aligned} \tag{10}$$

where $\mathbf{x}_{K+1}^{(1)}, \mathbf{x}_{K+1}^{(2)}$ are the two lagged vectors added to the trajectory matrices of the two time series respectively. Letting $\mathbf{u} := \mathbf{x}_{K+1}^{(1)}$ and $\mathbf{v} := \mathbf{x}_{K+1}^{(2)}$, the modified covariance matrix can then be expressed as

$$\begin{aligned} \tilde{\mathcal{C}} &= \mathbf{X}^{(1)}\mathbf{X}^{(1T)} + \mathbf{x}_{K+1}^{(1)}\mathbf{x}_{K+1}^{(1)^T} + \mathbf{X}^{(2)}\mathbf{X}^{(2T)} + \mathbf{x}_{K+1}^{(2)}\mathbf{x}_{K+1}^{(2)^T} \\ &= \mathcal{C} + \mathbf{u}\mathbf{u}^T + \mathbf{v}\mathbf{v}^T. \end{aligned} \tag{11}$$

As Eq. (11) implies, due to the additive property of the covariance matrix in Eq. (8), every set of new sensor measurements from all sensors adds to the covariance matrix a sum of outer products of the respective lagged vectors. Thus, our quest for finding a single vector whose components are functions of the individual lagged vectors boils down to solving for a vector $\mathbf{w}$ such that

$$\mathbf{w}\mathbf{w}^T = \mathbf{u}\mathbf{u}^T + \mathbf{v}\mathbf{v}^T. \tag{12}$$

Expanding Eq. (12) as

$$\begin{bmatrix} w_1^2 & \mathrm{x} & \dots & \mathrm{x} \\ \mathrm{x} & w_2^2 & \dots & \mathrm{x} \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{x} & \mathrm{x} & \dots & w_L^2 \end{bmatrix} = \begin{bmatrix} u_1^2 + v_1^2 & \mathrm{x} & \dots & \mathrm{x} \\ \mathrm{x} & u_2^2 + v_2^2 & \dots & \mathrm{x} \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{x} & \mathrm{x} & \dots & u_L^2 + v_L^2 \end{bmatrix}$$

and examining the diagonal entries, one arrives at a solution to the ith component of the aggregate test vector given by

$$w_i = \sqrt{u_i^2 + v_i^2}. \tag{13}$$

Note that for simplicity reasons, only the positive square roots were considered in the solution, which were empirically found to yield a sufficiently accurate end result. Also note that the arbitrary non-diagonal entries need not be considered for the solution since the u's and v's on the right-hand side are known.

Let $\mathbf{w}_\delta$ be the δth aggregate test vector, then for the general n-dimensional case, the components w_i of $\mathbf{w}_\delta$ can naturally be obtained as

$$w_i = \sqrt{\left(x_i^{(1)}\right)^2 + \left(x_i^{(2)}\right)^2 + \ldots + \left(x_i^{(n)}\right)^2}, \tag{14}$$

where $i = \delta - L + 1, \delta - L + 2, \ldots, \delta$.

Eq. (14) effectively implies that the test vectors $\mathbf{x}_{K+1}^{(1)}, \mathbf{x}_{K+1}^{(2)}, \ldots, \mathbf{x}_{K+1}^{(n)}$ from all n sensors can be represented by a single aggregate test vector $\mathbf{w}_\delta$.

If we let $\mathbf{y}_i = (x_i^{(1)}, x_i^{(2)}, \ldots, x_i^{(n)})^T$, then the δth aggregate test vector is precisely

$$\mathbf{w}_\delta = (||\mathbf{y}_{\delta-L+1}|| \; ||\mathbf{y}_{\delta-L+2}|| \; \dots \; ||\mathbf{y}_\delta||)^T, \tag{15}$$

where $|| \cdot ||$ is the ℓ_2 norm (see (4) in Fig. 1).

Importantly, the $(\delta+1)$th aggregate test vector only requires computing $||\mathbf{y}_{\delta+1}||$ since the previous $L-1$ components have already been computed for the preceding vector due to the sequential nature of the algorithm. Therefore, computing the norm of the vector containing the most recent sensor measurements according to Eq. (14) is the *only* added complexity in the detection phase.

Now that an aggregate test vector is constructed at every iteration, the partial isometry can be used to map the time-series data from the trajectory space to the signal subspace (see (5) in Fig. 1).

Thereafter, at the δth iteration during the detection phase, if the squared Euclidean distance between the test vector and the centroid $\mathbf{c}$ of the cluster formed by the training vectors is to be used as a metric, the δ^{th} departure score would then be evaluated as

$$D_\delta = ||\mathbf{U}^T\mathbf{w}_\delta - \mathbf{c}||^2. \tag{16}$$

Finally, as in the univariate case, whenever D_δ crosses a predetermined threshold, M-PASAD raises an alert. The threshold is determined by testing on a validation subseries of sensor readings during normal operating conditions. Based on the obtained departure scores, we then choose a statistically plausible level beyond which the algorithm generates an alarm.

4. Discussion and remarks

Both PASAD and M-PASAD are train-only-once models, making them practical and easy to maintain. However, the training model might need to be updated (offline) when there is a change in the sensor dynamics. This could happen, for instance, when the sensor is connected to a control loop and the logic/configuration of the controller has been altered/updated. That said, regularly updating the training parameters could be plausible if there are enough resources to account for latent factors that may affect the detection accuracy in the long term.

It is also worth noting that, in industrial environments, such sudden events as emergency stop signals may sometimes be triggered. These events are typically rare and may be triggered once every 5 years. As these events may be deemed anomalous by our method, we argue that blacklisting and/or whitelistening can be a good complementary measure for flagging such rare events.

When it comes to how long the training data should be, the rule of thumb is to train on a long enough subseries that contains a whole cycle of the underlying signal (and preferably multiple cycles). But longer data does not necessarily mean longer time if the sampling rate is taken into account. Also, the starting position for training in the time series of sensor measurements can be arbitrary.

Multivariate PASAD is not a direct application of multivariate singular spectrum analysis (SSA) [4,5]. The main task in SSA of general time series is to reconstruct the signal and perform forecasting by identifying a so-called linear recurrent relation. SSA is also primarily used for exploratory analysis of complex time series by identifying trends, periods, and other structures. The construction we used for spectral decomposition in M-PASAD (stacked Hankel matrix) is known and used in some SSA-related methodologies. However, the successful M-PASAD procedure is due to the mechanism of constructing a meaningful aggregate test vector on the fly as sensor measurements arrive, thereby enabling the remaining procedures developed for PASAD, namely, projection onto subspace and computing departure scores.

Finally, as is typically the case with every data-driven anomaly-based detection algorithm, we make the implicit assumption that the data used for training is attack-free.

5. Evaluation

We evaluate M-PASAD using the Tennessee-Eastman (TE) process control model. While the original algorithm has been shown to work in more complex scenarios involving sensors with more varying dynamics [1,6,7], this choice of simulation platform is particularly suitable for evaluating M-PASAD since it was used to evaluate PASAD,[1] which makes the results comparable. We first start with a high-level description of the TE process, then we present experimental results followed by an evaluation of M-PASAD's performance.

We investigate the detection accuracy of M-PASAD and benchmark its performance against PASAD with respect to running time of both the training phase and the detection phase.

5.1. The tennessee-eastman process

The attacks were performed using the open-source DVCP-TE Simulink implementation of the TE process control simulation model [8,9].[2] The model simulates a real plant-wide chemical process, which was originally released to challenge the control theory community to develop and benchmark different optimized control strategies. Later, however, the TE model has become a popular choice amongst security researchers for evaluating attack-detection solutions for being a realistic and safe environment for experimentation [8,10,11].

The simulated chemical process produces two liquid products (G, H) from four gaseous reactants (A, C, D, E), in addition to a byproduct (F) and an inert (B), making a total of eight chemical components, coded after the first eight letters of the alphabet. There are five main operation units: reactor, condenser, recycle compressor, vapor-liquid separator, and stripping column. The gaseous reactants, fed by three different feeds, react to form liquid products. These products, along with residual reactants, leave the reactor as vapors, which are then cooled by the condenser to return to the liquid state. Next, the vapor-liquid separator isolates the non-condensed vapors, which are fed once again to the reactor by using a centrifugal compressor. The condensed components, on the other hand, move to a stripping column to remove the remaining residual reactants. The final product (mix of G and H) exits the stripper and heads towards a refining section that separates its components. Finally, the inert and the byproduct are purged in the vapor-liquid separator as vapor.

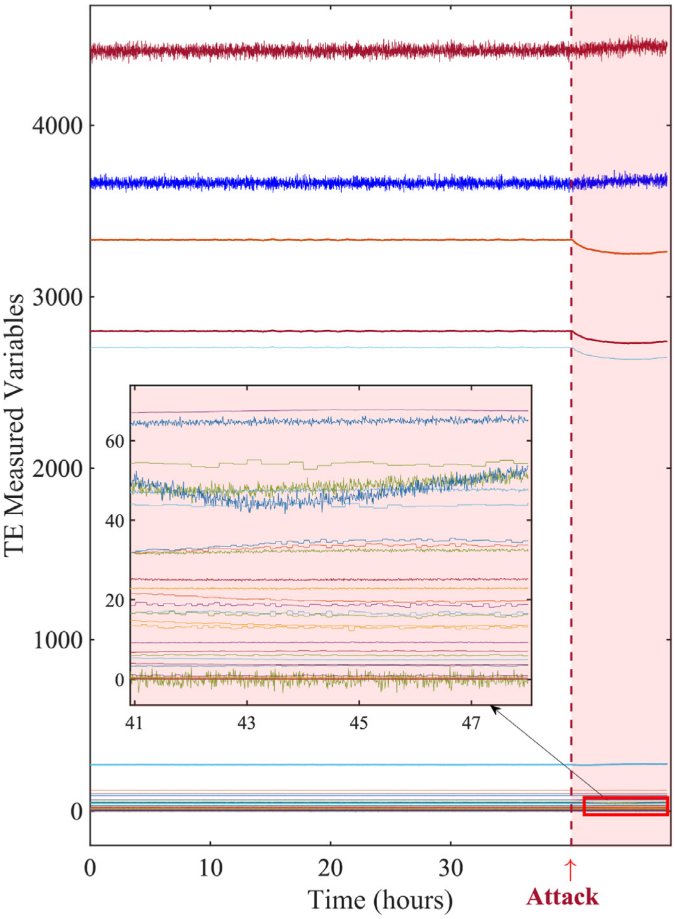


Fig. 2. Evolution of the 41 TE measured variables over time before and after an attack.

The process has 41 measured variables that comprise the readings of the sensors. The controller reads the measured values and, based on the implemented control strategy, sends commands to actuators that control different process flows.

The time series of sensor measurements displayed in Fig. 2 show the evolution of the sensors before and after an attack is in effect. As shown in the figure, the sensors react differently to the attack, some exhibiting less obvious change in dynamics than others.

5.2. Experimental results

Integrity attacks on the TE process were simulated on both sensors and actuators as depicted in Fig. 3. Once attackers gain access to a control network in charge of a process, they can either compromise the data fed to the controller by tampering with the process readings transmitted by the sensors, or tamper with the commands sent by the controller to the actuators. In the former case, the controller makes decisions based on maliciously modified data, potentially leading to the destabilization of the process.

[1] Available at https://github.com/mikeliturbe/pasad.
[2] Available at https://github.com/satejnik/DVCP-TE.

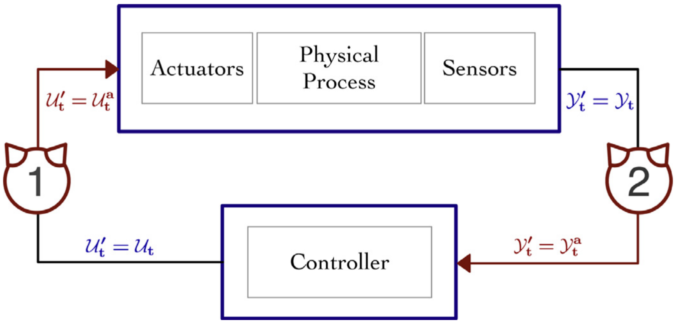

Actuators
Physical Process
Sensors
$\mathcal{U}'_t = \mathcal{U}^a_t$
$\mathcal{Y}'_t = \mathcal{Y}_t$
1
2
$\mathcal{U}'_t = \mathcal{U}_t$
Controller
$\mathcal{Y}'_t = \mathcal{Y}^a_t$

Fig. 3. Attack scenarios on control systems: Attacks on actuator signals (1) and attacks on sensor signals (2).

In the latter case, the process acts on arbitrary commands sent by the attacker rather than on the commands sent by the controller.

The attacks were designed with two main objectives in mind: *(i) Stealth attacks*, designed to cause slow damaging perturbations and aim to degrade the performance of the process; and *(ii) direct damage attacks* where the attacker's goal is to cause damage to physical equipment (e.g., reactor, stripping column, pipes, etc.) that is essential for the process to run, mainly by driving the process to unsafe operating conditions (e.g., high temperature or pressure).

The attacker model proposed by Krotofil et al. [11] is used in the experiments, where measured variables $u'_i(t)$ and $y'_i(t)$ are simulated as

$$u'_i(t), y'_i(t) = \begin{cases} u_i(t), y_i(t) & \text{for } t \notin T_a \\ u^a_i(t), y^a_i(t) & \text{for } t \in T_a \end{cases} \tag{17}$$

such that $u_i(t)$, $y_i(t)$ and $u^a_i(t), y^a_i(t)$ correspond to the i^{th} original and modified measured variables at time $0 \leq t \leq T$ respectively, T is the duration of the simulation run, and T_a is the attack interval.

5.2.1. Stealth attacks

In stealth attacks, attackers try to remain undetected by keeping the process readings under a set of thresholds, which if exceeded, alarms are raised and operators are alerted. In the stealth attack used in this work, the manipulated variable corresponding to the purge valve that controls the output of accumulating reactor gases is modified. Opening this valve more than necessary would result in products being wasted, since in order to maintain the production rate, more reactants would need to be purged from the reactor and fed to the process. However, opening the purge valve too much would drive the reactor pressure to a too low level, causing the process to halt. In this scenario, we set the manipulated variable to 28% open, which is wide enough to degrade the performance of the process without interrupting the process execution.

5.2.2. Direct damage attacks

Direct-damage attacks aim to sabotage equipment and eventually lead to the interruption of the process. In the direct-damage attack used in this work, the manipulated variable corresponding to the valve that controls the cooling water flow to the reactor to prevent its pressure from reaching dangerous levels is modified. Therefore, it is a critical valve in the process. In this scenario, the valve is set to 35.9% open, slightly less than the optimal setting. Consequently, the pressure adds up inside the reactor and the TE process execution eventually stops due to reaching the predefined safety limits.

5.2.3. Description of results

We apply M-PASAD under both of the described attack scenarios, where all 41 sensors were monitored concurrently. The results are displayed in Fig. 4a and b for the `direct-damage` and `stealth` attacks respectively. As mentioned earlier, in the direct-damage attack, the manipulated variable corresponding to the valve that controls the cooling water flow to the reactor is modified to cause the pressure to add up inside the reactor and reach dangerous levels. In the stealth attack, the manipulated variable corresponding to the purge valve that controls the output of accumulating reactor gases is carefully modified to degrade the performance of the process without interrupting the process execution.

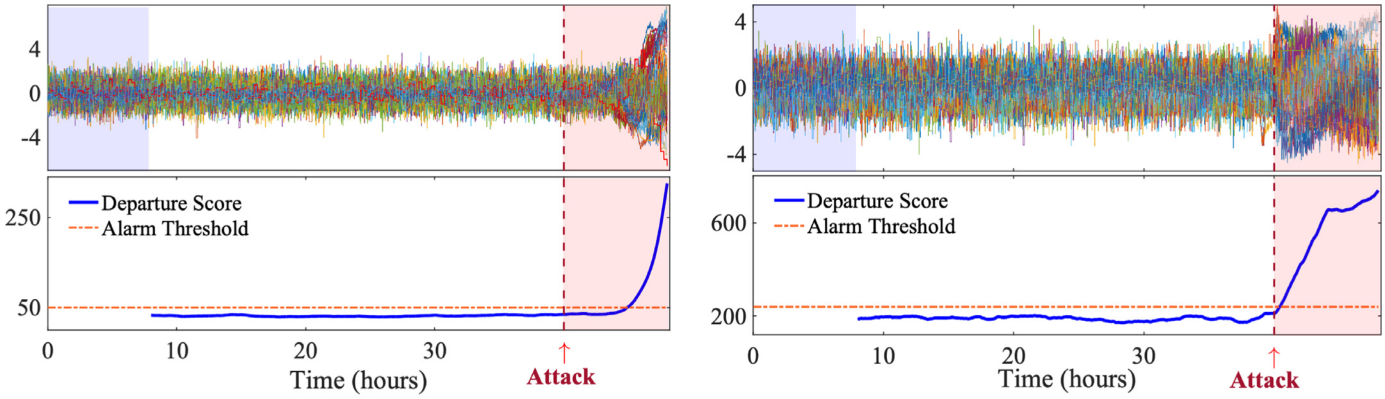
Note that the blue shaded region in the figures corresponds to the subseries used for training and the red shaded region marks the attack time frame. The upper plots show the evolution of the standardized sensor measurements both under normal operating conditions and after the attacks were initiated at $T = 40$ h. The lower plots display the departure scores computed iteratively by M-PASAD for every sensor measurement during the detection phase, and the predetermined thresholds. As explained in Section 3.2, the departure score is computed by evaluating the distance between the most recent test vector and the cluster of normal vectors formed during the training phase. Note that the difference in scale between the two attack scenarios is due to the fact that stealthy attacks require a larger value for the lag parameter to allow for the detection of subtle changes in the sensor dynamics. It is also worth pointing out that it takes longer for the direct-damage attack to be detected because it takes time for the pressure to add up inside the reactor to abnormal levels after changing the valve state. Once such levels are reached however, the impact of the attack on the process accelerates and quickly drives it to an unsafe state.

As the results in Fig. 4 indicate, M-PASAD successfully detects both types of attacks. Furthermore, one particular advantage of PASAD over similar methods in the domain is its distinctive capability to detect subtle changes in the noisy signal, typically induced by a stealthy attack. As demonstrated in Fig. 5, this important feature is not compromised in M-PASAD. The 8 sensors displayed in Fig. 5a were specifically picked from the dataset for having a subtle non-obvious reaction to a `stealth` attack. Then, Fig. 5b shows how M-PASAD indeed manages to detect the subtle changes, suggesting that the procedure for aggregating test vectors does not incur noticeable information loss.

5.3. Performance benchmarking

The reduction in memory footprint using our approach is fairly clear: M-PASAD requires storing only one $r \times L$ matrix to perform the detection, independently of the number of sensors it monitors, whereas PASAD requires storing n such matrices, thereby reducing the space complexity from linear in the size of the partial isometry to constant.

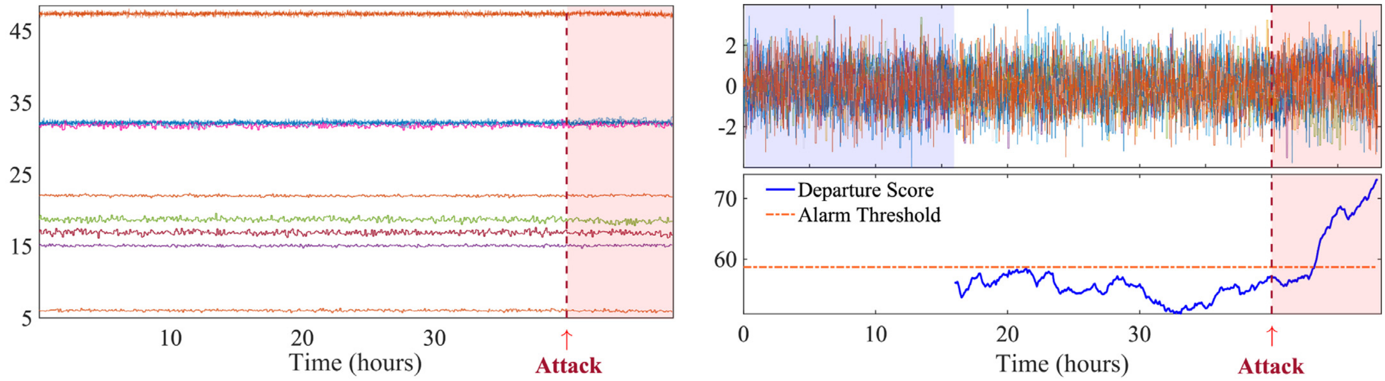
To highlight the gain in execution time, on the other hand, we have benchmarked M-PASAD against PASAD with respect to the time-to-train and the time-to-test. As we claimed in Section 3, M-PASAD's performance is superior to that of PASAD's because the only overhead imposed is constructing the trajectory matrix according to Eq. (8) in the training phase, and evaluating the Lth component of the δth aggregate test vector according to Eq. (15) in the detection phase. In Fig. 6, we substantiate this claim by comparing training time and detection time of both algorithms as the number of monitored sensors increases gradually to 1000 sensors. In each algorithmic run with n sensors, the univariate version PASAD is run n times and the processing time is added accordingly, whereas the multivariate version M-PASAD is run only once. As showcased in Fig. 6a and b respectively, the multivariate algorithm runs an *order of magnitude* faster during the training phase, and is consistently more efficient at computing the departure scores than the univariate algorithm in the detection phase.



(a) Detection of a `direct-damage` attack.

(b) Detection of a `stealth` attack.

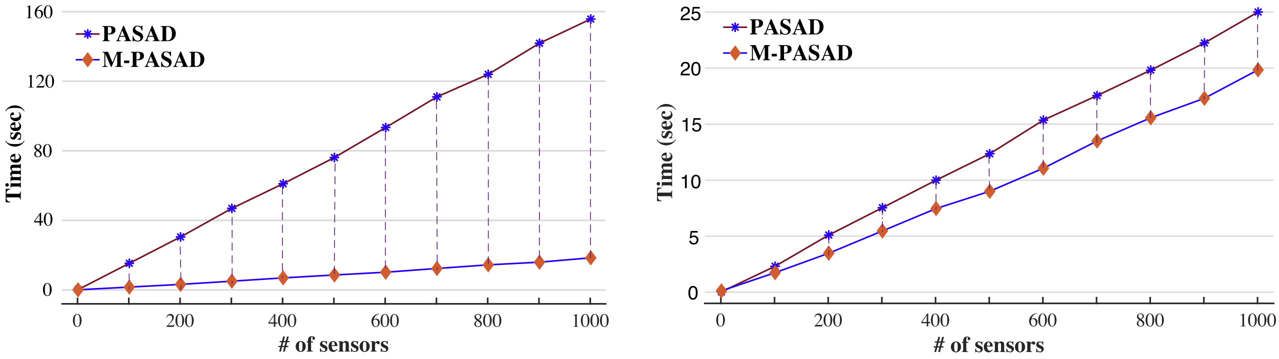
Fig. 4. M -PASAD detecting various attacks on the TE process by monitoring all sensors.



(a) TE sensors with subtle reaction to the attack.

(b) M-PASAD can detect stealthy attacks.

Fig. 5. The procedures for combining individual trajectory matrices into one common matrix and combining lagged vectors into an aggregate test vector do not undermine M-PASAD's capability of detecting subtle structural changes.



(a) Time-to-train w.r.t. number of sensors.

(b) Time-to-test w.r.t. number of sensors.

Fig. 6. Benchmarking M-PASAD against PASAD with respect to time-to-train (a) and time-to-test (b) as the number of sensors increases. M-PASAD exhibits an order of magnitude boost in training performance and is consistently faster during the detection phase.

In light of these results, we argue that M-PASAD scales well in industrial environments with ubiquitous sensing, while still preserving the detection accuracy of the original method.

6. Related work

The new generation of cyber-physical systems, equipped with advanced sensing and communication, form the backbone of the emerging smart industrial environments. Process-level attack detection is concerned with monitoring the triad of sensing, actuation, and control to identify implausible behavior in the physical process.

Efforts to develop attack-detection mechanisms suitable for the ubiquitous, complex, and heterogeneous cyber-physical systems are gaining traction in the control-engineering discipline as well as in the cyber-security community. Following is a non-exhaustive list of research works on detecting various kinds of cyberattacks in industrial environments. A more comprehensive account of related work can be found in the surveys [12–14].

State Estimation methods are frequently used in this domain [10,15–18], where state-space models are typically created from measurements and knowledge about the system to mathematically represent the physical process. Then, the difference between the estimated state and the actual state of the monitored system is analyzed to detect if the physical process is drifting from normal dynamics. *Statistical methods* employ different statistical means such as Auto-Regression [19], χ^2 statistic [20], and Kalman filters [21], to perform statistical tests on residuals in order to identify significant deviations that may be attributed to malicious acts. *Machine Learning and Data Mining methods*, which use machine learning techniques, e.g., LSTM Neural Networks [22,23], and Data Mining techniques, e.g., common path and clustering [24,25], trained on features extracted from process data to define a baseline behavior and thereafter detect anomalies.

Our approach is different from the related work in that it builds upon a model-free time-series based method that does not require models of the physical process as it learns the system dynamics purely from raw historical sensor measurements.

7. Conclusion

The forward-looking progression of industrial environments is heavily dependent on advanced sensing and communication capabilities, which renders critical-infrastructure more vulnerable to cyberattacks. A process-level attack-detection mechanism, named PASAD, which monitors sensors for malicious behavior, has recently been proposed in the literature. Being univariate, monitoring multiple sensors requires multiple instances of PASAD running concurrently, a fact that greatly limits its scalability. In this paper, we introduced M-PASAD, a multivariate extension of PASAD, that overcomes the mentioned limitation by adapting the underlying theory such that multiple sensors can be monitored concurrently by one instance of the algorithm. We showed that the proposed algorithm is consistently faster than the original one, and that it has a considerably smaller memory footprint. We also argued that since the extension is at the theoretical level, M-PASAD inherits all the benefits from the univariate version, making it equally capable of detecting subtle behavioral changes induced by stealthy attacks.

Declaration of Competing Interest

Authors declare that they have no conflict of interest.

CRediT authorship contribution statement

Wissam Aoudi: Conceptualization, Data curation, Formal analysis, Writing - original draft, Writing - review & editing. **Magnus Almgren:** Writing - original draft, Writing - review & editing.

References

[1] W. Aoudi, M. Iturbe, M. Almgren, Truth will out: departure-based process-level detection of stealthy attacks on control systems, in: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS '18, ACM, New York, NY, USA, 2018, doi:10.1145/3243734.3243781.

[2] N. Golyandina, A. Zhigljavsky, Singular Spectrum Analysis for Time Series, Springer Science & Business Media, 2013, doi:10.1007/978-3-642-34913-3.

[3] M. Almgren, W. Aoudi, R. Gustafsson, R. Krahl, A. Lindhé, The nuts and bolts of deploying process-level IDS in industrial control systems, in: Proceedings of the 4th Annual Industrial Control System Security Workshop, ICSS '18, ACM, New York, NY, USA, 2018, pp. 17–24, doi:10.1145/3295453.3295456.

[4] H. Hassani, R. Mahmoudvand, Multivariate singular spectrum analysis: a general view and new vector forecasting approach, Int. J. Energy Stat. 1 (01) (2013) 55–83.

[5] H. Hassani, R. Mahmoudvand, Multivariate Singular Spectrum Analysis, Palgrave Macmillan UK, London, pp. 49–86. 10.1057/978-1-137-40951-5_2

[6] W. Aoudi, A. Hellqvist, A. Overland, M. Almgren, A probe into process-level attack detection in industrial environments from a side-channel perspective, in: Proceedings of the Fifth Annual Industrial Control System Security (ICSS) Workshop, ICSS, Association for Computing Machinery, New York, NY, USA, 2019, pp. 1–10, doi:10.1145/3372318.3372320.

[7] M.S. Kemal, W. Aoudi, R.L. Olsen, M. Almgren, H.-P. Schwefel, Model-free detection of cyberattacks on voltage control in distribution grids, in: 2019 15th European Dependable Computing Conference (EDCC), IEEE, 2019, pp. 171–176.

[8] T. McEvoy, S. Wolthusen, A plant-wide industrial process control security problem, in: IFIP Advances in Information and Communication Technology, Springer Berlin Heidelberg, 2011, pp. 47–56, doi:10.1007/978-3-642-24864-1_4.

[9] T. Larsson, K. Hestetun, E. Hovland, S. Skogestad, Self-optimizing control of a large-scale plant: the tennessee eastman process, Ind. Eng. Chem. Res. 40 (22) (2001) 4889–4901, doi:10.1021/ie000586y.

[10] A.A. Cárdenas, S. Amin, Z.-S. Lin, Y.-L. Huang, C.-Y. Huang, S. Sastry, Attacks against process control systems: risk assessment, detection, and response, in: Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security, ASIACCS '11, ACM, New York, NY, USA, 2011, pp. 355–366, doi:10.1145/1966913.1966959.

[11] M. Krotofil, J. Larsen, D. Gollmann, The process matters: ensuring data veracity in cyber-physical systems, in: Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security, ASIA CCS '15, ACM, New York, NY, USA, 2015, pp. 133–144, doi:10.1145/2714576.2714599.

[12] D. Ding, Q.-L. Han, Y. Xiang, X. Ge, X.-M. Zhang, A survey on security control and attack detection for industrial cyber-physical systems, Neurocomputing 275 (2018) 1674–1683, doi:10.1016/j.neucom.2017.10.009. http://www.sciencedirect.com/science/article/pii/S0925231217316351

[13] D. Ramotsoela, A. Abu-Mahfouz, G. Hancke, A survey of anomaly detection in industrial wireless sensor networks with critical water system infrastructure as a case study, Sensors 18 (8) (2018), doi:10.3390/s18082491.

[14] J. Giraldo, D. Urbina, A. Cardenas, J. Valente, M. Faisal, J. Ruths, N.O. Tippenhauer, H. Sandberg, R. Candell, A survey of physics-based attack detection in cyber-physical systems, ACM Comput. Surv. 51 (4) (2018) 76:1–76:36, doi:10.1145/3203245.

[15] F. Pasqualetti, F. Dörfler, F. Bullo, Attack detection and identification in cyber-physical systems, IEEE Trans. Autom. Control 58 (11) (2013) 2715–2729, doi:10.1109/TAC.2013.2266831.

[16] D. Urbina, J. Giraldo, A. Cárdenas, N.O. Tippenhauer, J. Valente, M. Faisal, J. Ruths, R. Candell, H. Sandberg, Limiting the impact of stealthy attacks on industrial control systems, in: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, ACM, 2016.

[17] A. Cárdenas, S. Amin, B. Sinopoli, A. Giani, A. Perrig, S. Sastry, Challenges for Securing Cyber Physical Systems, Workshop on Future Directions in Cyber–Physical Systems Security, 2009.

[18] Y. Guan, X. Ge, Distributed attack detection and secure estimation of networked cyber-physical systems against false data injection attacks and jamming attacks, IEEE Trans. Signal Inf. Process. Netw. 4 (1) (2018) 48–59, doi:10.1109/TSIPN.2017.2749959.

[19] D. Hadžiosmanović, R. Sommer, E. Zambon, P.H. Hartel, Through the eye of the PLC: semantic security monitoring for industrial processes, in: Proceedings of the 30th Annual Computer Security Applications Conference, ACM, 2014.

[20] Y. Shoukry, P. Martin, Y. Yona, S. Diggavi, M. Srivastava, PyCRA: physical challenge-response authentication for active sensors under spoofing attacks, in: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, ACM, 2015.

[21] K. Manandhar, X. Cao, F. Hu, Y. Liu, Detection of faults and attacks including false data injection attack in smart grid using Kalman filter, IEEE Trans. Control Netw. Syst. 1 (4) (2014) 370–379, doi:10.1109/TCNS.2014.2357531.

[22] C. Feng, T. Li, D. Chana, Multi-level anomaly detection in industrial control systems via package signatures and LSTM networks, in: 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), IEEE, 2017.

[23] Y.-j. Xiao, W.-y. Xu, Z.-h. Jia, Z.-r. Ma, D.-l. Qi, NIPAD: a non-invasive power-based anomaly detection scheme for programmable logic controllers, Front. Inf. Technol. Electron. Eng. 18 (2017) 519–534.

[24] S. Pan, T. Morris, U. Adhikari, Developing a hybrid intrusion detection system using data mining for power systems, IEEE Trans. Smart Grid 6 (2015) 3104–3113.

[25] I. Kiss, B. Genge, P. Haller, A clustering-based approach to detect cyber attacks in process control systems, Industrial Informatics (INDIN), 2015.