SAPIENZA
UNIVERSITÀ DI ROMA

Sapienza University of Rome

Dipartimento di Ingegneria Informatica Automatica e Gestionale "Antonio Ruberti"
Master in Control Engineering

Thesis For The Degree Of Master

Model free online cyber-attack detection using PCA and Reinforcement learning

Relatori
Francesco Delli Priscoli
Abolghasem Daeichian

Correlatori
Andrea Tortorelli
Francesco Liberati

Candidate
Elham Honarvar
1915877

Academic Year 2022

Abstract

To ensure the CPS's safe and proper operation, attacks must be rapidly recognized, identified, and pinpointed, and fast action must be taken to defend the entire system. Due to the unpredictability of attacks, the first step in strengthening resilience during the attack period and/or post-attack phase is to detect cyberattacks successfully. Regarding Anomaly Detection Systems(ADS), there are certain challenges. First, to determine the system's typical behavior, the majority of cyber-attack detection approaches, such as state estimation methods, build a model based on the available data. The system then assesses whether or not the system's behavior is normal by comparing the estimated outputs of the model with the actual process outputs. In relation to the second problem, some intrusion detection techniques, such as Machine Learning (ML), can automatically build the model based on the training data set, which contains data instances that can be characterized using a set of attributes (features) and associated labels. Massive amounts of data—often referred to as "big data"—are needed but handling them is challenging. To tackle all these issues in this study, an online model-free algorithm is proposed to detect False Data Injection (FDI) and jamming attack on cyber-physical systems. Using Principal Component Analysis (PCA) in the observation space, the proposed method reconstructs expected observations in a reduced dimension space based on the most effective principle components. To begin with, the existence of an attack or normal operation is decided based on the measurement residual by using the Euclidean detector and the cosine-similarity metric detector. The proposed method has been evaluated by performing simulations on an IEEE-14 bus power system with 23 smart meters. In addition, the results have been compared with the model-based Kalman estimation method which shows the outperformance of the proposed method in terms of statistical measures of binary classification problems such as precision, recall, and F-score. To extend our research work we use the paradigm of model-free reinforcement learning (RL) to describe the online attack/anomaly detection problem as Partially Observable Markov Decision Process (POMDP) problem using the measurement provided by our new PCA-based technique. Numerical studies using Matlab show that the suggested RL-based algorithm is successful in detecting cyber-attacks on the smart grid.

Keywords: Anomaly detection, cyber attack, FDI attack, Jamming attack, PCA, Euclidean detector, Cosine similarity detector, RL, POMDP.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview and Motivation

The Cyber-Physical System (CPS) technologies are becoming increasingly significant since they ensure that a comfortable, healthy, secure, and safe environment is provided [103]. For instance, the so-called Industry 4.0 includes all devices covered by the Internet of Things (IoT) [5, 30]. An IoT system, as a typical CPS, is composed of two key components: the cyber space and the physical systems. Cyber space is a network of interconnected physical elements which is used to connect the cyber and physical systems. The communication over the cyber space makes CPS vulnerable to cyber-attacks despite its technological benefits [123]. Cyber attacks can significantly increase not only the operating costs but also the risk of system performance [97]. In addition to voiding the Confidentiality, Integrity, and Availability (CIA) trinity, successfully penetrated cyber-physical attacks can have a wide range of consequences. For example, on December 2015, half of the Ivano-Frankivsk Region in Ukraine was de-energized due to a hacker attack [18]. Another example is "2017 WannaCry ransomware Cyber Attack" in which around 200000 machines in more than 150 countries were compromised and cost the world almost 6 billion pounds [111].

To maintain the CPSs safe and appropriate operation, the attacks must be quickly detected, identified, and pinpointed, as well as take immediate action to defend the entire system [95]. Because of the unexpected nature of attacks, the first step in enhancing resilience in the attack phase and/or post-attack phase is to successfully detect cyber-attacks. As a result, several research projects using various methodologies have been conducted in the previous decade in order to successfully detect cyber-attacks. Early detection of cyber-attacks along with the acceptable accuracy is critical for effective response, which can be either making a modification or waiting for the next time period to collect more data [34].

Cyber attack defense mechanisms can be categorized as prevention, resilience or detection and isolation. Only authorized users should be able to interact with the system/network, hence prevention is required. Some prevention methods are cryptography [48], connection verification and signature detection [61], and randomization [3]. A system's ability to withstand, recover from, and adapt to a cyber attack is referred to as cyber resilience. Examples of resilience approaches are hardening [99], defensive islanding [51], energy storage [43], distributed generation [99] and network reconfiguration [13]. However, most cyber attack detection systems construct a model based on the available data to establish the system's normal behavior. Then, using a comparison between the model's estimated outputs and the actual process outputs, it determines whether the system's behavior is normal or not. Isolation of an attack refers to determining which element of the system is directly affected by the cyber attack. As a result of this isolation, decoupled models of the system are created which are only vulnerable to cyber attacks in a specific areas of the system [85]. Many detection methods can be found in the literature including Machine Learning [31], Artificial Intelligence [58], ensemble learning [89], state estimation [76], game theory [115], matrix separation approaches [65], deep learning [97], and reinforcement learning approaches [25, 62]. Learning techniques for detecting anomalies include both supervised and unsupervised methods. To build the predictive model, supervised methods require a labeled training set containing both normal and anomalous samples. Because they have access to more information, supervised methods should theoretically yield a higher detection rate than semi-supervised and unsupervised methods. However, there are several technical limitations that make these methods appear less accurate than they should be. The first difficulty is the scarcity of a comprehensive training data set. Furthermore, acquiring precise labels is difficult, and training sets frequently contain noises, resulting in greater false alarm rates [98]. State estimation approach is frequently used for attack detection [56]. An adversary can launch an intelligent attack by modifying sensor values within the system's operational limitations which can be tackled using an estimator like Kalman filter or other estimators. The Kalman filter attempts to track sensor values while also removing noise. Once convergence is reached, the estimator tracks the sensor's updated values, and the difference (residue) between measured and estimated values decreases. Although this method can be effective, the attack still impacting the plant's performance, using knowledge of the system's structure and functioning which is one of the drawbacks of model-based techniques.

Massive amounts of data, often referred to as a big data, are difficult to handle. In this respect, data can be translated into a lower-dimensional subspace by for example the spectral-based approaches such as the Principal Component Analysis (PCA). It is usually used to reduce the dimensionality of large data sets by transforming a large set of variables into a smaller one that retains the bulk of the information in the larger set without losing important traits. With the rise of high-dimensional datasets in CPS, the demand for sparse feature extraction has grown, and a number of approaches have been proposed in the literature. Considering CPS security, on both sides of attacker and defender, the PCA-based technique are employed [10]. On the attacker's side, PCA-based techniques are used to get beyond the defense barrier. On the other hand, PCA-based approaches are used on the defense side to construct quick and effective defense strategies. For instance, in [16, 120], the PCA is used to convert measured data into a new subspace while preserving as many spatial features as possible. Because of the Jacobian matrix's dimension, they recommend using the number of state variables as the number of primary components in the smart grid. It is possible to design FDI attacks based on the data of the new projected space using these methods. In [118], a PCA-based attack defense method has been proposed, which employs the fast gradient sign method, non-target attack method, and a white box assault to address the problem of machine learning security and defense against adversarial attacks. A blind FDI attack construction technique has been suggested in [120], based on the PCA. It has been demonstrated that using measurements based on its subspace information, the attacker can successfully and stealthily attack the system. However, it has two significant drawbacks. The accurate selection of the dimensionality parameter, which needs knowledge of the entire number of states in any system, is the first crucial parameter in the PCA-based approach. This approach cannot be employed as a zero-knowledge attack because an external attacker is uninformed of the whole amount of states in a system. Secondly, usually noiseless observations or random noise with a Gaussian distribution are assumed whereas the Gross errors (i.e., errors that do not follow a Gaussian distribution) are widespread in smart grid measurement data due to sensor failures or transmission issues. Theoretically, it has been proved that the existence of gross errors in the data affects the accuracy of PCA. Thus, to sum up, the detection methods which not only do not rely on training data but also reduce the size of data are more appreciated. [34, 73].

1.2 Contribution

This thesis presents an online model-free approach to cyber attack detection, specially FDI and jamming attacks on CPSs. We suppose that a cyber-attack is launched against the system at an unknown time τ, and our goal is to identify the attack as fast as possible after it occurs, given that the attacker's skills and strategies are unknown. The proposed approach tackle the curse of dimensionality problem and reconstructs the expected observations in a reduced dimension space that incorporates the principal components by applying the PCA onto the observation space. The presence of an attack is then determined using model-free Reinforcement Learning detection algorithm for POMDP, which eliminates the need for a system model. The goal of this challenge is to minimize both the average detection delay and the false alarm rate while increasing the precision and F-score. Moreover the results are compared to both the model-based approach using Kalman filter and the PCA-based one without any learning algorithm where the Euclidean detector and the cosine-similarity metric detector are performed on the measurement residual. Some of the key features of the proposed approach are:

- In order to deal with the dimensionality curse, PCA was adopted;
- No prior knowledge of the system is required because of the model-free nature of the system.
- We use projection rather than estimation space in this method.
- Using this approach, it is possible to detect a wide variety of attacks.
- A PCA-based algorithm outperforms a model-based technique, such as Kalman filtering, according to several performance evaluation methods.
- The capability of RL to compare the expected utility of available actions without a model of the environment is one of its strengths.
- Reinforcement Learning is a technique in which an agent learns how to solve a problem without a teacher.

1.3 Organization of the thesis

The next chapters of the thesis discusses the background, the literature review, and the contribution. In chapter 2 the structure of cyber physical systems and their security objectives are discussed in details. Then this chapter review the literature on cyber physical attacks and their classification based on different criteria and finally some identification and detection methods of cyber attacks are reviewd. In chapter 3, the system model, state estimation, and different attack scenarios are described. Moreover, two attack detectors are investigated; Chapter 4 introduces the proposed PCA-based algorithm and the proposed detection method(Reinforcement Learning); in chapter 5 simulations to validate the proposed solution are presented and discussed in detail; finally, chapter 6 draws the conclusions of the present thesis outlining future research lines.

Chapter 2

Background and literature review

2.0.1 Cyber Physical Systems

A new generation of networked systems known as Cyber-Physical System (CPS) combines the computation, communication, and control. The CPS technologies are becoming increasingly significant since they ensure that a comfortable, healthy, secure, and safe environment is provided [103]. For instance the so-called Industry 4.0 includes computerized vehicle, aircraft controls, wireless sensor networks, smart grids, and practically all devices covered by the Internet of Things [8, 30, 92]. To connect the cyber and physical worlds, cyber-physical systems use a network (cyber space) of interconnected physical elements such as sensors, actuators, robotics, and computing engines. These systems are intelligent, automated, and collaborative. Monitoring, Networking, Computing, and Actuation are the four fundamental processes in the typical operation of a Cyber-Physical System [38].

1. Monitoring: It is CPS's primary role which can assess previous actions and also provide feedback. Monitoring ensures that subsequent actions are carried out properly.

2. Networking: CPS sensors can generate real-time data, therefore networking deals with data aggregation and diffusion. Because the CPS incorporates a large number of these sensors, the data gathered can be aggregated or diffused for analysis by the analyzer. Moreover, networking communications allows many apps to engage at the same time.

3. Computing: It analyzes data obtained during monitoring and ensures that physical processes meet pre-determined criteria. If the criteria are not met, remedial actions are developed and implemented.

4. Actuation: It Carries out the activities that were determined during the computation step and activate a variety of actions, such as rectifying the CPS's online conduct and altering the physical process.

2.0.2 Cyber-Physical System configuration

A CPS can be developed on a large scale with many huge components, and it can have intricate connections between real-world physical systems and cyber-world control software. In Figure 2.1, a hierarchical CPS structure and its components can be seen.

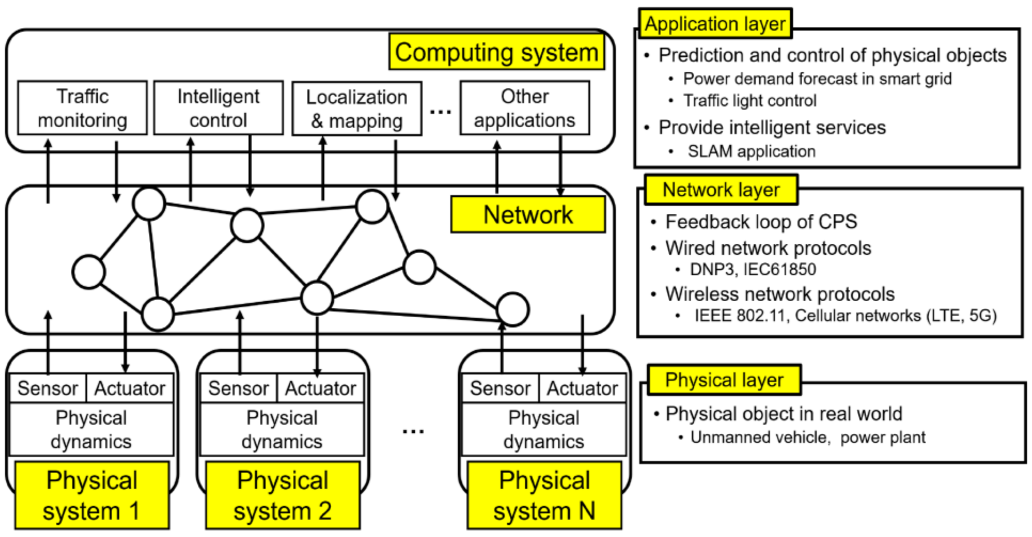

Figure 2.1: A hierarchical cyber-physical system structure[60]

1. Physical System Layer

   The physical system layer represents a number of physical systems, which are real-world items. In the physical system layer, several physical systems have sensors and actuators, with sensors reporting the states of physical items to the computing system in the cyber-world and actuators operating physical objects based on commands from the computing system. The physical system accomplishes the physical processes through sensing and actuating in the physical system. In a continuous time domain, the operation of the physical system is represented as system dynamics. We consider a single-input single-output (SISO) linear time-invariant (LTI) system to simplify the statement of physical dynamics:

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned} \tag{2.1}$$

   where x represents the physical system's state with n state variables, A denotes the system matrix, B represents the input matrix, u indicates the control input signal, and y provides the sensor measurement. The state transition by the physical system's characteristics is represented by the system matrix term $Ax(t)$, the input matrix term $Bu(t)$ represents the state transition caused by the control input signal $u(t)$, and the state measurement by the sensor attached to the physical system is represented by the output matrix term $Cx(t)$, where all physical state variables in the state vector $x(t)$ are not accessible. The physical systems provide a sensor measurement $y(t)$ to the computing system in order to report status information, and the computer system then sends the control input signal $u(t)$ to the physical systems, enabling them to operate as envisioned. The control input signal $u(t)$ on the computing system controls the actuation process, which governs the dynamics of physical systems. The following formula is used to calculate the control input signal $u(t)$:

$$\begin{aligned} \dot{\hat{x}}(t) &= A\hat{x}(t) + Bu(t) + l(y(t) - C\hat{x}(t) \\ u(t) &= -K\hat{x}(t) \end{aligned} \tag{2.2}$$

where $x(t)$ is the physical system's state estimation, L is the observer gain, and K is the controller gain . Based on the system model Equation 2.1, the computing system conducts state estimation and sensor measurement $y(t)$ with an observer because sensors cannot indicate the entire state of physical systems. The state feedback controller with controller gain K calculates the control input signal based on the state estimation in Equation 2.2. The controller gain K and the observer gain L are assumed well-designed to maintain the physical systems stability. The power plants and electric equipment that physically transmit electricity are components of the physical system layer in a smart grid.

2. Network layer

   The network layer manages communication between numerous physical systems in the real world and computational systems in the cyber realm. A network in CPS, allows the computing system to remotely control numerous physical systems, making the CPS more advantageous than traditional point-to-point control systems in terms of system configuration cost and adaptability of system administration. The network exchanges sensor measurements from physical systems and control input signals from the application layer. Physical systems can malfunction when a network mistake occurs, such as missing control-related data and/or a long transmission delay. As a result, a network that creates a feedback control loop between physical systems and computational systems must ensure high-level data transmission reliability. The distributed network protocol (e.g., DNP3) connects the numerous power plants with the centralized control system in a power grid.

3. Application Layer

   In the cyber-world, the application layer demonstrates the computing system with which software performs intelligent operations. Based on sensor data from physical systems, several CPS applications are implemented, and the applications manage physical systems, anticipate the state of physical systems in the following time step, and give intelligent services to CPS users. In the next time step, a computing system determines the system dynamics Equation 2.1 based on the results of the application execution. Because of the power constraints, the physical system is reliant on the computing system to perform intelligent operations that necessitate complicated computations. The application layer in most CPSs is expected to have sufficient computational capacity and no electrical power constraints. Furthermore, because of real-time communication in the network layer, power-limited hardware devices can operate more intelligently than typical embedded systems. A supervision system in a smart grid predicts power consumption for entire areas in real time based on sensor measurements from a large number of smart watt-hour meters in the physical system layer and past power consumption time-series data. As a result, power demand forecasts can be used to feedback management of electricity generation at power plants.

2.0.3 Security objective of Cyber Physical Systems

As multiple recent instances have highlighted, one of the primary concerns in CPS is security. The communication over a cyber space makes CPSs vulnerable to cyber-attacks [123]. Cyber attacks can significantly increase not only the operating costs but also the risk of system performance. Cyber attacks can typically cause three types of damage to a business or its data, irrespective of their objectives and methods. Cyber attacks compromise confidentiality by stealing data, integrity by altering data, and availability by denying access to data, services, and systems. Some cyber attacks may combine two or more of these types, but these three cyberattack types comprise the majority of malicious cyber activities[37]. For example, "the 2007 Estonia Cyber Attack" during which around 58 Estonian websites, including government, banking, and media services, going offline [29]. On December 2015, half of the Ivano-Frankivsk Region in Ukrain was de-energized due to a hacker attack. The power disruptions caused by the Ukraine attacks are the first publicly confirmed events[18]. Another example is "2017 WannaCry ransomware Cyber Attack" in which around 200000 machines in more than 150 countries were compromised and cost the world almost 6 billion pounds [111]. All vulnerable aspects into which the attack can enter are detailed in the following sections. As mentioned earlier, cyber threats affect different criteria including Confidentiality, Integrity and Availability(CIA). Here is a more specific categories [38, 107]:

- **Confidentiality**: The ability of a system to protect and avoid the disclosure of information to unauthorized users or systems from the outside. In other words, confidentiality is interpreted as restricted access to information.
- **Integrity and Authentity**: The protection of any data or resources against unauthorized alterations. Integrity is a guarantee that information is reliable and accurate. Authenticity can also be included in this category. It assures the data, transactions, and communications are authentic.
- **Availability and Reliability**: Availability is a guarantee of reliable access to the information by authorized people. The system (all components of the CPS, including physical components, communication channels, and processors) must be available to fulfill its mission. Availability required for every party involved to be validated.
- **Accountability**: To ensure exact accountability, a combination of auditability and authentication must be provided. Auditability refers to a comprehensive examination of all current and historical system behaviors. It is primarily concerned with identifying causes of system failures and determining the scope of defects or the consequences of a security event. For example smart grid can be traceable and recordable because of auditing and accessibility. This security requirement is pertinent for determining liability[45].

2.0.4 Different Cyber physical attacks

A cyber physical attack is considered as manipulation of CPS components that results in a physical system malfunction and process, such as the divergence of state $x(t)$ in system dynamics. Because the CPS combines the computing system, networks, and physical systems, if one of the CPS components is compromised, the states of all physical systems become unstable. In contrast to traditional embedded systems, the CPS is particularly vulnerable due to network connectivity, which allows an attacker to harm computing systems, networks, and physical systems. Furthermore, CPSs are vulnerable due to a lack of effective CPS protections in terms of design, setup, and operation[60].

1. **Physical system layer attacks**

- **Sensor attack** A sensor attack is described as a network-based modification of sensor measurement $y(k)$, which is illustrated by adding a sensor-attack signal to the valid sensor measurement as follows:

$$\tilde{y}(t) = Cx(t) + y^a(t) \tag{2.3}$$

where $y(t)$ is the sensor measurement that is modified and $y^a(t)$ is the sensor-attack signal of the attacker. An estimation mistake produces calculation flaws in the control input signal $u(t)$ for the actuating process since deceiving the computing system that is conducting state estimation is the objective of a sensor attack. Then, in the physical system Equation 2.1, a fault control input signal in a sensor attack might create physical system failures, such as divergence in the physical system's state trajectory.

- **Controller attack**
A controller attack is described as a change to a network's control input signal $u(t)$, which is expressed by adding the controller attack signal to the legitimate control input signal, as follows:

$$\dot{\tilde{x}}(t) = A\tilde{x}(t) + B(u(t) + u^a(t)) \tag{2.4}$$

where $\tilde{x}(t)$ represents the physical system's attacked state and $u^a(t)$ is the malicious user's control attack signal. By introducing an unexpected control input signal u, a controller attack attempts to disrupt the physical state $x(t)$. Because both the computing system's control input signal $u(t)$ and the physical dynamics, $x(t)$ are influenced by the controller attack signal $u^a(t)$, when a controller assault is conducted, the physical system does not perform as planned by the computational system. To prevent detection by traditional detectors that are model-based, advanced controller attacks are used. These kind of controller attacks, in particular, take use of zero-dynamics , which is a unique characteristic of the sensor measurement $y(t) = 0$ in relation to system dynamics. The so-called Zero Dynamics Attack (ZDA) is described as an attack on the control input signal u(t) and the starting physical state $x(0)$ in several cyber-physical security studies. The ZDA is designed to target either the physical system's unstable zero-dynamics or the unstable zero-dynamics induced by discretization. Internal physical state $x(t)$ under ZDA attack diverges to infinity; however, due to a zero-dynamics characteristic, this divergence is not apparent in sensor measurement $y(t)$ that is called stealthy.

- **Combined attack**
A combined attack is defined as modifying a sensor measurement and the control input signal at the same time, which is represented as:

$$\begin{aligned} \dot{x}(t) &= Ax(t) + B(ut + u^a(t)) \\ \tilde{y}(t) &= Cx(t) + y^a(t) \end{aligned}$$

A combination attack, as opposed to a sensor-only or controller-only attack, requires eavesdropping on both the sensor measurement $y(t)$ and the control attack signal $u(t)$ channels; thus, a combined attack is difficult to conduct successfully. To put it another way, because most combined assaults are more complicated than sensor-only and controller-only attacks, detection tactics for combined attacks necessitate additional computational resources. One of the most common combined attacks is the covert attack, in which the attacker has complete knowledge of the system dynamics and uses this knowledge for generating both a sensor signal $y^a(t)$ and a control input signal $u^a(t)$ contaminated with attack. The attacker steals the real sensor measurement $y(t)$ from the physical system and deceives the computational system by sending the created sensor measurement y with perfect system information across the sensor measurement transmission channel. A malicious control input signal, $u = u(t) + u^a(t)$, generated with perfect system information and the legitimate sensor measurement signal $y(t)$ from the sensor measurement transmission channel, controls the physical system as the attacker aims to attack the control input signal transmission channel.

2. **Network layer attacks**
   The network layer is in charge of sharing a range of data with high levels of dependability and real-time features between application and physical system layer. The attacker can disrupt data transmission at the network layer, which violates data integrity and real-time limitations on the CPS, as a result of which the physical system becomes unstable. The three most important cyber-physical attacks in the network layer are denial of service attack, flooding attack and packet manipulation. Although network layer attacks do not appear to have a direct relationship with physical dynamics, they can have a major impact on the physical system's stability.

   - **Denial of Service attack**
     A DoS attack is defined as the disruption of the transmission of control-related data, such as control input signals and sensor measurements, by particular network attacks. As a result, from the standpoint of control theory, a DoS attack is characterized as a discrete-time signal drop, which has the same mean as a network disruption attack in the cyber security area. Considering the discritized physical dynamics as below:

$$\begin{aligned} x(k+1) &= A_d x(k) + B_d u(k) \\ y(k) &= C_d x(k) \end{aligned} \tag{2.5}$$

     Because of the missed packets, the control input signal cannot be updated when a DoS attack is performed against the network:

$$x(k+1) = A_d x(k) + B_d u(k-1) \tag{2.6}$$

     The physical system's control performance is hampered by intermittent packet drops caused by a DoS attack. Furthermore, if a DoS attack is undertaken, by producing a state $x(k)$ divergence, cause the physical system to become unstable.

   - **Flooding Attack**
     Flooding is defined as the intentional exhaustion of network resources, such as network bandwidth or network device memory, through the generation of huge amounts of network

traffic. When an attacker initiates a flooding assault, the enormous amount of packets generated denies legal communication nodes transmission possibilities or causes network equipment to run out of memory. Transmission delays for genuine packets grow as a result of a flooding assault, violating the CPS's real-time limitations. In order to succeed with a flooding attack, the attacker needs merely generate a large amount of data over the network quickly, and advanced network understanding is not required. In control theory, time-varying delays in a network are recognized to have a negative impact on the physical system's stability and control performance. The following equation depicts the influence of network delay on physical dynamics:

$$x(k+1) = e^{AT}x(k) + \sum_{j=0}^{\bar{d}-d} \int_{T-t_{j+1}^k}^{T-t_j^k} e^{As} ds Bu(k+j-\bar{d}) \tag{2.7}$$

where T is the physical system's sensor measurement sampling period, t_j^k is the j-th time instance at time step k and d and $\bar{d}$ are$[\frac{\tau_{max}}{T}]$and $[\frac{\tau_{min}}{T}]$ respectively, where τ_{max} and τ_{min} are the maximum and minimum time-varying network delay bounds. When a flooding attack is conducted against a network, physical systems are destabilized or switch into a fail-safe mode that stops or restricts the physical system's operation to ensure safety, similar to a DoS assault.

- **Packet Manipulation**
  Packet manipulation considered as the alteration of the header or payload of packets carried over the network. It is divided into two phases: packet stealing and packet modification. Using flaws in the network protocol, CPS implementation, and communication nodes, the attacker gains access to the network and deceives communication nodes, including physical systems and computing systems. After that, the attacker receives the packets on the network. After stealing packets from a valid source node, the attacker alters the header or payload of the packets and sends the modified packets to the original destination. When the attacker tries to influence the error-checking field in the header during the modification process, such as the checksum under the user datagram protocol, the destination node dismisses the manipulated packet because it determines that the received packet has an error due to the modified checksum (UDP). As a result, the packet manipulation attack can potentially be used as a DoS attack. In other words, if the attacker adjusts the payload while strictly complying to network protocols, the packet modification deceives the destination nodes. This packet manipulation attack has the same effect as physical layer attacks when sensor measurements or control input signals are changed.

3. **Application layer attacks**
   The application layer, which delivers intelligent services, benefits CPS users and compute performance-limited physical devices. The attacker can intrude into the computing system via the input/output (I/O) interfaces, such as the serial communication port or Network Interface Card (NIC), and gain access to essential computing components, such as file systems, cache memory, and process schedulers, from which with the aim of disrupting the computing system the attacker can launch attacks. The larger and more complicated the computing

system becomes as the application layer becomes more sophisticated; nevertheless, due to the computing system's complexity, it is difficult to prevent an attack that targets the system. Two different forms of application layer attacks are:

- **Application Software Attack**
  An application software attack causes the physical system layer to return faulty results for service requests, resulting in defective services for CPS users and false instructions for compute resource-constrained physical systems. When improper operational directives are injected into physical systems, the physical system is controlled incorrectly, which is something the CPS user does not expect. From the standpoint of physical dynamics, an application software attack is viewed as a damage in an actuation process. Under this type of attack, memory modification related to the state estimation process and the control input calculation process involves manipulating the observer gain L and controller gain K, respectively. Even though the network layer and physical system layer are valid, the physical state $x(t)$ diverges to infinity when the observer gain L and controller gain K in the actuation process are replaced to stabilize physical dynamics.
- **Computing Hardware Attack**
  A computing hardware attack is described as the breakdown of a computing system component, such as the power supply, dynamic random access memory (DRAM), the CPU, or storage systems, either directly or indirectly. The two types of computing hardware attacks that are classified, are a computer system-down attack and a data modification attack in memory via unauthorized accesses. The repercussions of the attacks on the physical system determine the criteria for computing hardware attack classification. A system crash caused by a computational systemdown assault is the same as a DoS attack in terms of physical dynamics. When a malicious power-off disrupts the computing system, the control application software stops as well, and the physical system terminates as well.

2.0.5 Famous cyber physical system attacks

1. **Zero dynamic attacks**
   Zero-Dynamic attacks are defined as a group of techniques that use unstable zeros as the bug to target smart meters. The enemies pose as the service providers' unpredictable zero dynamics. The falsified output will then be sent across the communication channel. As a result, the real state becomes more real as time passes, nearing the output-nulling space. So, the result is extremely near to zero, which is also known as a stealthy[28]. In [108] the set of open-loop stealthy attacks is investigated. The attack is open-loop in the sense that it is built without the usage of any online data. As a result, the attack policy is described in terms of the a priori information accessible, namely the system's dynamical model. Because a property of the system known as zero-dynamics is proved to describe this class of attacks, we refer to it as the class of zero-dynamics attacks. The system under a zero-dynamics attack is characterized as an autonomous dynamical system with a particular initial state using a geometric control framework.
2. **FDI Attack**

In False Data Injection attack, malicious data is inserted into the measurements of a subset of meters in an additive manner. In practice, an FDI attack can be carried out by tampering with network communication channels or hacking smart grid meters and/or control centers. During these attacks, the attacker injects erroneous data or measures into the system in attempt to change the system's status measurements.

$$y_t = Hx_t + w_t + b_t \| \{t \geq \tau\} \tag{2.8}$$

b_t denoted the injected malicious data

3. **Jamming Attack**
   In the event of a jamming attack, it is assumed that the attacker continuously produces Additive White Gaussian Noise (AWGN) to the network communication channels in order to compromise a subset of meter readings. AWGN jamming is considered, because (i) it is a widely used jamming model in the literature, (ii) it is for implementation, and (iii) in an additive noise channel with Gaussian input, the Gaussian noise maximizes the mean squared error of estimating the channel input given the channel output for a given mean and variance, among all noise distributions. As a result, an attacker can use AWGN to jam communication channels in order to cause the most harm to the state estimation method.

$$y_t = Hx_t + w_t + u_t \| \{t \geq \tau\} \tag{2.9}$$

   u_t denotes the random noise

4. **Hybrid Attack**
   In a hybrid (mixed) attack, FDI and jamming attacks are launched at the same time against the system.

$$y_t = \begin{cases} Hx_t + w_t, & \text{if } t < \tau \\ \bar{H}x_t + w_t + b_t + u_t, & \text{if } t \geq \tau \end{cases} \tag{2.10}$$

   Note that the analyzed hybrid FDI/jamming attacks span all feasible data attacks if the noise terms in normal system operation are AWGN and the jamming noise terms are mutually independent over the meters. This is because the mean and variance of a Gaussian random variable define it, and the mean and variance of the density of meter measurements may be modified arbitrarily using hybrid attacks. In the event of a DoS attack, for example, meter readings are suppressed, and the control center only receives a random or zero signal. As a result, a DoS attack can be thought of as a subset of hybrid cyber-attacks, i.e., a DoS attack can be equivalent to an FDI attack with false data of the same magnitude as the actual signal but with the opposite sign, or a jamming attack with high level noise variances, allowing the actual signal to be ignored in favor of the noise signal. On the other hand, if the jamming noise is correlated across meters or is not regularly distributed, the attack does not fit the jamming attack model under consideration. A non-parametric goodness-of-fit test is used as a countermeasure in such instances.

5. **Replay attacks**
   The adversaries are enticed to launch replay attacks in order to steal energy or destroy the end

devices physically. A replay attack, unlike FDI attacks, will keep uploading the stealthy data in the end devices several times within a given time frame. Before starting replay attacks, it is necessary to hack PUMs. For a period of time, the opponents will listen in on the data. The enemies will then keep repeating the same facts in order to trick the central authority. Due to the restricted capabilities of cryptography operations in real-world circumstances, replay attacks are difficult to detect. As a result, the adversaries may create a wormhole tunnel or communication channel between the end devices, allowing replay attacks to take place (Ding et al., 2018). These assaults will cause a lack of real data in the central authority, causing routing services to be disrupted. An attacker, for example, can see and process the genuine data collected in order to execute replay assaults by repeatedly employing a forged signal in the system.

6. **Denial of Service (DoS) attacks and time synchronization attacks**
   These are two common types of smart grid attacks. Availability is traditionally defined as "ensuring timely and dependable access to and use of information," which is the target of Denial-of-Service (DoS) attacks. be included in the term as well. In this sense, we broaden the concept of DoS attacks in the context of the Smart Grid. DoS attacks in the SG, we feel, demand a more fine-grained and holistic description that includes the following dimensions:

   - Denial-of-Service attacks on availability.
   - Denial-of-Control attacks on computing, communications, or the power itself.
   - DoS attacks on data integrity (such as misleading state estimation and situational awareness).
   - Denial-of-Electric-Service attacks even when sufficient power is available.

   Any of these DoS attacks in for example Smart Grid domain could result in a cascading blackout, leaving hundreds, if not millions, of customers without power over a lengthy period of time.

2.0.6 Cyber threats classification

Various criteria are used to classify cyber-physical attacks. This section provides a comprehensive overview of the many classifications for cyber physical attacks.

Classification of cyber-attacks based on objective

Successfully penetrated cyber-physical attacks can have a wide range of effects in addition to voiding the CIA (i.e., Confidentiality, Integrity, and Availability) trinity.

1. Confidentiality

   Attack method:

   An adversary modifies data from particular meters and network switches in order to deceive the control center with a false network topology while evading discovery[7, 84].

   Countermeasures:

- Instead of utilizing the same key to encrypt every message, dynamically update the secret key that the smart meter shares with the data concentrator unit.
- change the key each time a message is encrypted.
- perform a system reset and/or examine the system configurations on a regular basis to remove any current malicious attack attempts.

2. Integrity and Authenticity

   Attack method:

   - The information from the smart meter is tinkered with, replaced, or eliminated before it is relayed to the smart meter provider. Attackers can employ false data injections to trick smart meters into reporting higher or lower electricity usage on a network[84].
   - Challenging privacy attacks aimed at obtaining unauthorized data from the system[1, 23, 75, 79, 125].

   Countermeasures:

   - Create a secret key pair for the sender and receiver (implementing more sophisticated authentication protocols).
   - PKI Standards for Smart Grids: PKI is a strong tool for providing secure authentication and permission for security association (SA) and key establishment.
   - Trust Anchor Security: Each RP (any device that authenticates a second party using the second party's certificate) must have safe means for loading and storing the root of trust or trust anchor (TA).
   - Developing a lightweight authenticated key agreement technique for smart grids that takes into account cutting-edge hardware to imitate smart meters.
   - Using various cryptographic features to provide an authentication architecture that prevents many of the threats.
   - Displaying a formal verification for an IoT-based smart grid authentication layout (ProVerif).

3. Availability and Reliability

   Attack method:

   - A denial-of-service attack that prevents utility providers or owners from accessing information stored in smart meters. When an attacker wants to buy some time to plan their next attack, they can turn off the smart meter device's availability, making it momentarily unavailable to receive or send signals from and to the utility provider. Attackers would turn off the smart meter device, jam the communication line, and even deploy a denial-of-service assault to do this.
   - The central authority's operation is halted due to a data flow obstruction. This prevents the data transmission network from being available to push the system over the brink[6, 41, 42, 66, 67, 84, 132].

Countermeasures:

- Replace the tampered-with smart meter with a new one.
- Change the message broadcast frequency on the channel.
- Activate security mode to the smart meter device that corresponds to the ZigBee standard.
- update the secret keys that are used to encrypt information on a regular basis.
- Four types of interdependencies are outlined, and a new state mapping model is proposed to relate cyber network failures to power network failures. Furthermore, two optimization models are introduced to optimize data connectivity in the cyber network and decrease load shedding in the power network in order to evaluate the influence of direct cyber-power interdependencies on the reliability indices.
- The indirect interdependencies between cyber and power networks is considered. A state updating-based model is suggested to quantitatively evaluate the dependability of cyber-power networks under indirect interdependencies, and certain applications of indirect interdependencies in modern power systems are described. To maximize data communication in the cyber network while minimizing load curtailment in the power network, two optimization models are applied.
- For current substation protection systems, a unique reliability modeling and analysis technique is presented. A typical IEC 61850-based substation protection system is created and studied, including both cyber and physical components such as Merging Units (MUs), Intelligent Electronic Devices (IEDs), and process bus. Individual component failure modes and their consequences for the entire system are modeled and numerically examined. The cyber-physical interaction matrix is also shown to be useful in the reliability study of a composite power system.
- The Roy Billinton Test System (RBTS) has been expanded to integrate new architecture substation protection systems, which is a key step as a test system.

4. Accountability

Attack method:

Smart meters, in most cases, will record the cost of energy, which will be added in the next bill. If the meters are taken over by criminals, the data may be tampered with, and the bills may be misleading [71, 72].

Countermeasures:

- Accountability systems will assign a number of witnesses to watch the observation object's activity. Those witnesses will provide pertinent observation evidence to back their findings after deviant behavior is detected. These proofs are usually irrefutable and thus reliable.
- Implement periodic audits and logging of the Smart Grid information system to ensure that the security mechanisms included during Smart Grid information system validation testing are still installed and operating correctly.

Classification of cyber-attacks based on Defense mechanism

To maintain the CPSs safe and appropriate operation, the attacks must be quickly detected, identified, and pinpointed, as well as take immediate action to defend the entire system. This is referred to as cyber resilience [57, 95]. Owing to the attack's unexpected nature, the initial step of resilience augmentation in the attack phase and/or post-attack phase is to successfully detect cyber-attacks. As a result, several research projects using various methodologies have been conducted in the previous decade in order to successfully detect and identify cyber-attacks as part of increasing the CPS's resilience. Due to the fact that any failure or anomaly in one component of it can cause significant damage to the entire system rapidly, early detection of cyber-attacks is vital for effective reaction which can be either makes a modification or waits for the following time period to collect more data. In general, the fast detection along with the acceptable detection accuracy is desired [34, 73]. Cyber attack defense mechanism can be categorized as prevention, resilience or detection and isolation. A schematic of defense mechanism can be seen in Figure 2.2.

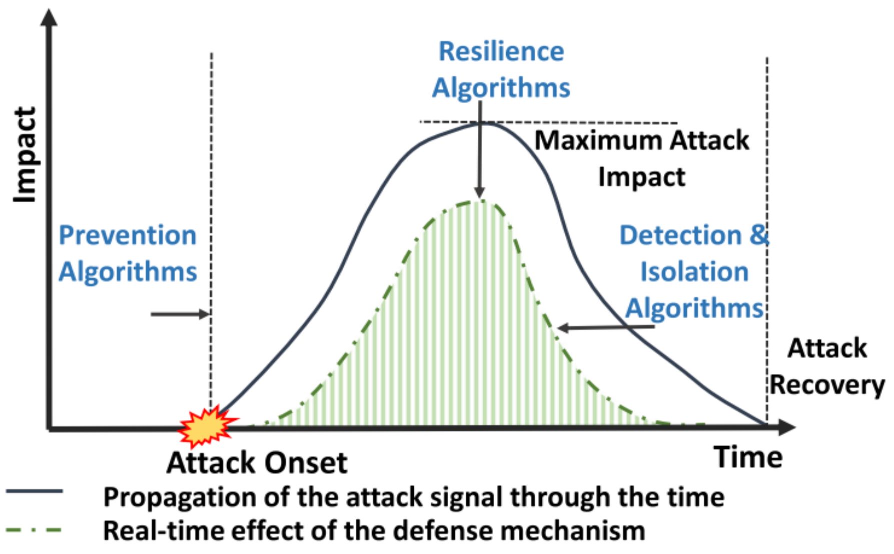
Impact
Resilience Algorithms
Maximum Attack Impact
Prevention Algorithms
Detection & Isolation Algorithms
Attack Recovery
Attack Onset
Time
Propagation of the attack signal through the time
Real-time effect of the defense mechanism

Figure 2.2: A schematic of defense mechanism[35]

1. Prevention
   Only authorized users should be able to interact with the system/network, hence prevention is required.Some prevention methods are cryptography, Connection verification and signature detection and randomization.

   - Cryptography

     Attack method:

     – A dynamic data encryption mechanism

     – More complex authentication techniques are being implemented includes [48, 53, 132, 132]:

       ∗ Symmetric Key

       ∗ Asymmetric Key

       ∗ Secure Hash Standard

    * Message Authentication
    * Key Management
    * Deterministic Random Number Generators
    * Nondeterministic Random Number Generators

- Connection verification and signature detection

  Attack method:

  Using this mechanism, a node will be able to authenticate the identity of another node (whether it is a legitimate or attacker node). If a node is identified as an attacker, other nodes will drop packets sent from that node.

- Randomization

  Attack method:

  - An attack detection model is proposed for power system based on machine learning that can be trained by using information and logs collected by phasor measurement units (PMUs). feature extraction is done, and then send the data to different machine learning models, in which random forest is chosen as the basic classifier of AdaBoost.
  - Based on an extremely randomized trees technique and kernel principal component analysis for dimensionality reduction, a unique approach to cyber-attack detection has been developed [3, 113].

- Proactive defense strategies

  Attack method:

  Defenders can take proactive steps to counteract the adversary's asymmetric advantage and improve security [26].

2. Resilience

   Cyber resilience refers to a system's ability to withstand, recover from, or adapt to a cyber attack. Some resilience approaches including Q-learning, hardening, defensive islanding, energy storage, distributed generation and network reconfiguration.

   - Q-learning

     Attack method:

     The WAPS wide-area damping control problem is solved using the Q-learning method: enabling wide-area damping controllers to achieve superior damping performance in the presence of large time delays and packet loss [39] .

   - Corrective Dispatch Scheme

     Attack method:

     The suggested solution approach locates the global optimum by using a finite number of Benders, such as Cuts, and the Lagrange multipliers can be configured as binary variables to provide an efficient procedure [21].

   - Hardening

     Attack method:

  - A restoration problem is established, in which the operator gradually reconnects the disturbed components over several periods in order to restore the DN's nominal performance.
  - As a basic metric for resilience, the suggested architecture uses a similar measure of availability. It shows how the ideas of resilience and dependency are inextricably linked and provides a measurement for the degree of functional dependency of loads on power networks [64, 99].

- Defensive islanding

  Attack method:

  - Proactive islanding and RCS (remote-controlled switch)-based fast fault isolation and service restoration are all taken into account to increase distribution systems' ability to withstand and recover quickly from extreme occurrences.
  - A resilience-oriented optimization technique is given, taking into account practical islanding in regular operation and critical load survivability during an emergency [51, 65].

- Fuel agent dispatch

  Attack method:

  The suggested paradigm uses an equivalent measure of availability as a baseline metric for resilience, as well as resistance and brittleness, to define additional crucial resilience-related concepts and metrics [64].

- Energy storage

  Attack method:

  The degree of functional dependency of loads on power networks is measured, and the concepts of resilience and dependency are shown to be inextricably linked [43, 64]
  .

- Distributed generation

  Attack method:

  - Using failure probability or the N-k contingency restriction to define numerous impact zones.
  - Assess the effectiveness of deploying a rapid response involving microgrid operations and DER dispatch in the aftermath of a disruption event involving strategic compromising of several DN components using microgrid operations and DER dispatch.
  - An autonomous load restoration architecture based on the IEC 61850-8-1 GOOSE communication protocol in active power distribution grids to improve feeder-level resilience [20, 54, 88, 99, 121].

- Network reconfiguration

  Attack method:

  A methodology has been presented for quantifying resiliency and maintaining power supply to critical loads (CLs) during extreme situations [13].

- Load control

  Attack method:

  In the aftermath of a disruption event including strategic compromising of numerous DN components, assess the usefulness of establishing a rapid response employing micro-grid operations and DER dispatch [99].

- Random behavior

  Attack method:

  - k percent of all network nodes are randomly labeled as hacked.
  - Simulator for WSN [40].

- SDN and virtualization

  Attack method:

  Mininet-based testing framework linked with ns-3 network simulator with Day Light SDN controller[11].

- New deployment

  Attack method:

  - Graph theory and clustering: In the context of smart metering, a graph-based analytical model is used to determine the required transmission power and number of gateways for wireless enabled mesh networks. Clustering is used to assign each smart meter to a gateway, and inter-cluster resilience addresses node capabilities to connect to other gateways in the event of a primary gateway failure.
  - To consider dependencies between ICT and measurement layers, a graph method is used.
  - The importance of each communication link and measurement unit is determined using the degree of centrality, and the resilience metric is defined as the deviation from ideal importance values [93, 96].

- Optimization approach

  Attack method:

  An optimal solution for assigning the feeder routing issue and substation facilities at the same time, as well as finding models of installed conductors and economic hardening of power lines in the event of unforeseen physical attacks on essential urban operational infrastructure To answer the problem of developing an optimal distribution network scheme (ODNS) and optimal resilient distribution network scheme, the grey wolf optimization (GWO) algorithm is used to determine the values of distribution networks (ORDNS)[44].

3. Detection and isolation

   This section summarizes some recent research in the topic of CPS attack detection and isolation. Both data-driven and estimation methods are included in these performed researches.

   - ML-based algorithms

     Classification, regression, and clustering are three types of machine learning techniques. Each of these areas contains a variety of methods.

Methods:

  - SVM-based, ANN, KNN, Naïve Bayes, AdaBoost.ANN and ARIMA, Density Cluster, MLR, NN and Fuzzy C-Means and K-means[9, 27, 31, 31, 70, 78, 80, 94, 104, 113, 129, 130].

- Ensemble learning

  Methods:

  - XGBoost, GBTD, Isolation Forest [4, 15, 87, 89]

- Randomized-tree-based

  Methods:

  - (KPCA)-based method [3]

- Artificial Intelligence-Based Algorithm

  Methods:

  - ANN and ELM [58]

- Deep learning

  Methods:

  - CNN and Encryption, MFEFD (Multi-task Feature Extracting Fraud Fetector), Deep autoencoder, GAN, RNN and LSTM, State Vector Estimator (SVE) and a Deep-Learning Based Identification (DLBI) scheme[24, 47, 50, 69, 114, 119]

- Reinforcement Learning

  Methods:

  - POMDP[25].

- Cloud computing

  Methods:

  - Computing capability of cloud, Cloud-based firewall[17, 36]

- State Estimation

  Methods:

  - ML, WLS, KF, EKF[2, 10, 19, 76, 81, 116]

- Matrix separation approach

  Methods:

  - Augmented lagrangian, low-rank matrix factorization, double-noise dual problem[68]

- BMS Algorithm

  Methods:

  - greedy algorithm[102]

- Game theory

  Methods:

  - Game theory and Bargaining game[46, 52, 115]

Classification of cyber-attacks based on attack model

1. Disclosure

   - Eavesdropping

     Attack objective:
     Confidentiality

     Attack effect:
     The data on a smart meter can be read and examined remotely with ease[132].

     Countermeasures:

     – Introducing a data encryption method to secure significant data is one strategy to protect an attack from eavesdropping.
     – Another method of preventing eavesdropping is to manually program bogus data into a smart meter as a decoy. When an attacker listens in on a smart meter, he will be reading a variety of data.
     – To safeguard sensitive data, implement a dynamic data encryption mechanism. Changing the encryption key on a regular basis would make it more secure while also making it more difficult for an attacker to spot trends in the key.

2. Deception

   - Replay attack

     Attack objective:
     Confidentiality,Integrity

     Attack effect:

     – The adversary must (a) capture and analyze data transferred between appliances and smart meters in order to obtain the customer's power use characteristics, and (b) construct and inject fraudulent control signals into the system in order to obtain the customer's power usage characteristics (A replay attack is carried out by stealing information that has been transmitted and using the stolen information to carry out falsification).
     – The lack of true data in the central authority will result in the routing services being disrupted as a result of these attacks.
     – The following incursion strategy will be used: First, it infiltrates the system and simply records the system's observed values over a period of time without altering any data; second, it invades the system and simply records the system's observed values over a length of time without altering any data;Then, using the present observed values as a starting point, begin to alter them. Observed values that had previously been recorded [52, 128].

     Countermeasures:

     – To prevent erroneous data injection, secure ports that have been validated and encrypted could be introduced. All other ports that aren't in use can be shut down.

    - A special process/algorithm might be implemented to check if the data is legitimate, providing an additional layer of safety.
    - Another option is to include a random increment in the original control input, which will raise the value of the state estimation fluctuation. If it is subjected to a replay assault, we can detect the attack by looking for abnormal fluctuations in the recorded value.

- FDI attack

  Attack objective:

  Confidentiality, Integrity

  Attack effect:

    - Such attacks target measurement and monitoring sub-systems with the aim of manipulating meter and phasor measurements, so as to misguide the operation and control of the utility provider[22, 33, 47, 91, 109, 124, 127].

  Countermeasures:

    - A State Vector Estimator (SVE) and a Deep-Learning Based Identification (DLBI) technique make up one detection mechanism. SVE assesses the accuracy of real-time measurement data by computing the l2-norm of measurement residual and comparing the result to a specified threshold. If larger than threshold and the measurement is considered compromised data, SVE raises an attack alarm.
    - Make the opponent unaware of the susceptibility of interconnected transmission lines that cover all buses (e.g., by proactively perturbing transmission-line susceptibility via a distributed flexible AC transmission system).
    - Smart meters and service providers can use a blockchain-based secure message transfer technique.

3. Disruption

- DoS attack

  Attack objective:

  Availability

  Attack effect:

    - An attacker is inserting their packets into the smart grid system's packet flow until the system can't manage or receive any more packets and fails[17, 36].

  Countermeasures:

    - Set up a reverse proxy system to filter incoming packets that need to be dropped. All random miscellaneous packets sent to overload the traffic will be divided into fractions and forwarded through multiple proxies utilizing this proxy technology, lessening the "flooding" effects on the principal control system.
    - While boosting bandwidth won't stop a DoS assault, it will give you more time to respond.
    - Because a DoS attack targets the smart grid's IP address, encrypting the IP addresses of smart grid devices may help to mitigate the attack. The IP addresses

can be kept anonymous, unlisted, and encrypted so that only the administrator has access to them. Another option is to use secure encrypted communications if possible, so that attackers cannot find the IP address through public channels.
    - Cloud-based firewall
    - Increased cloud computing capability

- Jamming attack

  Attack objective:

  Availability

  Attack effect:

    - The basic goal of jamming is to prevent the smart meter from communicating with the provider. To disrupt the wireless medium, noise signals are used. There are two types of jamming techniques:
        * Jamming by an attacker using his own device to send noise signals to obstruct the wireless signal is known as proactive jamming.
        * Reactive jamming occurs when a jammer first listens in on a wireless signal and then attempts to disrupt it only after determining that the signal is being sent across open space[82].

  Countermeasures:

    - Utilizing the TDMA protocol to encrypt link-layer packets, making it incredibly difficult for an adversary to jam them.
    - Channel surfing, encryption of link-layer packets, spatial retreat, and the TDMA protocol are some of the countermeasures for reactive jamming attacks.
    - The Ant system is another popular anti-jamming solution. Packets are transported across random nodes until they reach their destination using this mechanism.
    - Use virtual machines to increase the system's defenses against jamming. This solution would run all of the programs on a virtual computer in a separate network, with no access to the physical hardware.

- Zero dynamic attack

  Attack effect:

    - It's an actuator attack in which the system states are pushed along the zero dynamics. No sign appears in the output due to the inherent nature of zero dynamics, making this attack covert[59, 77, 100, 108, 122].

  Countermeasures:

    - Deploy a mechanism of intermittent output sampling in addition to periodic sampling.
    - Altering the null-space by switching through a set of topologies such as pause, update and resume attack before (after) topology switching to avoid detection.
    - Relocate any unstable zero to a stable region in the stage of digital implementation through modified digital samplers and holders
    - Modifying the system's structure

Summary of cyber-attack classification

Table 2.1: Summary of classification of cyber-attacks based on attack objective

References	Objective
[7, 84, 84]	**Confidentiality**
[1, 23, 75, 79, 125]	**Integrity and Authenticity**
[6, 41, 42, 66, 67, 84, 132]	**Availability and Reliability**
[71, 72]	**Accountability**

Table 2.2: Summary of classification of cyber-attacks based on attack model

References	Attack model
[132]	**Disclosure**: -Eavesdropping
[52, 128],[22, 33, 47, 91, 109, 124, 127]	**Deception**: -Replay attack -FDI attack
[17, 36, 82],[59, 77, 100, 108, 122]	**Disruption**: -DoS attack -Jamming attack -Zero dynamic attack

2.0.7 Most recent methods for identification and detection of cyber attacks

Machine learning, deep neural networks, and state estimation have gotten a lot of attention in recent years for detecting and mitigating physical system threats [56, 101]. The technique of inferring the values of a system's state variables using a limited quantity of measurable data at specific points in the system is known as State Estimation (SE). As a result, SE is essentially a numerical method for mapping data measurements to state variables [32]. An adversary can launch an intelligent attack by modifying sensor values within the system's operational limitations which can be tackled using an estimator like Kalman filter or other estimators. Kalman filter try to track sensor measurements and remove the noise. The estimator tracks the sensor's modified values, and the difference (residue) between measured and estimated values diminishes once convergence is achieved. Although this method can be effective, the attack still impacting the plant's performance, using knowledge of the system's structure and functioning which is one of the drawbacks of model-based techniques. Machine learning approaches that allow creating and maintaining anomaly detection systems (ADS) with less human interaction appear to be the only viable route to achieving the next generation of intrusion detection systems as the complexity and number of diverse attacks grows. Machine learning algorithms for intrusion detection can create the model automatically based on the training data set, which contains data instances that can be described using a set of qualities (features) and labels. Different types of attributes, such as categorical and continuous, can be used. All anomaly-based network intrusion detection systems (A-NIDS) techniques have a similar architecture. The following core modules or steps are present in all of these methods: parameterization, training, and detection. Collecting raw data from a monitoring environment is part of parameterization. The raw data should be reflective of the modeled system (e.g. Packet data from a network). During the

Table 2.3: Summary of classification of cyber-attacks based on defence mechanism

References	Defense mechanism
[3, 26, 48, 53, 113, 132, 132]	**Prevention**: -Cryptography -Connection verification and signature detection, -Proactive defense strategies -Randomization
[21, 39, 44, 99],[51, 64, 64, 65],[11, 13, 40, 43, 64, 99] [20, 54, 88, 93, 96, 99, 121]	**Resilience**: -Q-learning -Corrective Dispatch Scheme -Hardening -Defensive islanding -Fuel agent dispatch Energy storage -Distributed generation -Network reconfiguration -Load control -Random behavior -Optimization -SDN and virtualization -New deployment
[27, 78, 94, 113], [31, 31, 70, 104, 130], [9, 80, 129]	**Detection and isolation**: -ML-based algorithm -AI-based algorithm -RL -Cloud computing -Ensemble learning -Game theory -State estimation -Deep learning -Matrix separation approach -BMS Algorithm

training step, conventional or automatic approaches are used to model the system. Depending on the technique employed, the behaviors depicted in the model will differ. Detection compares the system that was created during the training step to the parameterized data component that was chosen. To identify abnormal data instances, threshold criteria will be chosen. Machine learning can automatically create the appropriate model based on some given training data. The availability of the necessary training data, or the ease with which it may be obtained when compared to the effort required to define the model manually, is one rationale for this approach. Techniques for detecting anomalies include both supervised and unsupervised methods. To build the predictive model, supervised methods (also known as classification methods) required a labeled training set containing both normal and anomalous samples. Because they have access to more information, supervised methods should theoretically yield a higher detection rate than semi-supervised and unsupervised methods. However, there are several technical limitations that make these methods appear less accurate than they should be. The first difficulty is the scarcity of a comprehensive training data set. Furthermore, acquiring precise labels is difficult, and training sets frequently contain sounds, resulting in greater false alarm rates [83, 98, 110]. So, the methods which not only do not rely on training data but also reduce the size of data are more appreciated. Dealing with massive amounts of data, often known as big data, can be tough at times. So, data can be translated into a lower-dimensional subspace where normal occurrences and anomalies appear different, which is the assumption for spectral-based approaches. The Principal Component Analysis (PCA) approach is one of the examples of spectral-based approaches. It is used to reduce the dimensionality of large data sets by transforming a large set of variables into a smaller set that retains the bulk of the information in the larger set. The goal is to find an orthogonal projection of the data that maximizes the variation observed. With the rise of high-dimensional datasets, the demand for sparse feature extraction has grown, and a number of approaches have been proposed in the literature. PCA techniques are used on both sides of the net, on both the attacker and defender sides. On the attacker's side, PCA techniques are used to get beyond the defense barrier. On the other hand, PCA approaches are used on the defense side to construct quick and effective defense strategies.

In [16, 120] PCA is used to transform measurement data into a new subspace while keeping as many spatial properties as possible. The dimensionality reduction is when the number of variables is reduced to less than the number of original variables. Because of the Jacobian matrix's dimension, they recommend using the number of state variables as the number of primary components in smart grid. It is possible to design FDI attacks based on the data of the new projected space using these methods. In [118] a PCA-based attack defense method has been proposed, which employs the fast gradient sign method (FGSM), non-target attack method, and a white box assault to address the problem of machine learning security and defense against adversarial attacks. The PCA has been performed on the MNIST dataset to protect deep neural network models from escape attacks. A blind FDI attack construction technique has been suggested in [120], based on the PCA. It has been proved to successfully and silently attack the system utilizing measurements based on its subspace information. Yet, it has two major disadvantages. The right selection of the dimensionality parameter, which requires knowledge of the entire number of states of any system, is the first crucial parameter in the PCA-based technique (rank of Jacobian matrix). This approach cannot be employed as a zero-knowledge attack because the whole number of states of a system is unknown to an outside attacker. Second, these data-driven attack tactics assume noiseless observations or random noise with a Gaussian distribution. Gross errors (e.g., errors that do not follow a Gaussian distribution) are common in smart grid measurement data due to sensor failures or transmission issues, just as they are in other application fields (e.g., image processing, bio-informatics). Theoretically, it has been demonstrated that the accuracy of PCA is influenced by the presence of gross errors in the data.

Chapter 3

Preliminaries and problem formulation

3.1 System model

The dynamic that governs the sampled data from the input and output of a dynamical system can be generally represented by a discrete time-varying nonlinear system:

$$\begin{aligned} x_{t+1} &= F\left(x_t, u_t, v_t, t\right) \\ y_t &= G\left(x_t, u_t, w_t, t\right) \end{aligned} \tag{3.1}$$

where $x_t \in \mathcal{X} \subset \mathbb{R}^n$, $u_t \in \mathcal{U} \subset \mathbb{R}^m$, and $y_t \in \mathcal{Y} \subset \mathbb{R}^q$ are the state vector, the input, and the output of the system at time step t, respectively. v_t and w_t are stationary or time-varying process and measurement noises. The state and output noises are typically independent Gaussian additive noises as $v_t \sim \mathcal{N}(0, \sigma_v^2)$ and $w_t \sim \mathcal{N}(0, \sigma_w^2)$. $F : (\mathcal{X}, \mathcal{U}, \mathbb{R}) \to \mathcal{X}$ and $G : (\mathcal{X}, \mathcal{U}, \mathbb{R}) \to \mathcal{Y}$ are vector-valued nonlinear analytic functions with static real values. Theoretical modeling of common practical CPSs to obtain the system model Eq.(3.1) is very challenging. Even if it is assumed that the system model is available, estimating the states of the system with an acceptable error is not straightforward and has a lot of computational burden. Also, linearization techniques such as considering the first order terms of Taylor expansion or the Linear Parameter Varying (LPV) models may not ease the problem. As a result, data-driven approaches are more interested.

3.2 Attack scenarios

Unlike prior embedded systems, the CPS is particularly susceptible due to network access, allowing an attacker to damage computing systems, networks, and physical systems [106]. As the CPS includes the computing system, networks, and physical systems, if one CPS component is attacked, the states of all physical systems become unstable. A deliberate attack can target application, network, or physical layers in a typical architecture of CPS as it is shown in Figure 3.1 . Penetration to the application and network layers can be prevented by the well developed authentication and authorization methods. However, handling physical layer attacks are more challenging.

The control signal u_t and the sensor observation y_t are exchanged by the system and controller through the network. So, these two forms of control-related data are susceptible to modification by an adversary with network access, resulting in a x_t deviation from the desired physical state.

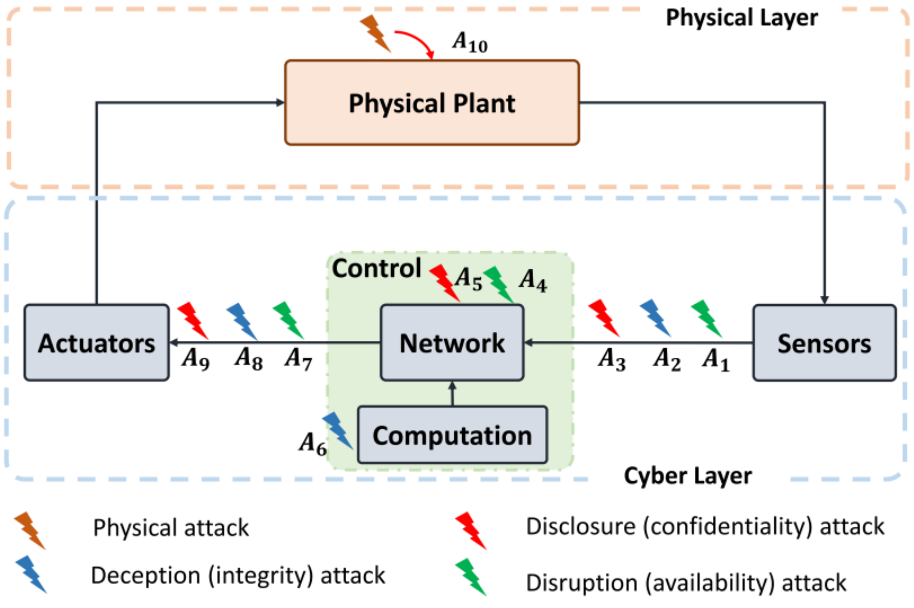

Figure 3.1: A schematic of various attacks[35]

Assuming a sensor attack, the attacker affects the sensors observations by the attack signal and delivers the contaminated signal to the controller. This action can be write down as follows:

$$\tilde{y}_t = H(y_t, y_t^a) \tag{3.2}$$

where $\tilde{y}_t$ is the contaminated sensor measurement and y_t^a is the sensor-attack signal at time step t. False Data Injection (FDI) and jamming attacks are considered as sensor-attacks [63], which corrupt the real measurement values by the adversarial attack signal additively.

FDI Attack

Attacks that use deception or misleading data injection are achieved when the signals deviate in some way from their actual value. They can appear in three different places in the closed-loop procedure (see Figure 3.1). In terms of the amount of damage they may inflict, deception attacks are the most powerful. For instance, it is simple to envision how the closed-loop system could be severely destabilized by a deception attack [14, 35].

- Attacks on sensors that alter the operating environment and destroy the accuracy of the measurements (A2).
- Attacks on actuators that cause the control signals to deviate from the required values (A8).
- Computational values attacks that change the control law in some CPSs (A6).

In general an FDI attack can be fulfilled in practice by hacking the system meters and/or control centers or altering network communication channels. In this thesis as the observation space is considered so our focus will be on sensor attacks. In this kind of FDI attack, malicious data is added to the measurements of a selection of meters in an additive manner [62, 126]. Considering an

FDI attack at time step τ, the data received from the network by the controller can be represented as follows:

$$\tilde{y}_t = y_t + y_t^{\mathrm{FDI}} 1_{t \geq \tau}, \tag{3.3}$$

where y_t^{FDI} denotes the adversary false data at time t and $1_{t \geq \tau}$ is a indicator function with value 1 when its condition $t \geq \tau$ is satisfied and 0 otherwise. To be more stealthy, the malicious data corrupts the observations for $t \geq \tau$ randomly. So, the attack signal can be generated by product of a random process $\mathcal{R}$ and a pseudo-random binary sequence $\mathcal{B}$, i.e. $y_t^{\mathrm{FDI}} = \mathcal{R}.\mathcal{B}$ where the binary process $\mathcal{B}$ takes value 1 with the probability of $0 < p < 1$ and zero otherwise at each time step.

Jamming Attack

Disruption attacks, also known as denial of service (DoS) attacks or jamming attacks considered as any deliberate alteration of data. Jamming may occur on the actuation signals (A7), the underlying network (A4), or the sensor data (A1). In a jamming attack, the attacker continuously send noisy data n_t, for example white Gaussian noise, to the network communication channels and corrupt the observation by that additive noise [117]. Thus, a jamming attack at time step τ can be represented as follows:

$$\tilde{y}_t = y_t + y_t^{\mathrm{Jam}} 1_{t \geq \tau} \tag{3.4}$$

where y_t^{Jam} denotes the jamming noise.

Hybrid Attack

When a hybrid (mixed) attack occurs, both FDI and jamming attacks are launched simultaneously on the system, resulting in the following received data model:

$$\tilde{y}_t = y_t + (y_t^{\mathrm{FDI}} + y_t^{\mathrm{Jam}}) 1_{t \geq \tau} \tag{3.5}$$

In this case, both the articulated FDI attack and Jamming noise are targeted the system.

3.3 Cyber attack detection

We consider the 5-tuple $\{\mathcal{X}, \mathcal{Y}, \mathcal{S}, \mathcal{O}, \mathcal{A}\}$. Here, $\mathcal{X}$ and $\mathcal{Y}$ are the set of possible system state and measurement. $\mathcal{S}$ is the CPS state including the system and cyber situation. $\mathcal{X}$, $\mathcal{Y}$, and $\mathcal{S}$ are not available to the attack detector. $\mathcal{O} \subset \mathbb{R}^q$ stands for the data received from the cyber space, which may be contaminated by the attacker. In addition, the set of possible outcomes (actions space) of the attack detector is denoted by $\mathcal{A}$ which includes but not limited to the normal and attacked situation. For example, someone may also define situations that include the type of attack. Any cyber attack detection algorithm is expected to provide a mapping π from $\mathcal{O}$ to $\mathcal{A}$, i.e. $\pi : \mathcal{O} \to \mathcal{A}$.

If the action space consists of only two possible actions, that is to say normal and attacked situations, then the possible events are shown in Figure 3.2. Some attacks can be detected truly (TA: True Alarm) while some others may be missed (MD: Miss Detection). In addition, some false alarms (FA) can be given. In this case, the proposed attack detection algorithm can be evaluated

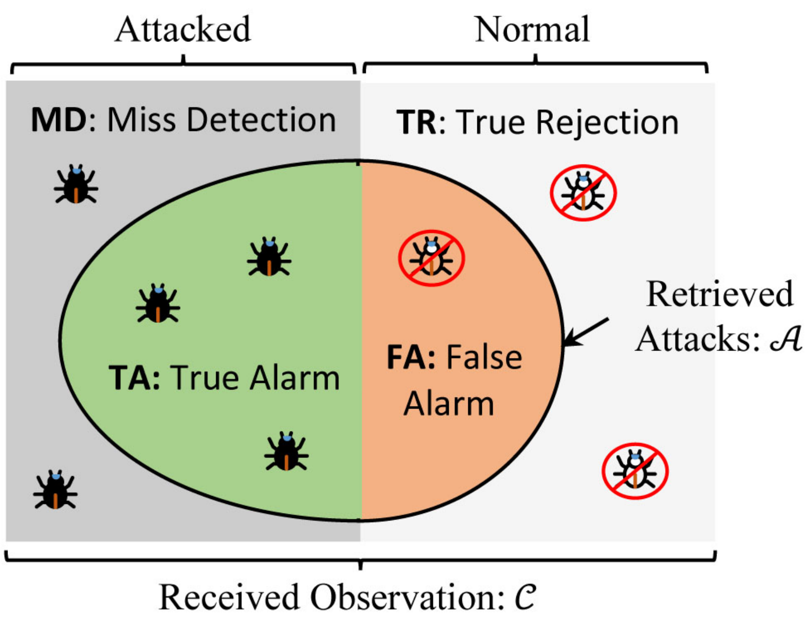

Figure 3.2: A schematic of possible situation/action

by the measures of binary classification problems using Monte-Carlo simulation as follows:

$$\begin{aligned}
\text{Precision} &= \text{TA}/(\text{TA}+\text{FA}) \\
\text{Recall} &= \text{TA}/(\text{TA}+\text{MD}) \\
\text{F-score} &= \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision}+\text{Recall}} \\
\text{MDR} &= \text{MD}/(\text{TA}+\text{MD}) \\
\text{PrFA} &= \text{FA}/(\text{TA}+\text{FA}) \\
\text{AvDD} &= \frac{1}{\text{TD}} \sum_{i=1}^{\text{TD}} (t_i^d - \tau_i)
\end{aligned}$$

where t_i^d and τ_i are the detection and actual attack time for the i-th Monte-Carlo episode, respectively. Also, MDR, PrFA, and AvDD denote miss detection ratio, probability of false alarm, and average detection delay, respectively. In conclusion, a detection algorithm with higher F-score and lower AvDD is desired.

3.3.1 Kalman based attack detector

One of the most commonly utilized techniques for distinguishing between a cyber intrusion and normal operation is the use of state estimators. Considering Gaussian process and measurement noise in a discrete-time linear dynamical system, the Kalman filter is an optimal state estimator in the sense of error covariance matrix. In a Kalman based attack detector, firstly, the states of the system are estimated by the Kalman filter as follows

$$\begin{aligned}
\hat{x}_{t|t-1} &= A\hat{x}_{t-1|t-1} \\
P_{t|t-1} &= AP_{t-1|t-1}A^T + \sigma_v^2 I_N \\
K_t &= P_{t|t-1}H^T(HP_{t|t-1}H^T + \sigma_w^2 I_K)^{-1} \\
\hat{x}_{t|t} &= \hat{x}_{t|t-1} + K_t(y_t - H\hat{x}_{t|t-1}) \\
P_{t|t} &= P_{t|t-1} - K_t H P_{t|t-1},
\end{aligned} \tag{3.6}$$

where P is the estimated state covariance matrix. Secondly, the expected output $\hat{y}_t$ is calculated. Thirdly, a measure of the expected output $\hat{y}_t$ and the received data $\tilde{y}_t$ is calculated. For example, the Euclidean distance

$$d(\tilde{y}_t, \hat{y}_t) = \sqrt{(\tilde{y}_t - \hat{y}_t)^2} \tag{3.7}$$

or the cosine similarity

$$cos(\theta) = \frac{\tilde{y}_t . \hat{y}_t}{||\tilde{y}_t|| ||\hat{y}_t||} \tag{3.8}$$

where $||.||$ stands for the norm. Finally, if the considered measure exceeds a predefined threshold, an attack is detected. Normally, the threshold should be determined by considering the normal operation of the system and with a trade-off between miss detection and false alarms. One major disadvantage of this approach is the need to a precise system model which is challenging. Another drawback of this method is that the quality of decision making is strongly rely on the selected value for the threshold. However, we will compare the proposed method with the conventional Kalman based detector.

3.3.2 PCA-based attack detector

In general, model extraction for CPSs with several plants, sensors, and actuators is difficult, if not impossible. So, model-free data-driven attack detection approaches possess an inherent advantage that does not require the usage of a system model. In order to compare the performance of the proposed method with another model-free approach, we present a recently developed PCA-based algorithm for attack detection in this subsection [49]. The PCA-based attack detection algorithm is founded on the intuition that the principal components of observations before and after attack would be different.

The PCA for a sliding window of length N of time-series received observations, $Y_t = [\tilde{y}_{t-N}, \cdots, \tilde{y}_t] \in R^{q\times N} \in \mathcal{O}$ fits a q-dimensional ellipsoid to the data, where each axis representing a main component. To obtain the axes of the ellipsoid, a covariance matrix of observations must be computed:

$$\text{cov}(Y_t) = \mathbf{E}[(Y_t - \mathbf{E}[Y_t])(Y_t - \mathbf{E}[Y_t])^T] \tag{3.9}$$

where $\mathbf{E}$ denotes the expectation. The observation can be reconstructed by utilizing the most effective principal components with the least reconstruction error. When the CPS is targeted by an attack at time τ, there is a transient time that the sliding window contains observation samples both from the pre- and post-attack, i.e. $Y_t = [\tilde{y}_{t-N}, \cdots, \tilde{y}_\tau, \cdots, \tilde{y}_t]$. Hence, it is expected that the difference between the measured value and the reconstructed value by the PCA rises in the transient time. Finally, the decision about the existence of an attack is made by comparing the Euclidean distance or the cosine-similarity of the reconstructed value and the measured value by a pre-specified threshold.

Chapter 4

Online model-free Attack detection

4.1 Proposed Online model-free Attack detection

4.1.1 PCA-based Detection method

During the first step of the attack detection algorithm, the principal components of time-series data Y_t, namely $\mathcal{P}_t$, are computed based on the new observations received by the PCA. The PCA can be interpreted as fitting a q-dimensional ellipsoid to the data that every axis represents a primary component. Firstly, the covariance matrix of observations must be constructed to discover the ellipsoid's axes:

$$\text{cov}(Y_t) \quad = \quad \text{E}[(Y_t - \text{E}[Y_t])(Y_t - \text{E}[Y_t])^T] \tag{4.1}$$

where E denotes the expectation. Secondly, the normalized eigenvectors and the corresponding eigenvalues of $\text{cov}(Y_t)$ should be computed. The eigenvectors can be interpreted as the axis of the ellipsoid fitted to the data. Thirdly, the principal components of data are sorted by importance based on the corresponding eigenvalues and the most effective ones are selected which is denoted by $\hat{\mathcal{P}}_t$. Then, the observation is reconstructed by using the reduced principal components $\hat{\mathcal{P}}_t$ which is denoted by $\check{y}_t$. Finally, whether or not an attack occurred is determined by comparing $d(\tilde{y}_t, \check{y}_t)$ with a specific threshold, where $d(.)$ can be a measure function.

Euclidean-based detectors consider the Euclidean distance as a decision criteria, i.e.

$$d(\tilde{y}_t, \check{y}_t) = \sqrt{(\tilde{y}_t - \check{y}_t)^2}. \tag{4.2}$$

An attack is identified when the deviation exceeds the threshold value. The threshold is set so that it is more than the value of Euclidean distance in normal conditions. Cosine similarity is another measure which quantify the similarity between two non-zero vectors of an inner product space. It is equal to the cosine of the angle between two vector, which is the same as the inner product of the vectors which are normalized to have the length of one. Given two vectors $\tilde{y}_t$ and $\check{y}_t$, the cosine similarity is represented as follows:

$$cos(\theta) = \frac{\tilde{y}_t . \check{y}_t}{\|\tilde{y}_t\| \|\check{y}_t\|} \tag{4.3}$$

The resulting similarity scale runs from -1 to 1, with 0 representing orthogonality or decorrelation and in-between values indicating moderate similarity or dissimilarity.

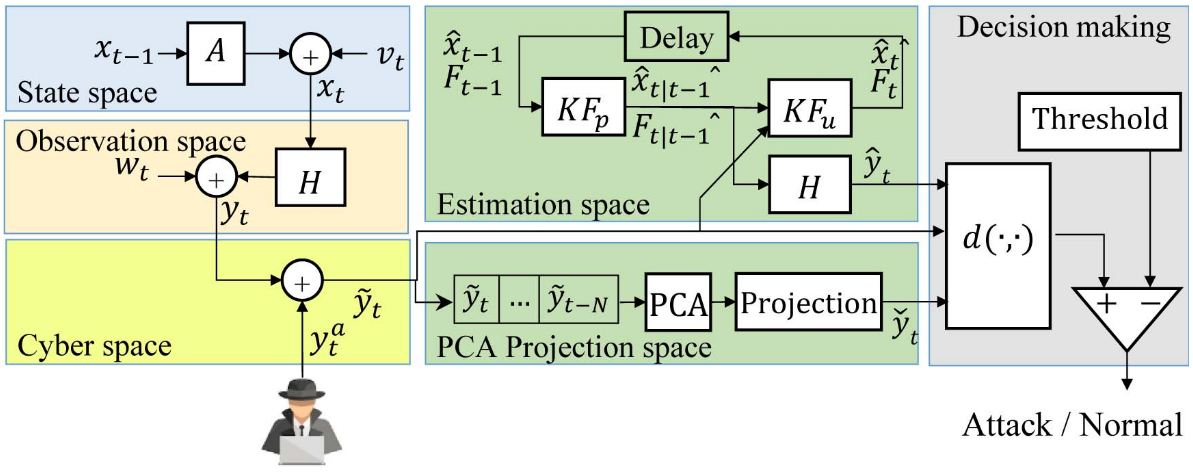

Figure 4.1: A schematic of data model

Figure 4.1 depicts the functional architecture of the proposed method. The Euclidean distance and cosine similarity are employed as the measure function in the Decision making block. Moreover, the proposed algorithm is summarised in Algorithm 1.

```
Algorithm 1: Online model-free attack detection
On each time step:
1: Push the received observation to the time-series data stack $Y_t$.
2: Calculate the normalized principal components of $Y_t$, namely $\mathcal{P}_t$.
3: Select the most effective components, namely $\hat{\mathcal{P}}_t$.
4: Reconstruct the observation $\check{y}_t$ using the reduced principal components.
5: Calculate the measure function $d(\tilde{y}_t, \check{y}_t)$.
6: Decide on attack based on the measure function output.
```

4.1.2 POMDP and Reinforcement Learning

Reinforcement learning is an artificial intelligence approach that emphasizes individual learning from interactions with their environment to create optimal behavior[90]. The majority of RL literature assumes that the environment can be described as a Markov Decision Process (MDP) with a fully observable Markovian state development. It has been demonstrated that a number of exploration–exploitation algorithms give high performance guarantees for MDPs. In fact, however, the premise of complete observability of the state development is frequently violated, and the agent may have only noisy views of the true state of the environment. In this situation, the partially-observable MDP or POMDP model is more applicable [12, 55]. Unlike the MDP which maps the underlying states to the actions, the policy function of POMDP maps the history of observations to the actions. The solution to a POMDP yields a sequence of optimal actions of the agent for interacting with its environment such that maximizes the expected rewards.

Formally, the POMDP problem is defined by the 7-tuple $\{\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \mathcal{O}, \Omega, \gamma\}$, where there are a set of states $\mathcal{S}$, a set of actions $\mathcal{A}$, a set of conditional probability of transition between states

$\mathcal{T}$, the reward function $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, a set of observations $\mathcal{O}$, a set of conditional observation probabilities Ω, and the discount factor $\gamma \in [0, 1]$. The environment is in a hidden state $s_{t-1} \in \mathcal{S}$ and the agent takes an action $a_{t-1} \in \mathcal{A}$. The action a_{t-1} causes the environment evolves to state s_t with probability $\mathcal{T}(s_t|s_{t-1}, a_{t-1})$. Simultaneously, an observation $o_t \in \mathcal{O}$ with probability $\Omega(o_t|s_t)$ is received by the agent. In addition, a reward (cost) $r_t = \mathcal{R}(s_{t-1}, a_{t-1})$ is obtained by the agent. Then, the process repeats. The aim of agent is to choose an action $\pi_t : \mathcal{O} \to \mathcal{A}$ at each time step t to maximize the received rewards or minimize the total cost. Considering the latter, the goal of the agent can be written as follows:

$$\min_{\pi_t} \mathbf{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t\right] \tag{4.4}$$

Now, the attack detection problem can be embedded into the POMDP framework, see figure 4.2. A simple case has been given in [62], however more general situation is introduced in this paper. Assuming that an attack of type i can launch to the system each at unknown time $\tau_i, i = 1, \cdots, \kappa$. The attacking policy and the attacker strategy are hidden from the defender. The defender makes every effort to identify the attack as quickly as possible while minimizing the false alarm. Assuming κ different types of attacks and a normal state (NS), a POMDP problem with $\kappa + 1$ unknown states can be defined, i.e. $\mathcal{S} : \{s_t^1, \cdots, s_t^\kappa, s_t^{NS}\}$. It worth noting that a combination of attacks can be considered as a new type of attack. The observation $\tilde{y}_t$ is received from the cyber space, and the defender should select one action from η actions including normal state and declaring attack of type $j = 1, \cdots, \eta$, i.e. $\mathcal{A} : \{a_t^1, \cdots, a_t^\eta, a_t^{NS}\}$. The conditional probability of transition between states and the conditional observation probability are unknown due to the fact that attack start time and the attacker strategy are unknown by the attack detector. The problem is regulated through adjusting the rewards $r_{i,j}$ which means selecting action j at state i. Considering normal state to be the true underlying state of the system, the correct detection is that the attack detector give out the normal operation while any other decision is a false alarm. Also, if the system is under attack, declaring a normal operation by the defender leads to more detection delay or miss detection. Finally, if it is desired that the attack-type of i results in action of j, then, any other defender action than a^j is a wrong type detected attack. The true detection costs the defender nothing, for example $r_{\text{NS},\text{NS}} = 0$ and $r_{i,j}|_{i \to j} = 0$, while the defender is punished on any detection delay (DD) ($r_{i,\text{NS}} = c_i^{\text{DD}}; i \neq \text{NS}$), false alarms ($r_{\text{NS},j} = c_j^{\text{FA}}; j \neq \text{NS}$), or wrong type detected (WTD) ($r_{i,j} = c_{i,j}^{\text{WTD}}; i, j \neq \text{NS}, i \nrightarrow j$).

The defender aims to choose its actions based on the received observations such that minimizing the total received cost from the environment. Assuming the detector declares the system is under attack at time step μ. Also, for simplicity, consider $c_i^{\text{DD}} = c^{\text{DD}}$, $c_j^{\text{FA}} = c^{\text{FA}}$, and $c_{i,j}^{\text{WTD}} = c^{\text{WTD}}$. Then, the total corresponding cost is as follows:

$$\mathbf{E}\left[c^{\text{FA}} 1_{t<\tau} + \sum_{t=\tau}^{\mu} c^{\text{DD}} 1_{\mu>\tau} + c^{\text{WTD}} 1_{i \nrightarrow j}\right] \tag{4.5}$$

where the discount factor is assumed to be $\gamma = 1$ due to the fact that current and past costs are equally weighted. The first term is the cost of false alarms, the second term is the cost of detection delay, and the last term is the cost of wrong attack type detection. So, substituting Eq.(4.5) into (4.4), the defender aims to minimize the following cost over the detection time μ:

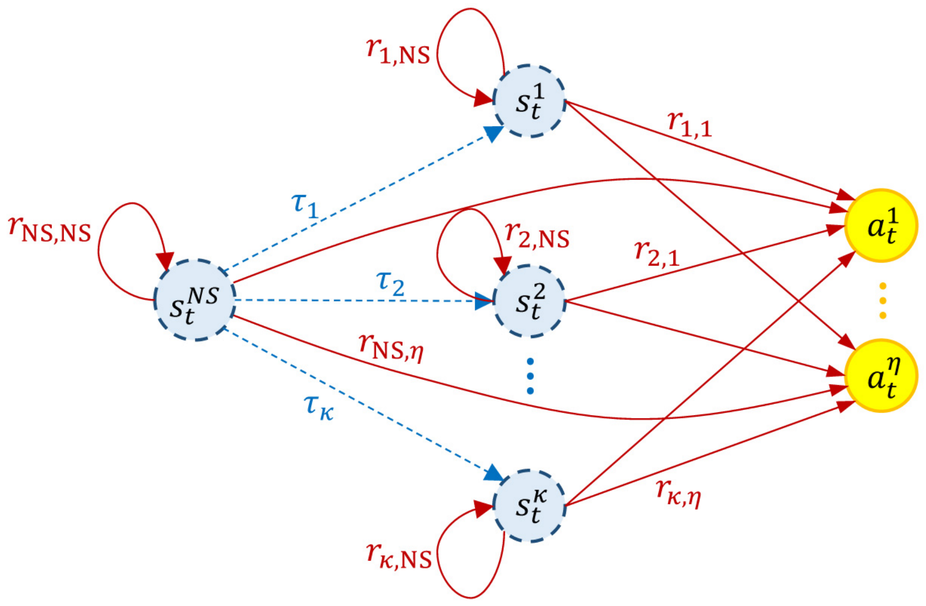

Figure 4.2: States, actions, and rewards of the proposed POMDP for attack detection

$$\begin{aligned}\min_{\mu,\pi} \big[c^{\mathrm{FA}}\mathbf{P}_\tau(\mu < \tau) + c^{\mathrm{DD}}\mathbf{E}(\mu - \tau)1_{\mu>\tau}\\ + c^{\mathrm{WTD}}\mathbf{P}(a^j|s^i, i \nrightarrow j)\big]\end{aligned} \tag{4.6}$$

where $\mathbf{P}$ denotes probability and $\mathbf{P}_\tau$ is the probability measure of launching attacks at time step τ. It worth noting that a trade-off between false alarm, average detection delay, and the accuracy of attack type detection can be obtained by adjusting the relative costs c^{FA}, c^{DD}, and c^{WTD}.

As the attack start time and the attacker policy is unknown, so Eq.(4.6) must be solved by a model-free RL method, that is to say, it should be learned and a direct mapping from observations to the actions is achieved. During the learning process, the agent tries to choose the best action in a given situation (better actions are usually rewarded more in the future). The purpose of learning is to approximate Q-function, such as determining true Q-values for each action in each state based on the observations [112]. The observations must have enough information so that the underlying system state can be inferred. Simply using the observations $\tilde{y}_t$ is not the best choice but using the deviations from the normal operation can leads to more better results. For instance, the difference between the real observations and the expected observations by the Kalman filter is utilized in [62]. However, in this study we exploit the PCA reconstruction error which do not need to the system model. Moreover, the perceptual aliasing, which occurs when similar observations are obtained before and after an attack, can reduce the quality of inference about the underlying state if only considering the reconstruction error at the present time. Hence, the ambiguity of deciding on an attack can be reduced by additional information which can be obtained by considering the history of observations albeit resulting in higher detection delay.

The trade-off between exploration and exploitation is one of the issues in RL. To receive various rewards, an RL agent must prioritize actions that it has already attempted and found to be useful in creating rewards (exploitation). In order to discover such acts, it must try actions it hasn't tried before (exploration). Exploration improves an agent's knowledge of each activity, leading to long-term reward. Exploitation chooses the greedy action by exploiting the agent's current action-value estimates [74]. For each conceivable observation-action pair (o, a) a $Q(o, a)$ value must be learned, i.e., the estimated future cost, using an RL algorithm. Epsilon-Greedy policy randomizes exploration (choosing greedy actions with probability $1 - \epsilon$ which has the highest Q values) and exploitation (random actions with probability ϵ) to balance them. We can start with ϵ closer to 1 initially (more exploration) and bring it closer to 0 in steps as we learn more and more about the environment.

4.1.3 Detection algorithm

In this part, the proposed approach for constructing a measurement residual using PCA is provided first. Model-free Reinforcement Learning is then used to detect an attack.

The proposed algorithm includes two phases "training" and "testing" which can be seen in tables 4.1 and 4.2 respectively. State Action Reward State Action (SARSA) is a model-free RL algorithm which differs from the traditional Q-learning technique in that, instead of using the maximum Q-value from the resulting state as an estimate of that state's value, it uses the Q-value of the resulting state and the action that was actually selected in that state. Thus, the learned values are sensitive to the performed policy. SARSA is considered as an on-policy temporal-difference control learning algorithm which was demonstrated to perform effectively in a model-free POMDP context using numerical analysis. Consequently, in the learning phase, the defender is trained with numerous episodes of experience using the SARSA algorithm and learns a Q-table [86, 105].

In the learning phase which is done for several episodes, a $Q(o, a)$ value must be learned, i.e., the estimated future cost, for each conceivable observation-action pair (o, a) using an RL algorithm in which all $Q(o, a)$ values are recorded in a Q-table of size $I^M \times 2$. After learning the Q-table that must meet formula (4.7) the defender's policy will consist of selecting the action a with the lowest $Q(o, a)$ for each observation o. $\hat{\mathcal{P}}_t$, denoted by $\check{y}_t$.

$$Q(o, a) \leftarrow Q(o, a) + \alpha[r + Q(o^{'}, a^{'}) - Q(o, a)] \tag{4.7}$$

In formula (4.7), α is learning rate, action $a^{'}$ from $o^{'}$ is chosen using ϵ-greedy policy, which uses greedy policy with probability $1 - \epsilon$ and random policy with probability ϵ(ϵ is the exploration rate).

In the testing phase, the states and measurements are calculated and then the measurements are contaminated with noise. The PCA algorithm is implemented on the contaminated and the error is calculated by comparing $\tilde{y}_t$ and $\check{y}_t$. Finally, the defender(agent) makes an action according to its observation and receives a cost in exchange, therefore the defender picks an action based on this experience and updates its Q-values.

During the testing phase, the action with the lowest cost is picked based on the observations, and this phase continues until the defender chooses to "stop" the activity. An attack is declared and the process is terminated.

Table 4.1: SARSA algorithm:Training phase

```
1   Initialize smart grid, detector parameters and $Q(o,a)$
2   for $i = 1$ to the number of episodes:
3     for $t = 0$ to the end of $i$-th episode:
4       Calculate states and collect measurements
5       Generate random $\tau$ from the geometric distribution
6       if $t \geq \tau$:
7         Contaminate measurements with attack values
8       Use the proposed algorithm to calculate measurement error
9       Quantize the error
10      Detect an attack and Determine the next action using epsilon-greedy policy based on Q-table: $Q(o,a) \leftarrow Q(o,a) + \alpha[r + Q(o^{'},a^{'}) - Q(o,a)]$
11      Go to next episode if any attack has been detected
12    end
13    Save results
14  end
```

Table 4.2: SARSA algorithm:Testing phase

```
1   Load saved data and Use Q from the training algorithm
2   for $i = 1$ to the number of episodes:
3     for $t = 0$ to the end of $i$-th episode:
4       Calculate states and collect measurements
5       Generate random $\tau$ from the geometric distribution
6       if $t \geq \tau$:
7         Contaminate measurements with attack values
8       Use the proposed algorithm to calculate measurement error
10      Decide on attack using model-free RL(SARSA) algorithm, Cosine similarity and Euclidean detector
11      Go to next episode if any attack has been detected
12    end
13    Save results
14  end
15  Calculate statistical measures
```

Chapter 5

Simulation results

5.0.1 simulation set up

By using Matlab software, the proposed algorithm has been applied to an IEEE-14 bus power grid with $n+1 = 14$ buses (see figure 5.1) and $q = 23$ meters. The system states $x_t = [x_{1,t}, x_{2,t}, \dots x_{n,t}]^T$ define the system state at time t, where the phase angle at bus i at time t is denoted by $x_{i,t}$, which is obtained using the DC optimal power flow algorithm for case-14 [131]. Let $y_{k,t}$ be the measurement taken at meter k at time t, and $y_t = [y_{1,t}, \dots, y_{q,t}]^T$ denote the measurement vector. The simplified state space model of Eq.(5.1) is used to represent the smart grid:

$$\begin{aligned} x_t &= Ax_{t-1} + v_t \\ y_t &= Hx_t + w_t \end{aligned} \tag{5.1}$$

where $x_t \in R^n$, and $y_t \in R^q$ are the state vector and the output of the system at time step t. $A \in R^{n\times n}$ and $H \in R^{q\times n}$ are state transition matrix and the measurement matrix respectively.The system matrix A is selected as an identity matrix, and the measurement matrix H is derived from the IEEE-14 bus power system. $v_t = [v_{1,t}, ..., v_{n,t}]^T$ is the vector of process noise and $w_t = [w_{1,t}, ..., w_{q,t}]^T$ is the measurement noise vector. The typical process and measurement noises have a Gaussian distribution and are independent additive noises. $v_t \sim \mathcal{N}(0, \sigma_v^2 I)$ and $w_t \sim \mathcal{N}(0, \sigma_w^2 I)$, where I denotes the identity matrix with the appropriate dimension. In addition, let the system be observable through measurements. The standard deviation of the Gaussian process and measurement noises are considered as $\sigma_v = 10^{-2}$ and $\sigma_w = \sqrt{2} \times 10^{-2}$.

The proposed algorithm has been implemented by considering a window size of 30 of received data and reconstructing the data by taking ten largest principal components of gathered data into account. The reconstruction error has been considered for attack detection using the Euclidean and Cosine-similarity detectors. This method's effectiveness has been validated under FDI, jamming, and hybrid (FDI+jamming) attacks of varying magnitudes that are launched at random times τ. Also, The proposed method has been compared with the traditional method based on using Kalman filter innovation for attack detection as shown in Figure 4.1. The simulation scenario has been given in Algorithm 2.

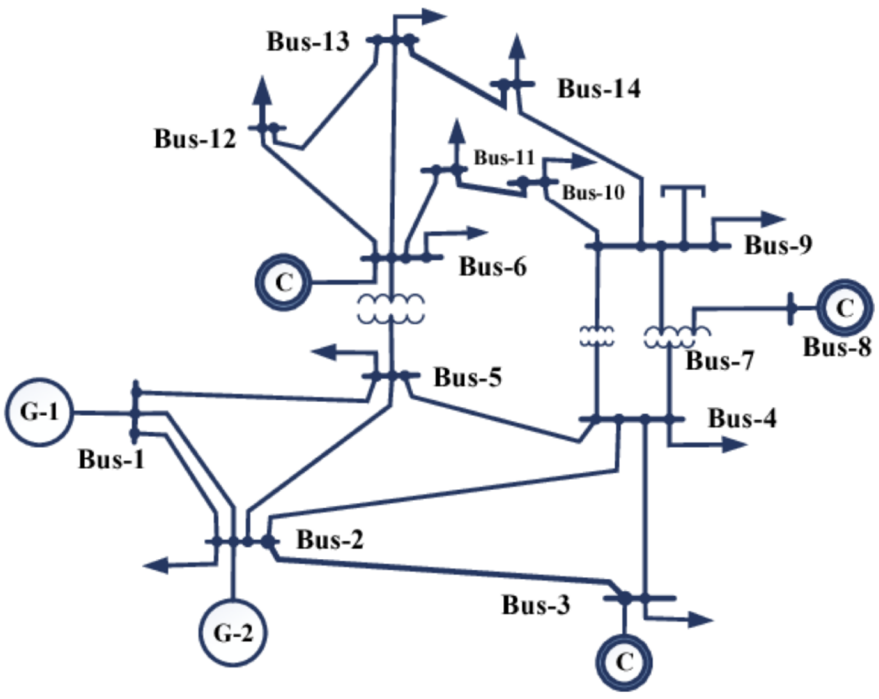

Figure 5.1: Schematic of an IEEE-14 bus power system

5.0.2 Performance Evaluation of PCA-based algorithm vs Kalman Filter using Euclidean and Cosine similarity detectors

The performance of the proposed scheme has been evaluated over 4×10^5 episodes where each attack has been launched at a random time τ. The time of attack has been generated from a geometric distribution with probability parameters from a uniform distribution of $\mathcal{U}[0.001, 0.01]$. An FDI-only, a jamming-only, and a hybrid attack are used to evaluate the proposed scheme's performance. Considering the possible events given in figure 3.2, The measures of binary classification problems are all considered.

FDI attack

First, we use a persistent FDI attack that is both random and time-varying, in which the attacker randomly selects the magnitudes and subsets of fraudulent data to be injected. Figure 5.2 depicts the performance of two proposed detectors, Euclidean and cosine-similarity metric based detectors, which assess the dissimilarity between the actual and projected measurement (by PCA) or predicted measurements (by Kalman filter) and, if the dissimilarity metric exceeds particular thresholds, announce an anomaly.

The measures of Eq.(3.6) based on different detection thresholds for both the PCA and Kalman methods are shown in Fig. 5.2. It can be observed, in overall, that the PCA technique surpasses the Kalman method in both Euclidean and cosine-similarity detectors. Although Kalman recall is slightly greater than PCA recall using the Euclidean starting at threshold 0.17 until 0.19 and cosine similarity starting at threshold $2.75 \times 10^{-}5$ until $3.35 \times 10^{-}5$, it rapidly declines after that, whereas the trend in the PCA method experiences a mild decrease throughout the whole thresholds. As a result, PCA performs better over a wider range of thresholds, which indicates less sensitivity to the

Algorithm 2: Simulation scenario

```
1   Initialize smart grid and detector parameters.
2   for $i = 1$ to the number of episodes:
3       for $t = 0$ to the end of $i$-th episode:
4           Calculate states and collect measurements.
5           Generate random $\tau$ from the geometric distribution.
6           if $t \geq \tau$:
7               Contaminate measurements with attack values.
8           Use Kalman filter to calculate innovation.
9           Use the proposed algorithm to calculate measurement error.
10          Decide on attack using cosine similarity and Euclidean detector.
11          Go to next episode if any attack has been detected.
12      end
13  end
14  Calculate statistical measures.
```

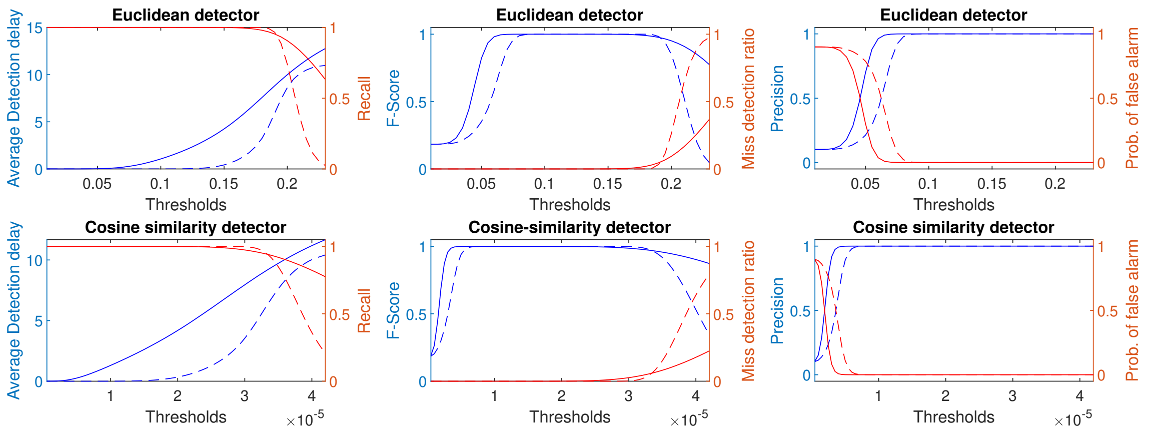

Figure 5.2: Detectors' performance for PCA(solid line) and Kalman filter(dashed line) approaches under FDI attack

threshold. The PCA-based F-score begins with a rapid rise, stabilizes for a while, and then begins to decline steadily. In the Kalman approach however, similar to the case of recall, there is a noticeable delay in F-score increase, it outstrips PCA between the thresholds of 0.18-0.19 in Euclidean and 2.85×10^-5 till 3.3×10^-5 in Cosine similarity detector, but then declines rapidly. Regarding the miss detection ratio, although PCA started with a higher miss detection ratio at thresholds 0.19 in Euclidean and 3.35×10^-5 in Cosine-similarity detector, it falls under the Kalman method and increases mildly, whereas in Kalman starting from the same threshold, the amount of miss detection ratio increases abruptly. In both detectors, the patterns corresponding to the PCA-based approach are without a doubt the most prominent in term of probability of false alarm and precision. As can be observed, the PCA-based method has a lower probability of false alarm than the Kalman-based method in both detectors. Moreover, in the proposed PCA method, the precision of the detection increased prior to the Kalman method.

Jamming attack

The detectors are analyzed in this section in the context of a zero-mean AWGN jamming attack and the evaluation criteria are shown in Figure 5.3. Except for specific thresholds where the Kalman method hits a peak that displays the maximum average detection delay, in both detectors and for both model-based and model-free techniques similar pattern can be observed for average detection delay. In this scenario, the shorter the average detection latency, the better. In terms of recall, the PCA-based technique clearly outperforms and it decreases prior to the Kalman-based method. Also, the F-score in the Euclidean detector has remained stable for a while before rising dramatically until reaching 0.6 and then abruptly declined. In the Kalman-based strategy, on the other hand, the F-score has grown slowly until it reaches a peak of 0.68, after which it drops sharply. Both detectors are following similar trends in terms of miss detection ratio; the only difference is that miss detection occurs for lower thresholds in the PCA approach. Finally, both detectors agree that the PCA-based technique has a lower chance of false alarm than the Kalman-based method. In terms of precision, the PCA approach is superior to the Kalman method, with the exception of a few thresholds, such as the Euclidean detector between 0.66 and 0.71 and the Cosine similarity detector between 0.65 and 0.69.

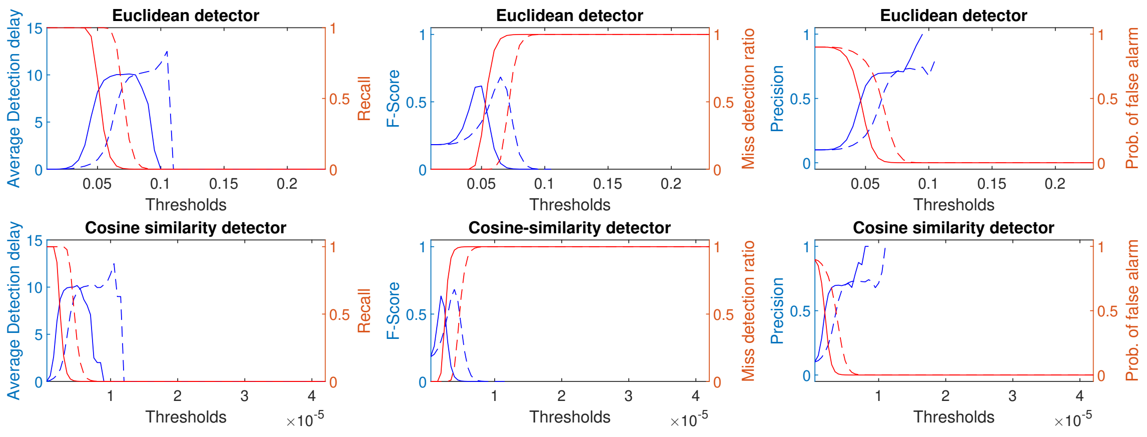

Figure 5.3: Detectors' performance for PCA(solid line) and Kalman filter(dashed line) approaches under Jamming attack

Hybrid attack

Fig 5.8 shows the performance evaluation criteria for both the PCA and Kalman approaches based on different detection thresholds. Starting with Average Detection Delay criteria, where the PCA approach has more detection delay due to the need for a window of data to decide on. However, it can be noted that the Kalman approach requires an accurate system model which is very restrictive. Moving on to recall, both PCA and Kalman remain unchanged for a while, but there is a gradual increase in Kalman during some thresholds, while it rapidly decreased in others, indicating that Kalman is sensitive to threshold selection, whereas PCA method covers a larger range of thresholds and has a smoother decrease. Almost the same thing happened in the presence of Hybrid attack as it did in the case of FDI attack, in the sense of F-score and miss detection ratio. The PCA-based F-score begins with a rapid rise, stabilizes for a while before progressively declining. However, there is a considerable delay in F-score rise in the Kalman technique; it surpasses PCA between the thresholds of 0.18-0.19 in Euclidean and $2.75 \times 10^{-}5$ till $3.3 \times 10^{-}5$ in Cosine similarity detector, but subsequently rapidly decreases. Considering precision and probability of false alarm, the patterns corresponding to the PCA-based technique are by far the most striking. As can be observed, the probability of false alarm in both detectors is lower in the PCA-based method than in the Kalman-based method. On the other hand the accuracy of the correct detection is obviously improved in the proposed PCA method.

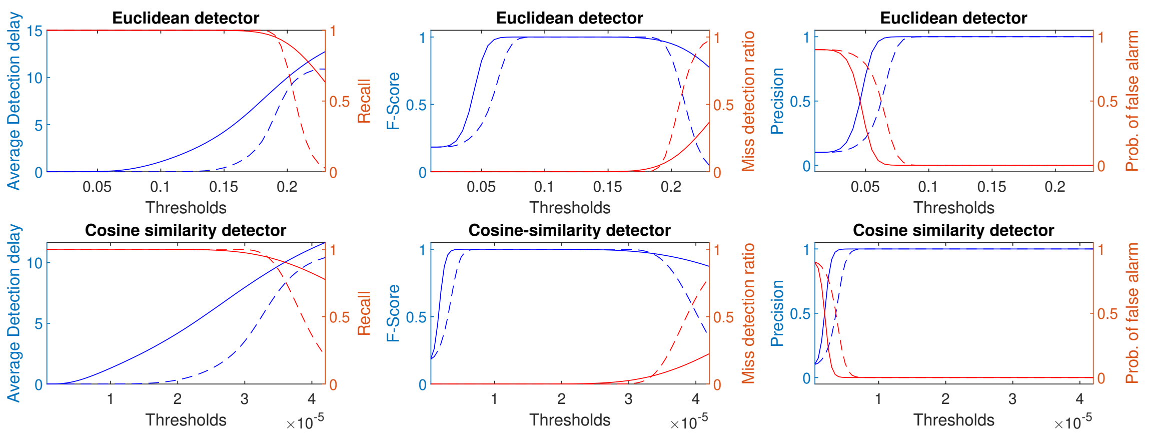

Figure 5.4: Detectors' performance for PCA(solid line) and Kalman filter(dashed line) approaches under Hybrid attack

The effect of attack magnitude on F-score

To investigate the effect of attack magnitude on the performance of the proposed detection method, the F-score of the PCA-based and the difference between the PCA and Kalman method are depicted in Figure 5.5 containing the isolines of F-score for different attack magnitude and detection thresholds. It can be seen that the detection threshold should be chosen more carefully for the low-magnitude attacks because the higher F-score can be obtained for a tighter interval of detection threshold. For instance, the yellow regions in Figure 5.5 indicate the F-score $= 1$ for different attack magnitude, which becomes narrower for lower attack magnitude, both for Euclidean and cosine-similarity detectors. Furthermore, the superiority of the PCA-based method over the Kalman-based method can be seen in Figure 5.5. A band with the bright color at lower threshold for different attack magnitude in the figure of the difference between the PCA and the Kalman-based approach for both Euclidean and cosine-similarity detectors demonstrates that, regardless of attack magnitude, the PCA algorithm has better F-score for lower threshold rather than Kalman-based algorithm. In addition, the bright regions for lower attack magnitude and higher detection threshold show that the PCA-based approach has higher F-score rather than Kalman-based algorithm.

The effect of attack covariance on F-score

Figure 5.6 depicts the F-score of the PCA-based and the difference between the PCA and Kalman methods in order to examine the influence of attack covariance on the performance of the proposed detection approach. In general, in comparison to attack magnitude, the impact of attack covariance

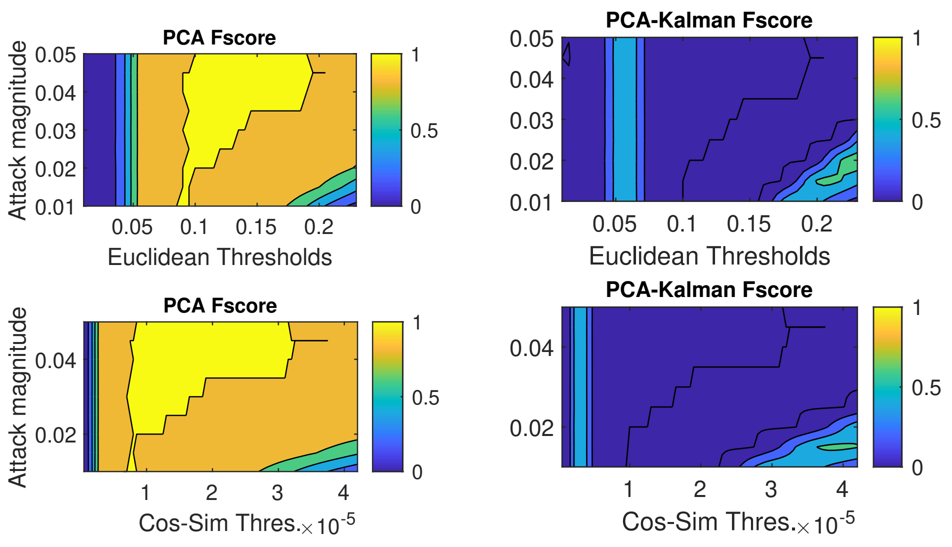

Figure 5.5: The effect of attack magnitude on F-score

has the same qualitative effect. When the attack covariance is small and we are in low thresholds, the F-score is around 0.7, but as attack covariance increases, the PCA F-score achieves its maximum value and remains at this value over narrower ranges of thresholds. The identical circumstance may be observed in the Euclidean detector, although in higher thresholds. Regarding the difference between PCA and Kalman F-score, it is noticeable that the PCA F-score is approximately 50%−60% higher than Kalman F-score in either attacks with low or high covariance. In other words the PCA F-score is greater for a broader range of thresholds in both attacks with low and high covariance and in both detectors.

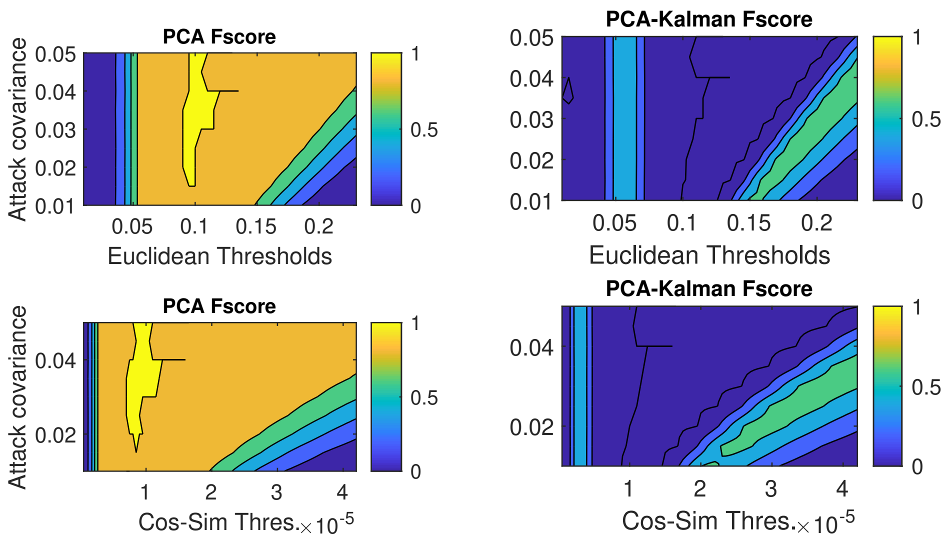

Figure 5.6: The effect of attack covariance on F-score

5.0.3 The effect of data window length on F-score

The more data collected to calculate the principal components of the observation space, the better the F-score. But there is a cost involved, which is computational burden. Thus, the minimum memory length with an acceptable F-score is interested. Figure 5.7 illustrates the obtained F-score by considering different window length for different threshold values. It can be seen that for higher memory length, we can achieve better F-score for broader threshold ranges. It can be seen in Figure 5.7 that considering a window length of 30 can be a good choice for memory length to achieve an acceptable F-score with reasonable computational load.

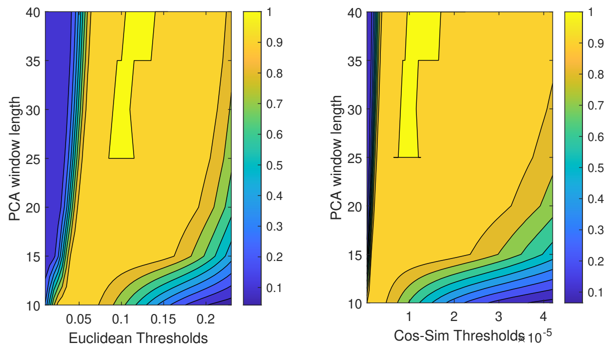

Figure 5.7: The effect of window length on F-score

5.0.4 Performance Evaluation of RL algorithm in comparison with PCA-based and Kalman-based algorithm

The performance of three proposed detectors has been evaluated over 4×10^5 episodes where each attack has been launched at a random time τ. The time of attack has been generated from a geometric distribution with probability parameters from a uniform distribution of $\mathcal{U}[0.001, 0.01]$. Two selected attacks named hybrid attack and FDI attack are used to evaluate the proposed scheme's performance based on the measures of binary classification problem using Monte-Carlo simulations. Figures 5.8 and 5.9 show the performance evaluation criteria based on different detection thresholds.

Hybrid attack

Figure 5.8 shows the performance evaluation criteria for RL, PCA and Kalman approaches based on different detection thresholds. Beginning with Fscore criteria, it is evident that RL has the highest amount of Fscore among other detectors from the beginning until PCA catches RL at threshold 0.55, at which point both detectors experienced the highest amount of Fscore and remained unchanged throughout the whole thresholds until 0.105 . In contrast, Kalman lags behind these methods until the threshold of 0.085, after which it follows them equally. Moving on to Miss Detection Ratio, there is a sharp increase in the amount of Miss Detection Ratio in RL, but from threshold 0.045 it rapidly decreased and remained zero until threshold 0.095, after which it increased gradually and remained at the amount of 0.001 throughout the rest of the analysis. The other two methods do not have any Missed Detection.
In the Precision and False Alarm Probability plot, it is evident that the RL detector outperforms the other two detectors. Prior to the threshold of 0.09, the precision of RL and PCA is greater than that of Kalman, but thereafter, they follow the same trend. Moving to False Alarm Probability RL has the lowest false alarm probability, followed by PCA and Kalman, in that order. PCA reaches RL at the threshold of 0.06 while Kalman catches up with a delay at the threshold of 0.09.
Average Detection Delay and Recall are the final criteria to consider. Beginning with Recall, we

observe a sudden decrease from 1 to 0.975; however, Recall then increases gradually, following other detectors, and remained at 1 throughout all thresholds. Moving on to Average Detection Delay, Kalman encountered zero Average Detection Delay across all thresholds. RL, on the other hand, begins with a greater Detection Delay in comparison to PCA, but at threshold 0.065, it catches up to PCA and follows its trend, except between thresholds 0.080 and 0.090, where it abruptly changes.

FDI attack

we use a persistent FDI attack that is both random and time-varying, in which the attacker randomly selects the magnitudes and subsets of fraudulent data to be injected. Figure 5.9 depicts the performance of of RL, PCA and kalman detectors under FDI attack. The F-score is the first criterion to evaluate. F-score in PCA begins with a higher value, but at threshold 0.045 it falls behind RL until threshold 0.06, after which they both follow the same trend to their maximum value of 1. Concerning Miss Detection Ratio, the RL detector exhibits a sudden drop from 0.02 to 0 whereas the PCA and Kalman detectors maintain a constant value of zero throughout all the thresholds.
The RL technique outperforms two other detectors in terms of accuracy and false alert probability. The precision of the RL technique surpasses that of PCA between the thresholds of 0.045 and 0.060, after which the two methods converge, but the precision of the Kalman approach falls behind that of PCA and RL between the thresholds of 0.085 and 0.085, and then reaches its maximum value. Moving on to the next criterion, false alarm probability, Kalman has the greatest false alarm probability compared to other detectors. In contrast, RL begins with a greater probability of false alarm than Kalman, yet PCA surpassed RL between thresholds 0.045 − 0.06. From a threshold of 0.09, this criterion converges to zero using all three techniques.
Regarding AVerage Detection Delay, similar trends are observed in the case of FDI attack as in the case of hybrid attack, with the exception that in the case of FDI attack, after a gradual decrease from 0.465 − 0.122, it experienced a gradual increase without any abrupt change until the final thresholds. PCA and Kalman surpassed RL in terms of recall beginning at thresholds between 0.04 − 0.06 and 0.09 − 0.105, although at other thresholds they have the same recall value.

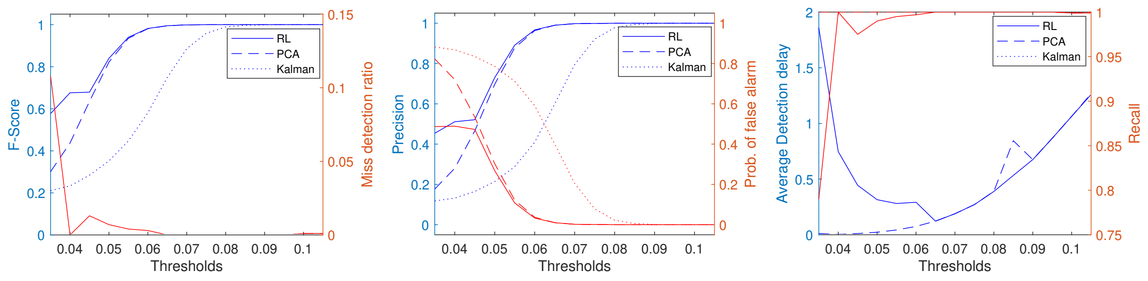

Figure 5.8: Detectors' performance under Hybrid attack

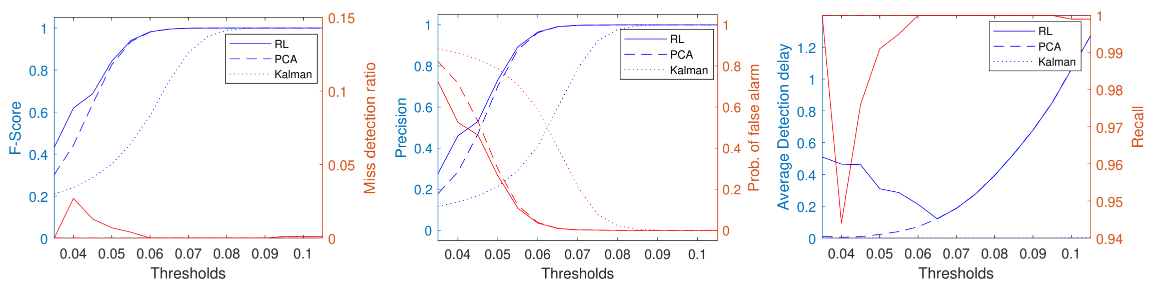

Figure 5.9: Detectors' performance under FDI attack

5.0.5 The effect of memory length on Fscore and Average Detection Delay of PCA-based, Kalman-based and RL detectors

In Figures 5.10 and 5.11 the impact of increasing memory length on the Fscore and Average Detection Delay is examined. The RL detector is compared to PCA-based and Kalman-based detectors in these figures. Consider the case when $M = 1$ (no memory) and we have just one quantization level which in this case is exactly the same as the thresholds of PCA-based and Kalman-based method. As it can be seen from Figures 5.10 and 5.11 the performance of RL algorithm is exactly the same as PCA-based algorithm when we do not have any memory. As the memory increases at lower thresholds, the Fscore in the RL method experiences a slight increase in comparison with PCA method, whereas at higher thresholds this trend is reversed and the Fscore experiences a slight decrease as the memory increases. In the kalman case, as memory increased, the RL detector appeared to perform better than the kalman detector, except at higher thresholds, where they performed almost identically.

Regarding Figure 5.11, in all three methods, it is evident that the Average Detection Delay in RL is higher with respect to other methods and it is reasonable because in this method we have memory. The Average Detection Delay for RL is approximately 50% greater than the PCA method when the memory length is between 3 and 4, and this difference rises as the memory length increases. In contrast, at higher thresholds, the Average Detection Delay is nearly identical in both these methods. In the diagram comparing the RL and Kalman methods, it is evident that at lower thresholds, as memory increased, the RL method experienced an increase in Average Detection Delay, whereas at thresholds between 0.06 and 0.07, the rate of change in Average Detection Delay decreased. We observe the same trend at higher thresholds regardless of the size of the memory, but RL typically has a longer average detection delay than Kalman.

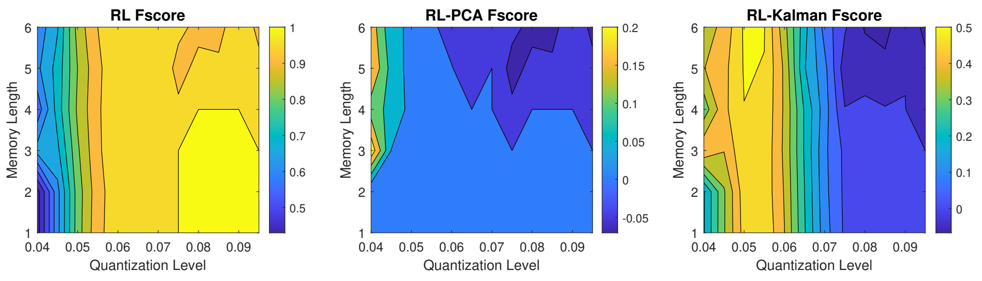

Figure 5.10: The effect of increasing memory length on F-score

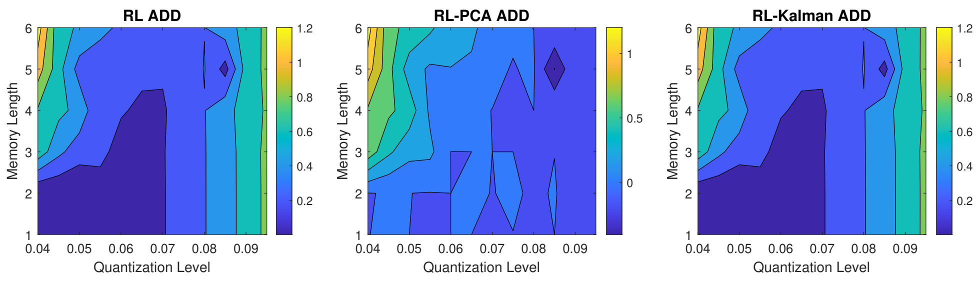

Figure 5.11: The effect of increasing memory length on Average Detection Delay

Chapter 6

Conclusion

Attacks must be immediately recognized, identified, and pinpointed in order to take quick action to defend the entire system and ensure the safe and proper operation of the CPS. Due to the unpredictable nature of attacks, correctly detecting cyberattacks is the first step in building resilience throughout the attack period and/or post-attack phase. Anomaly detection systems face a number of difficulties (ADS). The majority of cyber-attack detection techniques, including state estimate methods, first create a model based on the available data to determine the system's typical behavior. The system then determines whether or not the behavior of the system is normal by contrasting the model's estimated outputs with the results of the actual process. Regarding the second issue, a number of intrusion detection techniques, including machine learning (ML), can automatically create the model from the training data set, which contains data instances that can be identified by a number of qualities (features) and associated labels. Although handling huge amounts of data, sometimes known as "big data," is difficult, it is necessary. An online model-free approach is suggested to identify false data injection and jamming attacks on cyber-physical systems in order to address all of the difficulties in this study. Using principal component analysis in the observation space, the proposed method reconstructs expected observations in a reduced dimension space based on the most effective principal components. Then, the existence of an attack or normal operation is decided based on the measurement residual by using the Euclidean detector and the cosine-similarity metric detector. Some of the key features of the proposed approach includes: first of all, No prior knowledge of the system is required because of the model-free nature of the system. Secondly, We use projections rather than estimations space in this method. By simulating an IEEE-14 bus power system with 23 smart meters, the proposed technique has been tested. Additionally, the findings have been contrasted with those obtained using the model-based Kalman estimate method, demonstrating the superiority of the suggested approach in terms of precision, recall, and F-score, three statistical measures of binary classification issues. By utilizing the measurement offered by our novel PCA-based approach, we extend our research by describing the online attack/anomaly detection problem as a partially observable Markov decision process (POMDP) problem using the paradigm of model-free reinforcement learning (RL) for POMDPs. The strength of RL algorithm is in the capability of RL to compare the expected utility of available actions without a model of the environment. Moreover, RL is a technique in which an agent learns how to solve a problem without a supervisor. Simulation results demonstrates that the recommended RL-based algorithm is effective at identifying cyber-attacks on the smart grid, according to numerical experiments done in Matlab.

In first part of simulation results both the PCA-based method and Kalman method are compared. Overall, it is evident that the PCA approach outperforms the Kalman technique in both cosine-similarity and Euclidean detectors. One of the most outstanding priority of our proposed method is the fact is that PCA performs better across a larger variety of thresholds, indicating less sensitivity to the threshold in comparison to the Kalman-based method. Regarding probabilty of false alarm, the PCA-based approach has less probability of false alarm in both detectors compared to Kalman method. In addition, the PCA method that was suggested had higher detection precision than the Kalman. We extend our research to check the effect of magnitude and covariance on F-score. It is concluded that regardless of attack magnitude, the PCA algorithm has better F-score for lower threshold rather than Kalman-based algorithm. In addition, the bright regions for lower attack magnitude and higher detection threshold show that the PCA-based approach has higher F-score rather than Kalman-based algorithm. In comparison to attack magnitude, the impact of attack covariance has the same qualitative effect. In other words the PCA F-score is greater for a broader range of thresholds in both attacks with low and high covariance and in both detectors.

In the second part of simulations the performance of three proposed detectors including RL, PCA and Kalman has been evaluated. Beginning with Fscore criteria, it is evident that RL has the highest amount of Fscore among other detectors. In the Precision and False Alarm Probability plot, it is evident that the RL detector outperforms the other two detectors. Moving on to the next criterion, false alarm probability, Kalman has the greatest false alarm probability compared to other detectors. In contrast, RL begins with a greater probability of false alarm than Kalman, yet PCA surpassed RL between some thresholds. One of the important facts regarding RL method is that in this method we have memory so we need to consider the effect of memory length on Fscore and average detection delay. Consider the case when M = 1 (no memory) and we have just one quantization level which in this case is exactly the same as the thresholds of PCA-based and Kalman-based method. The performance of RL algorithm is exactly the same as PCA-based algorithm when we do not have any memory. As the memory increases at lower thresholds, the Fscore in the RL method experiences a slight increase in comparison to PCA method, whereas at higher thresholds this trend is reversed and the Fscore experiences a slight decrease as the memory increases. In the kalman case, as memory increased, the RL detector appeared to perform better than the kalman detector, except at higher thresholds, where they performed almost identically. In all three methods, it is evident that the Average Detection Delay in RL is higher with respect to other methods and it is reasonable because in this method we have memory.The Average Detection Delay for RL is approximately 50% greater than the PCA method when the memory length is between 3 and 4, and this difference rises as the memory length increases. In contrast, at higher thresholds, the Average Detection Delay is nearly identical in both these methods. By comparing the RL and Kalman methods, it is evident that at lower thresholds, as memory increased, the RL method experienced an increase in Average Detection Delay, whereas at some thresholds, the rate of change in Average Detection Delay decreased. We observe the same trend at higher thresholds regardless of the size of the memory, but RL typically has a longer average detection delay than Kalman. So in general PCA method and RL method have proved to be effective in attack detection. However, attack detection algorithms which make decision based on the level of an estimation error are highly sensitive to common system changes such as load variations. One potential solution can be taking the history of reconstruction error into account and using machine learning algorithms which is the prospect of our future work.

Bibliography

[1] D. Abbasinezhad-Mood and M. Nikooghadam. Design and hardware implementation of a security-enhanced elliptic curve cryptography based lightweight authentication scheme for smart grid communications. *Future Generation Computer Systems*, 84:47–57, 2018.

[2] A. Abbaspour, A. Sargolzaei, P. Forouzannezhad, K. K. Yen, and A. I. Sarwat. Resilient control design for load frequency control system under false data injection attacks. *IEEE Transactions on Industrial Electronics*, 67(9):7951–7962, 2019.

[3] M. R. C. Acosta, S. Ahmed, C. E. Garcia, and I. Koo. Extremely randomized trees-based scheme for stealthy cyber-attack detection in smart grid networks. *IEEE access*, 8:19921–19933, 2020.

[4] S. Ahmed, Y. Lee, S.-H. Hyun, and I. Koo. Unsupervised machine learning-based detection of covert data integrity assault in smart grid networks utilizing isolation forest. *IEEE Transactions on Information Forensics and Security*, 14(10):2765–2777, 2019.

[5] M. T. Ahvanooey, M. X. Zhu, Q. Li, W. Mazurczyk, K.-K. R. Choo, B. B. Gupta, and M. Conti. Modern authentication schemes in smartphones and iot devices: An empirical survey. *IEEE Internet of Things Journal*, 2021.

[6] O. Akinpelumi and K. Kopsidas. Impact analysis of cyber-related failures on power system reliability-a review. *2021 IEEE Madrid PowerTech*, pages 1–6, 2021.

[7] M. S. Al-kahtani and L. Karim. A survey on attacks and defense mechanisms in smart grids. *International Journal of Computer Engineering and Information Technology*, 11(5):94–100, 2019.

[8] M. A. Alohali, F. N. Al-Wesabi, A. M. Hilal, S. Goel, D. Gupta, and A. Khanna. Artificial intelligence enabled intrusion detection systems for cognitive cyber-physical systems in industry 4.0 environment. *Cognitive Neurodynamics*, pages 1–13, 2022.

[9] Y. An and D. Liu. Multivariate gaussian-based false data detection against cyber-attacks. *IEEE Access*, 7:119804–119812, 2019.

[10] A. Anwar, A. N. Mahmood, and M. Pickering. Modeling and performance evaluation of stealthy false data injection attacks on smart grid in the presence of corrupted measurements. *Journal of Computer and System Sciences*, 83(1):58–72, 2017.

[11] A. Aydeger, K. Akkaya, M. H. Cintuglu, A. S. Uluagac, and O. Mohammed. Software defined networking for resilient communications in smart grid active distribution networks. In *2016 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE, 2016.

[12] K. Azizzadenesheli, A. Lazaric, and A. Anandkumar. Reinforcement learning of pomdps using spectral methods. In *Conference on Learning Theory*, pages 193–256. PMLR, 2016.

[13] P. Bajpai, S. Chanda, and A. K. Srivastava. A novel metric to quantify and enable resilient distribution system using graph theory and choquet integral. *IEEE Transactions on Smart Grid*, 9(4):2918–2929, 2016.

[14] H. E. Brown and C. L. DeMarco. Risk of cyber-physical attack via load with emulated inertia control. *IEEE Transactions on Smart Grid*, 9(6):5854–5866, 2017.

[15] M. M. Buzau, J. Tejedor-Aguilera, P. Cruz-Romero, and A. Gómez-Expósito. Detection of non-technical losses using smart meter data and supervised learning. *IEEE Transactions on Smart Grid*, 10(3):2661–2670, 2018.

[16] T. T. Cai, Z. Ma, and Y. Wu. Sparse pca: Optimal rates and adaptive estimation. *The Annals of Statistics*, 41(6):3074–3110, 2013.

[17] A. Califano, E. Dincelli, and S. Goel. Using features of cloud computing to defend smart grid against ddos attacks. In *10th Annual symposium on information assurance (Asia 15), ALBANY*, pages 44–50, 2015.

[18] D. U. Case. Analysis of the cyber attack on the ukrainian power grid. *Electricity Information Sharing and Analysis Center (E-ISAC)*, 388:1–29, 2016.

[19] Y. Chakhchoukh, H. Lei, and B. K. Johnson. Diagnosis of outliers and cyber attacks in dynamic pmu-based power state estimation. *IEEE transactions on power systems*, 35(2):1188–1197, 2019.

[20] S. Chanda, A. K. Srivastava, M. U. Mohanpurkar, and R. Hovsapian. Quantifying power distribution system resiliency using code-based metric. *IEEE Transactions on Industry Applications*, 54(4):3676–3686, 2018.

[21] L. Che, X. Liu, and Z. Li. Mitigating false data attacks induced overloads using a corrective dispatch scheme. *IEEE Transactions on Smart Grid*, 10(3):3081–3091, 2018.

[22] P.-Y. Chen, S.-M. Cheng, and K.-C. Chen. Smart attacks in smart grid communication networks. *IEEE Communications Magazine*, 50(8):24–29, 2012.

[23] T. Chen, X. Yin, and G. Wang. Securing communications between smart grids and real users; providing a methodology based on user authentication. *Energy Reports*, 7:8042–8050, 2021.

[24] Y. Chen. *Translating film subtitles into Chinese: A multimodal study*. Springer, 2019.

[25] Y. Chen, S. Huang, F. Liu, Z. Wang, and X. Sun. Evaluation of reinforcement learning-based false data injection attack to automatic voltage control. *IEEE Transactions on Smart Grid*, 10(2):2158–2169, 2018.

[26] R. Colbaugh and K. Glass. Proactive defense for evolving cyber threats. In *Proceedings of 2011 IEEE International Conference on Intelligence and Security Informatics*, pages 125–130. IEEE, 2011.

[27] B. C. Costa, B. L. Alberto, A. M. Portela, W. Maduro, and E. O. Eler. Fraud detection in electric power distribution networks using an ann-based knowledge-discovery process. *International Journal of Artificial Intelligence & Applications*, 4(6):17, 2013.

[28] L. Cui, Y. Qu, L. Gao, G. Xie, and S. Yu. Detecting false data attacks using machine learning techniques in smart grid: A survey. *Journal of Network and Computer Applications*, 170:102808, 2020.

[29] C. Czosseck, R. Ottis, and A.-M. Talihärm. Estonia after the 2007 cyber attacks: Legal, strategic and organisational changes in cyber security. *International Journal of Cyber Warfare and Terrorism (IJCWT)*, 1(1):24–34, 2011.

[30] A. Daeichian and E. Honarvar. Modified covariance intersection for data fusion in distributed nonhomogeneous monitoring systems network. *International Journal of Robust and Nonlinear Control*, 28(4):1413–1424, 2018.

[31] M. De Nadai and M. Van Someren. Short-term anomaly detection in gas consumption through arima and artificial neural network forecast. In *2015 IEEE Workshop on Environmental, Energy, and Structural Monitoring Systems (EESMS) Proceedings*, pages 250–255. IEEE, 2015.

[32] K. Dehghanpour, Z. Wang, J. Wang, Y. Yuan, and F. Bu. A survey on state estimation techniques and challenges in smart distribution systems. *IEEE Transactions on Smart Grid*, 10(2):2312–2322, 2018.

[33] R. Deng and H. Liang. False data injection attacks with limited susceptance information and new countermeasures in smart grid. *IEEE Transactions on Industrial Informatics*, 15(3):1619–1628, 2018.

[34] S. Y. Diaba, M. Shafie-khah, and M. Elmusrati. On the performance metrics for cyber-physical attack detection in smart grid. *Soft Computing*, pages 1–10, 2022.

[35] S. M. Dibaji, M. Pirani, D. B. Flamholz, A. M. Annaswamy, K. H. Johansson, and A. Chakrabortty. A systems and control perspective of cps security. *Annual reviews in control*, 47:394–411, 2019.

[36] R. Diovu and J. Agee. A cloud-based openflow firewall for mitigation against ddos attacks in smart grid ami networks. In *2017 IEEE PES PowerAfrica*, pages 28–33. IEEE, 2017.

[37] S. Donaldson, S. Siegel, C. K. Williams, and A. Aslam. *Enterprise cybersecurity: how to build a successful cyberdefense program against advanced threats.* Apress, 2015.

[38] J. Dsouza, L. Elezabeth, V. P. Mishra, and R. Jain. Security in cyber-physical systems. In *2019 Amity International Conference on Artificial Intelligence (AICAI)*, pages 840–844. IEEE, 2019.

[39] J. Duan, H. Xu, and W. Liu. Q-learning-based damping control of wide-area power systems under cyber uncertainties. *IEEE Transactions on Smart Grid*, 9(6):6408–6418, 2017.

[40] O. Erdene-Ochir, M. Abdallah, K. Qaraqe, M. Minier, and F. Valois. Routing resilience evaluation for smart metering: Definition, metric and techniques. In *2014 IEEE 25th Annual International Symposium on Personal, Indoor, and Mobile Radio Communication (PIMRC)*, pages 1867–1871. IEEE, 2014.

[41] B. Falahati and Y. Fu. Reliability assessment of smart grids considering indirect cyber-power interdependencies. *IEEE Transactions on Smart Grid*, 5(4):1677–1685, 2014.

[42] B. Falahati, Y. Fu, and L. Wu. Reliability assessment of smart grid considering direct cyber-power interdependencies. *IEEE Transactions on Smart Grid*, 3(3):1515–1524, 2012.

[43] E. Galvan, P. Mandal, and Y. Sang. Networked microgrids with roof-top solar pv and battery energy storage to improve distribution grids resilience to natural disasters. *International Journal of Electrical Power & Energy Systems*, 123:106239, 2020.

[44] M. Ghiasi, M. Dehghani, T. Niknam, H. R. Baghaee, S. Padmanaban, G. B. Gharehpetian, and H. Aliev. Resiliency/cost-based optimal design of distribution network to maintain power system stability against physical attacks: A practical study case. *IEEE Access*, 9:43862–43875, 2021.

[45] M. Z. Gunduz and R. Das. Cyber-security on smart grid: Threats and potential solutions. *Computer networks*, 169:107094, 2020.

[46] S. Hasan, A. Dubey, G. Karsai, and X. Koutsoukos. A game-theoretic approach for power systems defense against dynamic cyber-attacks. *International Journal of Electrical Power & Energy Systems*, 115:105432, 2020.

[47] Y. He, G. J. Mendis, and J. Wei. Real-time detection of false data injection attacks in smart grid: A deep learning-based intelligent mechanism. *IEEE Transactions on Smart Grid*, 8(5):2505–2516, 2017.

[48] M. E. Hellman. An overview of public key cryptography. *IEEE Communications Magazine*, 40(5):42–49, 2002.

[49] E. Honarvar and A. Daeichian. A pca-based algorithm for online fdi and jamming attack detection in cps. Manuscript submitted for publication, 2022.

[50] T. Hu, Q. Guo, X. Shen, H. Sun, R. Wu, and H. Xi. Utilizing unlabeled data to detect electricity fraud in ami: A semisupervised deep learning approach. *IEEE transactions on neural networks and learning systems*, 30(11):3287–3299, 2019.

[51] A. Hussain, V.-H. Bui, and H.-M. Kim. Resilience-oriented optimal operation of networked hybrid microgrids. *IEEE Transactions on Smart Grid*, 10(1):204–215, 2017.

[52] T. Irita and T. Namerikawa. Detection of replay attack on smart grid with code signal and bargaining game. In *2017 American Control Conference (ACC)*, pages 2112–2117. IEEE, 2017.

[53] S. Iyer. Cyber security for smart grid, cryptography, and privacy. *International Journal of Digital Multimedia Broadcasting*, 2011, 2011.

[54] P. Jamborsalamati, M. J. Hossain, S. Taghizadeh, G. Konstantinou, M. Manbachi, and P. Dehghanian. Enhancing power grid resilience through an iec61850-based ev-assisted load restoration. *IEEE Transactions on Industrial Informatics*, 16(3):1799–1810, 2019.

[55] B. Jang, M. Kim, G. Harerimana, and J. W. Kim. Q-learning algorithms: A comprehensive classification and applications. *IEEE access*, 7:133653–133667, 2019.

[56] L. Jiang, H. Zhang, J. Liu, X. Shen, and H. Xu. A multi-sensor cycle-supervised convolutional neural networks for anomaly detection on magnetic flux leakage signals. *IEEE Transactions on Industrial Informatics*, 2022.

[57] V. Karpagam, A. Kayalvizhi, N. Bharathi, and S. Balamurugan. Internet of things: Applications, vulnerabilities, and the need for cyber resilience. *Cyber-Physical Systems and Industry 4.0: Practical Applications and Security Management*, pages 45–55, 2022.

[58] K. Khanna, B. K. Panigrahi, and A. Joshi. Ai-based approach<? show [aq="" id=" q1]"?> to identify compromised meters in data integrity attacks on smart grid. *IET Generation, Transmission & Distribution*, 12(5):1052–1066, 2018.

[59] J. Kim, G. Park, H. Shim, and Y. Eun. A zero-stealthy attack for sampled-data control systems via input redundancy. *arXiv preprint arXiv:1801.03609*, 2018.

[60] S. Kim and K.-J. Park. A survey on machine-learning based security design for cyber-physical systems. *Applied Sciences*, 11(12):5458, 2021.

[61] C. Kreibich and J. Crowcroft. Honeycomb: creating intrusion detection signatures using honeypots. *ACM SIGCOMM computer communication review*, 34(1):51–56, 2004.

[62] M. N. Kurt, O. Ogundijo, C. Li, and X. Wang. Online cyber-attack detection in smart grid: A reinforcement learning approach. *IEEE Transactions on Smart Grid*, 10(5):5174–5185, 2018.

[63] M. N. Kurt, Y. Yılmaz, and X. Wang. Real-time detection of hybrid and stealthy cyber-attacks in smart grid. *IEEE Transactions on Information Forensics and Security*, 14(2):498–513, 2018.

[64] A. Kwasinski. Quantitative model and metrics of electrical grids' resilience evaluated at a power distribution level. *Energies*, 9(2):93, 2016.

[65] H. Lei, B. Chen, K. L. Butler-Purry, and C. Singh. Security and reliability perspectives in cyber-physical smart grids. In *2018 IEEE Innovative Smart Grid Technologies-Asia (ISGT Asia)*, pages 42–47. IEEE, 2018.

[66] H. Lei and C. Singh. Power system reliability evaluation considering cyber-malfunctions in substations. *Electric Power Systems Research*, 129:160–169, 2015.

[67] H. Lei, C. Singh, and A. Sprintson. Reliability modeling and analysis of iec 61850 based substation protection systems. *IEEE Transactions on Smart Grid*, 5(5):2194–2202, 2014.

[68] B. Li, T. Ding, C. Huang, J. Zhao, Y. Yang, and Y. Chen. Detecting false data injection attacks against power system state estimation with fast go-decomposition approach. *IEEE Transactions on Industrial Informatics*, 15(5):2892–2904, 2018.

[69] Y. Li, Y. Wang, and S. Hu. Online generative adversary network based measurement recovery in false data injection attacks: A cyber-physical approach. *IEEE Transactions on Industrial Informatics*, 16(3):2031–2043, 2019.

[70] Y. Liang, D. He, and D. Chen. Poisoning attack on load forecasting. In *2019 IEEE Innovative Smart Grid Technologies-Asia (ISGT Asia)*, pages 1230–1235. IEEE, 2019.

[71] J. Liu, Y. Xiao, and J. Gao. Achieving accountability in smart grid. *IEEE Systems Journal*, 8(2):493–508, 2013.

[72] J. Liu, Y. Xiao, S. Li, W. Liang, and C. P. Chen. Cyber security and privacy issues in smart grids. *IEEE Communications Surveys & Tutorials*, 14(4):981–997, 2012.

[73] X. Ma, X. Tu, J. Huang, and J. He. A cyber-physical system based framework for motor rehabilitation after stroke. In *Proceedings of the 1st International Conference on Wireless Technologies for Humanitarian Relief*, pages 285–290, 2011.

[74] M. C. Machado, S. Srinivasan, and M. Bowling. Domain-independent optimistic initialization for reinforcement learning. In *Workshops at the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[75] K. Mahmood, S. A. Chaudhry, H. Naqvi, T. Shon, and H. F. Ahmad. A lightweight message authentication scheme for smart grid communications in power sector. *Computers & Electrical Engineering*, 52:114–124, 2016.

[76] K. Manandhar, X. Cao, F. Hu, and Y. Liu. Detection of faults and attacks including false data injection attack in smart grid using kalman filter. *IEEE transactions on control of network systems*, 1(4):370–379, 2014.

[77] Y. Mao, H. Jafarnejadsani, P. Zhao, E. Akyol, and N. Hovakimyan. Detectability of intermittent zero-dynamics attack in networked control systems. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 5605–5610. IEEE, 2019.

[78] G. M. Messinis, A. E. Rigas, and N. D. Hatziargyriou. A hybrid method for non-technical loss detection in smart distribution grids. *IEEE Transactions on Smart Grid*, 10(6):6080–6091, 2019.

[79] A. R. Metke and R. L. Ekl. Security technology for smart grid networks. *IEEE Transactions on Smart Grid*, 1(1):99–107, 2010.

[80] M. Mohammadpourfard, A. Sami, and Y. Weng. Identification of false data injection attacks with considering the impact of wind generation and topology reconfigurations. *IEEE Transactions on Sustainable Energy*, 9(3):1349–1364, 2017.

[81] R. Moslemi, A. Mesbahi, and J. M. Velni. A fast, decentralized covariance selection-based approach to detect cyber attacks in smart grids. *IEEE Transactions on Smart Grid*, 9(5):4930–4941, 2017.

[82] A. Mpitziopoulos, D. Gavalas, C. Konstantopoulos, and G. Pantziou. A survey on jamming attacks and countermeasures in wsns. *IEEE Communications Surveys & Tutorials*, 11(4):42–56, 2009.

[83] S. Omar, A. Ngadi, and H. H. Jebur. Machine learning techniques for anomaly detection: an overview. *International Journal of Computer Applications*, 79(2), 2013.

[84] S. G. I. PANEL. A white paper developed by the smart grid interoperability panel smart grid cybersecurity committee february 2014. 2014.

[85] C. M. Paredes, D. Martínez-Castro, V. Ibarra-Junquera, and A. González-Potes. Detection and isolation of dos and integrity cyber attacks in cyber-physical systems with a neural network-based architecture. *Electronics*, 10(18):2238, 2021.

[86] L. Peshkin, N. Meuleau, and L. Kaelbling. Learning policies with external memory. *arXiv preprint cs/0103003*, 2001.

[87] M. Pilz, F. B. Naeini, K. Grammont, C. Smagghe, M. Davis, J.-C. Nebel, L. Al-Fagih, and E. Pfluegel. Security attacks on smart grid scheduling and their defences: a game-theoretic approach. *International Journal of Information Security*, 19(4):427–443, 2020.

[88] S. Poudel, A. Dubey, and A. Bose. Risk-based probabilistic quantification of power distribution system operational resilience. *IEEE Systems Journal*, 14(3):3506–3517, 2019.

[89] R. Punmiya and S. Choe. Energy theft detection using gradient boosting theft detector with feature engineering-based preprocessing. *IEEE Transactions on Smart Grid*, 10(2):2326–2329, 2019.

[90] W. Qiang and Z. Zhongli. Reinforcement learning model, algorithms and its application. In *2011 International Conference on Mechatronic Science, Electric Engineering and Computer (MEC)*, pages 1143–1146. IEEE, 2011.

[91] M. A. Rahman and H. Mohsenian-Rad. False data injection attacks with incomplete information against smart power grids. In *2012 IEEE Global Communications Conference (GLOBECOM)*, pages 3153–3158. IEEE, 2012.

[92] R. Rekha. Medical cyber-physical systems security. *Cyber-Physical Systems and Industry 4.0: Practical Applications and Security Management*, pages 57–74, 2022.

[93] J. R. Renofio, M. E. Pellenz, A. Santin, E. Jamhour, M. C. Penna, and R. D. Souza. Insights on the resilience and capacity of ami wireless networks. In *2016 IEEE Symposium on Computers and Communication (ISCC)*, pages 610–615. IEEE, 2016.

[94] J. Sakhnini, H. Karimipour, and A. Dehghantanha. Smart grid cyber attacks detection using supervised learning and heuristic feature selection. In *2019 IEEE 7th International Conference on Smart Energy Grid Engineering (SEGE)*, pages 108–112. IEEE, 2019.

[95] A. Salvi, P. Spagnoletti, and N. S. Noori. Cyber-resilience of critical cyber infrastructures: Integrating digital twins in the electric power ecosystem. *Computers & Security*, 112:102507, 2022.

[96] M. Shahraeini and P. Kotzanikolaou. A dependency analysis model for resilient wide area measurement systems in smart grid. *IEEE Journal on Selected Areas in Communications*, 38(1):156–168, 2019.

[97] S. Sharmeen, S. Huda, J. Abawajy, C. M. Ahmed, M. M. Hassan, and G. Fortino. An advanced boundary protection control for the smart water network using semi supervised and deep learning approaches. *IEEE Internet of Things Journal*, 2021.

[98] K. Shaukat, S. Luo, V. Varadharajan, I. A. Hameed, S. Chen, D. Liu, and J. Li. Performance comparison and current challenges of using machine learning techniques in cybersecurity. *Energies*, 13(10):2509, 2020.

[99] D. Shelar, S. Amin, and I. Hiskens. Resilience of electricity distribution networks-part ii: leveraging microgrids. *arXiv preprint arXiv:1812.01745*, 2018.

[100] H. Shim, J. Back, Y. Eun, G. Park, and J. Kim. Zero-dynamics attack, variations, and countermeasures. In *Security and Resilience of Control Systems*, pages 31–61. Springer, 2022.

[101] R. Singh, A. Singh, and P. Bhattacharya. A machine learning approach for anomaly detection to secure smart grid systems. In *Research Anthology on Smart Grid and Microgrid Development*, pages 911–923. IGI Global, 2022.

[102] T. Sreeram and S. Krishna. Managing false data injection attacks during contingency of secured meters. *IEEE Transactions on Smart Grid*, 10(6):6945–6953, 2019.

[103] Q. Su, H. Wang, C. Sun, B. Li, and J. Li. Cyber-attacks against cyber-physical power systems security: State estimation, attacks reconstruction and defense strategy. *Applied Mathematics and Computation*, 413:126639, 2022.

[104] M. Sun, I. Konstantelos, and G. Strbac. C-vine copula mixture model for clustering of residential electrical load pattern data. *IEEE transactions on power systems*, 32(3):2382–2393, 2016.

[105] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction.* MIT press, 2018.

[106] J. M. Taylor and H. R. Sharif. Security challenges and methods for protecting critical infrastructure cyber-physical systems. In *2017 International Conference on Selected Topics in Mobile and Wireless Networking (MoWNeT)*, pages 1–6. IEEE, 2017.

[107] A. Tchernykh, U. Schwiegelsohn, E.-g. Talbi, and M. Babenko. Towards understanding uncertainty in cloud computing with risks of confidentiality, integrity, and availability. *Journal of Computational Science*, 36:100581, 2019.

[108] A. Teixeira, I. Shames, H. Sandberg, and K. H. Johansson. Revealing stealthy attacks in control systems. In *2012 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1806–1813. IEEE, 2012.

[109] C.-W. Ten, C.-C. Liu, and G. Manimaran. Vulnerability assessment of cybersecurity for scada systems. *IEEE Transactions on Power Systems*, 23(4):1836–1846, 2008.

[110] A. Toshniwal, K. Mahesh, and R. Jayashree. Overview of anomaly detection techniques in machine learning. In *2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC)*, pages 808–815. IEEE, 2020.

[111] L. J. Trautman and P. C. Ormerod. Wannacry, ransomware, and the emerging threat to corporations. *Tenn. L. Rev.*, 86:503, 2018.

[112] G. Vladimir and A. Kabysh. Supporting coherence in multi-agent system: Related temporal difference with influence trace.

[113] D. Wang, X. Wang, Y. Zhang, and L. Jin. Detection of power grid disturbances and cyber-attacks based on machine learning. *Journal of information security and applications*, 46:42–52, 2019.

[114] J. Wang, D. Shi, Y. Li, J. Chen, H. Ding, and X. Duan. Distributed framework for detecting pmu data manipulation attacks with deep autoencoders. *IEEE Transactions on smart grid*, 10(4):4401–4410, 2018.

[115] Q. Wang, W. Tai, Y. Tang, M. Ni, and S. You. A two-layer game theoretical attack-defense model for a false data injection attack against power systems. *International Journal of Electrical Power & Energy Systems*, 104:169–177, 2019.

[116] R. Xu, R. Wang, Z. Guan, L. Wu, J. Wu, and X. Du. Achieving efficient detection against false data injection attacks in smart grid. *IEEE Access*, 5:13787–13798, 2017.

[117] L. Xue, B. Ma, J. Liu, and Y. Yu. Jamming attack against remote state estimation over multiple wireless channels: A reinforcement learning based game theoretical approach. *ISA transactions*, 2022.

[118] J. Yang, M. Wu, and X.-Z. Liu. Defense against adversarial attack using pca. In *International Conference on Artificial Intelligence and Security*, pages 627–636. Springer, 2020.

[119] D. Yao, M. Wen, X. Liang, Z. Fu, K. Zhang, and B. Yang. Energy theft detection with energy privacy preservation in the smart grid. *IEEE Internet of Things Journal*, 6(5):7659–7669, 2019.

[120] Z.-H. Yu and W.-L. Chin. Blind false data injection attack using pca approximation method in smart grid. *IEEE Transactions on Smart Grid*, 6(3):1219–1226, 2015.

[121] W. Yuan, J. Wang, F. Qiu, C. Chen, C. Kang, and B. Zeng. Robust optimization-based resilient distribution network planning against natural disasters. *IEEE Transactions on Smart Grid*, 7(6):2817–2826, 2016.

[122] M. Zamani, U. Helmke, and B. D. Anderson. Zeros of networked systems with time-invariant interconnections. *Automatica*, 61:97–105, 2015.

[123] S. Zanero. Cyber-physical systems. *Computer*, 50(4):14–16, 2017.

[124] H. Zhang, J. Wang, and Y. Ding. Blockchain-based decentralized and secure keyless signature scheme for smart grid. *Energy*, 180:955–967, 2019.

[125] L. Zhang, L. Zhao, S. Yin, C.-H. Chi, R. Liu, and Y. Zhang. A lightweight authentication scheme with privacy protection for smart grid communications. *Future Generation Computer Systems*, 100:770–778, 2019.

[126] T.-Y. Zhang and D. Ye. False data injection attacks with complete stealthiness in cyber–physical systems: A self-generated approach. *Automatica*, 120:109117, 2020.

[127] Y. Zhang, L. Wang, Y. Xiang, and C.-W. Ten. Power system reliability evaluation with scada cybersecurity considerations. *IEEE Transactions on Smart Grid*, 6(4):1707–1721, 2015.

[128] J. Zhao, J. Wang, and L. Yin. Detection and control against replay attacks in smart grid. In *2016 12th International Conference on Computational Intelligence and Security (CIS)*, pages 624–627. IEEE, 2016.

[129] K. Zheng, Q. Chen, Y. Wang, C. Kang, and Q. Xia. A novel combined data-driven approach for electricity theft detection. *IEEE Transactions on Industrial Informatics*, 15(3):1809–1819, 2018.

[130] K. Zheng, Y. Wang, Q. Chen, and Y. Li. Electricity theft detecting based on density-clustering method. In *2017 IEEE Innovative Smart Grid Technologies-Asia (ISGT-Asia)*, pages 1–6. IEEE, 2017.

[131] R. D. Zimmerman, C. E. Murillo-Sánchez, and R. J. Thomas. Matpower: Steady-state operations, planning, and analysis tools for power systems research and education. *IEEE Transactions on power systems*, 26(1):12–19, 2010.

[132] Y. Zou, J. Zhu, X. Wang, and V. C. Leung. Improving physical-layer security in wireless communications using diversity techniques. *IEEE Network*, 29(1):42–48, 2015.

Appendix A

Appendix

Matlab Code

```matlab
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Thesis simulation
%%%%%%%%Elham Honarvar and Abolghasem Daeichian
clear all;
clc;
close all
rng('default')
s=rng;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Defining system parameters
K = 23; N = 13;
sigvsq = 1e-4; sigwsq = 2e-4;
load H;
load A;
load new_angles;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Defining simulation parameters
no_episodes      =4e4;          % number of training episodes (need to be chosen higher for better learning)
% Win_Data          = 30-1;     % The number of data samples to be considered in PCA
NofPCAComponent = 10;          % The number of principal components to be considered
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Parameters of Euclidean detector
TestResult_euc_F_kalman=[];
TestResult_euc_F_pca=[];
TestResult_cos_F_kalman=[];
TestResult_cos_F_pca=[];

TestResult_euc_precision_kalman=[];
TestResult_euc_precision_pca=[];
TestResult_cos_precision_kalman=[];
```

```
TestResult_cos_precision_pca=[];
for Win_Data=10:5:40
% for attack_covariance=0.01:0.005:0.05
% for attack_domain=0.01:0.005:0.05
euc_thresh_all       = 0.01:0.005:0.23;
euc_sum_delay_pca    = zeros(1,length(euc_thresh_all));
euc_cnt_add_pca      = zeros(1,length(euc_thresh_all));
euc_cnt_falarm_pca   = zeros(1,length(euc_thresh_all));
euc_detected_pca     = zeros(1,length(euc_thresh_all));

euc_sum_delay_kalman    = zeros(1,length(euc_thresh_all));
euc_cnt_add_kalman      = zeros(1,length(euc_thresh_all));
euc_cnt_falarm_kalman   = zeros(1,length(euc_thresh_all));
euc_detected_kalman     = zeros(1,length(euc_thresh_all));
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Parameters of cosine-similarity
cos_thresh_all       = 5e-7:5e-7:42e-6;

cos_sum_delay_pca    = zeros(1,length(cos_thresh_all));
cos_cnt_add_pca      = zeros(1,length(cos_thresh_all));
cos_cnt_falarm_pca   = zeros(1,length(cos_thresh_all));
cos_detected_pca     = zeros(1,length(cos_thresh_all));

cos_sum_delay_kalman    = zeros(1,length(cos_thresh_all));
cos_cnt_add_kalman      = zeros(1,length(cos_thresh_all));
cos_cnt_falarm_kalman   = zeros(1,length(cos_thresh_all));
cos_detected_kalman     = zeros(1,length(cos_thresh_all));
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Episodes
fwaitbar=waitbar(0,'Please wait...');
for i = 1:no_episodes
    waitbar(i/no_episodes,fwaitbar,'Please wait...')
    t = 1;
    terminal = 0;
    P = eye(N);
    G = zeros(N,K);
    x = new_angles;      % initial state parameters obtained in Matpower
    hx0_m = new_angles; % initialization of the state estimates

    rho                  = 1e-2 + 9e-2*rand(1);  % rho is uniform r.v. U
        [0.0001,0.001]
    attack_time          = Win_Data+1 + geornd(rho);  % attack launch time
        is geometric r.v. with parameter rho
    MaxAdmissDelayInDetection =20;  % maximum tolerable detection delay
         (if not detected, then missed)
    max_time            = attack_time + MaxAdmissDelayInDetection;    % max. duration of an episode (if "stop" is chosen, then episode is terminated)

    euc_tmpadd_pca                = zeros(1,length(euc_thresh_all));
    euc_flag_det_delay_pca        = zeros(1,length(euc_thresh_all));
    euc_flag_falarm_pca           = zeros(1,length(euc_thresh_all));
    cos_tmpadd_pca                = zeros(1,length(cos_thresh_all));
    cos_flag_det_delay_pca        = zeros(1,length(cos_thresh_all));
    cos_flag_falarm_pca           = zeros(1,length(cos_thresh_all));      %it remains zero
    euc_tmpadd_kalman             = zeros(1,length(euc_thresh_all));    %it remains zero
    euc_flag_det_delay_kalman     = zeros(1,length(euc_thresh_all));
    euc_flag_falarm_kalman        = zeros(1,length(euc_thresh_all));
    cos_tmpadd_kalman             = zeros(1,length(cos_thresh_all));    %it remains zero
    cos_flag_det_delay_kalman     = zeros(1,length(cos_thresh_all));
    cos_flag_falarm_kalman        = zeros(1,length(cos_thresh_all));

    while (t <= max_time && terminal == 0)
        %%%%%%%%%%%%%%%%%%%%%% generating data
        % State update at time t
        v = mvnrnd(zeros(N,1),sigvsq*eye(N))';
        x = A*x + v;
        w = mvnrnd(zeros(K,1),sigwsq*eye(K))';
        if (attack_time<=t) % Under attack
            ddd = randn(1);
            sgn = 1*(ddd>0) - 1*(ddd<=0);
            %%%%%%%%%%%%%%%%different false data and different attack domains
          fdata = sgn*(0.02 + 0.04*rand(K,1)); % false data(basic)
%         fdata = sgn*(attack_domain + 0.04*rand(K,1)); % false data(basic)
%         fdata = sgn*(0.02 + attack_covariance*rand(K,1));
            %%%%%%%%%%%%%%%%different jamming noise
            ss = (randn(1) > 0)*2e-4; % with prob. 0.5, jamming occurs(basic)
            %ss = 1e-3; 0; 5e-4;
            %ss = sqrt(8e-5);
            %sigma = (ss+ss*rand(K,1));    % jamming noise variance
            tmp_mtrx = ss*randn(K);
            tmp_cov = tmp_mtrx*tmp_mtrx';
            jamm_noise = mvnrnd(zeros(K,1),tmp_cov)';%(basic)
            %jamm_noise = mvnrnd(zeros(K,1),diag(sigma))'; % correlated (over space) jamming noise
%             y(:,t) = H*x + w + fdata ;
%             y(:,t) = H*x + w +jamm_noise;
             y(:,t) = H*x + w + fdata + jamm_noise; %(basic)
        else  % Normal operation
             y(:,t) = H*x + w;
        end
        if (t<Win_Data+1)
            t=t+1;
            continue;
        end
        %%%%%%%%%%%%%%%%%%%%%%%%%% Implementation of kalman filter
        % Prediction Step of Kalman Filter
        hx0_p = A*hx0_m;
        P = A*P*A' + sigvsq*eye(N);            %P is F in the article
        % Measurement Update Step of Kalman Filter
        G = P*H'/(H*P*H'+(sigwsq)*eye(K));
        hx0_m = hx0_p + G*( y(:,t)-H*hx0_p);
        P = P - G*H*P;
        yhat=H*hx0_m;
        %%%%%%%%%%%%%%%%%%%%%%%%%% Implementation of PCA
        %Covariance matrix(the eigen values and eigen vectors of C helps us to find principle components)
        Cov_y=cov(y(:,t-Win_Data:t));
        [EVec_y,EVal_y]=eig(Cov_y);
        % selecting the major principal components
        [EVal_max(:,t),EVal_max_ind]=maxk(diag(EVal_y),NofPCAComponent);
        %u is transforming vector which is the second column of V(u=V(:,2))
        u_transform_max=EVec_y(:,EVal_max_ind);
        %%%%%%%%%%%%%%%%%%%%%%%%%% Transform Data
        z1=y(:,t-Win_Data:t)*u_transform_max;
        %Decode data(Y has same size as y)
        %Y is the projected value of our original data which has largest possible varoance of the data
        yPCA=z1* u_transform_max';
        yPCA=yPCA(:,end);
        %%%%%%%%%%%%%%%%%%%%%%%%% Error of kalman measurement estimation
        e_kalman(1,t) = norm(y(:,t)-yhat);
        %%%%%%%%%%%%%%%%%%%%%%%%% Error of PCA measurement estimation
        e_PCA(1,t)  = norm(y(:,t)-yPCA);
        %%%%%%%%%%%%%%%%%%%%%% Euclidean detector for pca
        euc_stat_pca                = norm(y(:,t)-yPCA,2);
        euc_flag_falarm_pca         = euc_flag_falarm_pca    + (euc_flag_falarm_pca == 0).*(euc_stat_pca >= euc_thresh_all)*(t < attack_time);
        euc_tmpadd_pca              = euc_tmpadd_pca + (t-attack_time)*(euc_flag_falarm_pca == 0).*(euc_flag_det_delay_pca == 0).*(euc_stat_pca >= euc_thresh_all)*(t >= attack_time);
        euc_flag_det_delay_pca      = euc_flag_det_delay_pca+ (euc_flag_falarm_pca == 0).*(euc_flag_det_delay_pca == 0).*(euc_stat_pca >= euc_thresh_all)*(t >= attack_time);
        euc_cond_pca                = euc_flag_falarm_pca(length(euc_flag_falarm_pca)) || euc_flag_det_delay_pca(length(euc_flag_det_delay_pca));
        %%%%%%%%%%%%%%%%%%%%%%% Euclidean detector for kalman
        euc_stat_kalman             = norm(y(:,t)-yhat,2);
        euc_flag_falarm_kalman      = euc_flag_falarm_kalman    + (euc_flag_falarm_kalman == 0).*(euc_stat_kalman >= euc_thresh_all)*(t < attack_time);
        euc_tmpadd_kalman           = euc_tmpadd_kalman          + (t-attack_time)*(euc_flag_falarm_kalman == 0).*(euc_flag_det_delay_kalman == 0).*(euc_stat_kalman >= euc_thresh_all)*(t >= attack_time);
        euc_flag_det_delay_kalman = euc_flag_det_delay_kalman+ (euc_flag_falarm_kalman == 0).*(euc_flag_det_delay_kalman == 0).*(euc_stat_kalman >= euc_thresh_all)*(t >= attack_time);
        euc_cond_kalman             = euc_flag_falarm_kalman(length(euc_flag_falarm_kalman)) ||  euc_flag_det_delay_kalman(length(euc_flag_det_delay_kalman));
        %%%%%%%%%%%%%%%%%%%%%%% cosine detector for PCA
        cos_stat_pca              = 1 - (y(:,t)'*yPCA) / ( norm(y(:,t),2)*norm(yPCA,2) );
        %cos_stat_pca2(1,t)       = 1 - (y(:,t)'*yPCA) / ( norm(y(:,t),2)*norm(yPCA,2) );
        cos_flag_falarm_pca       = cos_flag_falarm_pca     + (cos_flag_falarm_pca == 0).*(cos_stat_pca >= cos_thresh_all)*(t < attack_time);
        cos_tmpadd_pca            = cos_tmpadd_pca          + (t-attack_time)*(cos_flag_falarm_pca == 0).*(cos_flag_det_delay_pca == 0).*(cos_stat_pca >= cos_thresh_all)*(t >= attack_time);
        cos_flag_det_delay_pca    = cos_flag_det_delay_pca + (
  cos_flag_falarm_pca == 0).*(cos_stat_pca >= cos_thresh_all).*(cos_flag_det_delay_pca == 0)*(t >= attack_time);
        cos_cond_pca              = cos_flag_falarm_pca(length(cos_flag_falarm_pca)) || cos_flag_det_delay_pca(length(cos_flag_det_delay_pca));
        %%%%%%%%%%%%%%%%%%%%%%%% cosine detector for kalman
        cos_stat_kalman              = 1 - (y(:,t)'*yhat) / ( norm(y(:,t),2)*norm(yhat,2) );
        %cos_stat_kalman2(1,t)        = 1 - (y(:,t)'*yhat) / ( norm(y(:,t),2)*norm(yhat,2) );
        cos_flag_falarm_kalman       = cos_flag_falarm_kalman      + (cos_flag_falarm_kalman == 0).*(cos_stat_kalman >= cos_thresh_all)*(t < attack_time);
        cos_tmpadd_kalman            = cos_tmpadd_kalman           + (t-attack_time)*(cos_flag_falarm_kalman == 0).*(cos_flag_det_delay_kalman == 0).*(cos_stat_kalman >= cos_thresh_all)*(t >= attack_time);
        cos_flag_det_delay_kalman = cos_flag_det_delay_kalman + (cos_flag_falarm_kalman == 0).*(cos_stat_kalman >= cos_thresh_all).*(cos_flag_det_delay_kalman == 0)*(t >= attack_time);
        cos_cond_kalman              = cos_flag_falarm_kalman(length(cos_flag_falarm_kalman)) || cos_flag_det_delay_kalman(length(cos_flag_det_delay_kalman));

        if ( euc_cond_pca && cos_cond_pca && euc_cond_kalman && cos_cond_kalman)
            terminal = 1;
        end

        % Time update
        t = t + 1;
    end
  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Euclidean and cosine similarity detector for pca
    euc_cnt_falarm_pca   = euc_cnt_falarm_pca + euc_flag_falarm_pca;
    euc_detected_pca     = euc_detected_pca + euc_flag_det_delay_pca;
    %if ( euc_flag_det_delay_pca(end) == 1 )
    euc_sum_delay_pca = euc_sum_delay_pca + euc_tmpadd_pca;
    euc_cnt_add_pca = euc_cnt_add_pca + euc_flag_det_delay_pca;
   % end

    cos_cnt_falarm_pca = cos_cnt_falarm_pca + cos_flag_falarm_pca;
```

```
        cos_detected_pca = cos_detected_pca + cos_flag_det_delay_pca;
        %if ( cos_flag_det_delay_pca(end) == 1 )
        cos_sum_delay_pca = cos_sum_delay_pca + cos_tmpadd_pca;
        cos_cnt_add_pca = cos_cnt_add_pca + cos_flag_det_delay_pca;
        %end
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Euclidean and cosine similarity
            detector for kalman
        euc_cnt_falarm_kalman = euc_cnt_falarm_kalman +
            euc_flag_falarm_kalman;
        euc_detected_kalman = euc_detected_kalman +
            euc_flag_det_delay_kalman;
        %if ( euc_flag_det_delay_kalman(end) == 1 )
        euc_sum_delay_kalman = euc_sum_delay_kalman + euc_tmpadd_kalman;
        euc_cnt_add_kalman = euc_cnt_add_kalman + euc_flag_det_delay_kalman
            ;
        %end

        cos_cnt_falarm_kalman = cos_cnt_falarm_kalman +
            cos_flag_falarm_kalman;
        cos_detected_kalman = cos_detected_kalman +
            cos_flag_det_delay_kalman;
         %if ( cos_flag_det_delay_kalman(end) == 1 )
        cos_sum_delay_kalman = cos_sum_delay_kalman + cos_tmpadd_kalman;
        cos_cnt_add_kalman = cos_cnt_add_kalman + cos_flag_det_delay_kalman
            ;
        % end
end
close(fwaitbar)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Euclidean and cosine similarity
    detector  for pca
euc_pfa_pca = euc_cnt_falarm_pca/no_episodes;   % probability of false
    alarm for all thresholds
euc_cnt_add_pca(euc_cnt_add_pca==0) = 1;
euc_add_pca = euc_sum_delay_pca./euc_cnt_add_pca;   % average detection
     delay
euc_mdr_pca = (no_episodes- euc_detected_pca - euc_cnt_falarm_pca )/
    no_episodes; % miss detection ratio
euc_precision_pca = euc_detected_pca./(euc_detected_pca +
    euc_cnt_falarm_pca);
euc_recall_pca = euc_detected_pca./(no_episodes - euc_cnt_falarm_pca);
euc_F_pca = (2*euc_precision_pca.*euc_recall_pca)./(euc_precision_pca +
     euc_recall_pca);

```

```
cos_pfa_pca = cos_cnt_falarm_pca/no_episodes;
cos_cnt_add_pca(cos_cnt_add_pca==0) = 1;
cos_add_pca = cos_sum_delay_pca./cos_cnt_add_pca;
cos_mdr_pca = (no_episodes- cos_detected_pca - cos_cnt_falarm_pca )/
    no_episodes;
cos_precision_pca = cos_detected_pca./(cos_detected_pca +
    cos_cnt_falarm_pca);
cos_recall_pca = cos_detected_pca./(no_episodes - cos_cnt_falarm_pca);
cos_F_pca = (2*cos_precision_pca.*cos_recall_pca)./(cos_precision_pca +
     cos_recall_pca);

disp('==========================================================')
tmp = length(cos_thresh_all);
fmt=['pca Cos Average detection delay:     ' repmat('%1.3f ',1,tmp) '\n'
    ];
fprintf(fmt,cos_add_pca);
fmt=['pca Cos Probability of false alarm: ' repmat('%1.3f ',1,tmp) '\n'
    ];
fprintf(fmt,cos_pfa_pca);
fmt=['pca Cos Miss detection ratio:        ' repmat('%1.3f ',1,tmp) '\n'
    ];
fprintf(fmt,cos_mdr_pca);
fmt=['pca Cos Precision:                   ' repmat('%1.3f ',1,tmp) '\n'
    ];
fprintf(fmt,cos_precision_pca);
fmt=['pca Cos Recall:                      ' repmat('%1.3f ',1,tmp) '\n'
    ];
fprintf(fmt,cos_recall_pca);
fmt=['pca Cos F-score:                     ' repmat('%1.3f ',1,tmp) '\n'
    ];
fprintf(fmt,cos_F_pca);

disp('==========================================================')
tmp = length(euc_thresh_all);
fmt=['pca Euc Average detection delay:     ' repmat('%1.3f ',1,tmp) '\n'
    ];
fprintf(fmt,euc_add_pca);
fmt=['pca Euc Probability of false alarm: ' repmat('%1.3f ',1,tmp) '\n'
    ];
fprintf(fmt,euc_pfa_pca);
fmt=['pca Euc Miss detection ratio:        ' repmat('%1.3f ',1,tmp) '\n'
    ];
fprintf(fmt,euc_mdr_pca);
```

```
fmt=['pca Euc Precision:                    ' repmat('%1.3f ',1,tmp) '\n'];
fprintf(fmt,euc_precision_pca);
fmt=['pca Euc Recall:                       ' repmat('%1.3f ',1,tmp) '\n'];
fprintf(fmt,euc_recall_pca);
fmt=['pca Euc F-score:                      ' repmat('%1.3f ',1,tmp) '\n'];
fprintf(fmt,euc_F_pca);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Euclidean and cosine similarity detector for kalman
cos_pfa_kalman = cos_cnt_falarm_kalman/no_episodes;
cos_cnt_add_kalman(cos_cnt_add_kalman==0) = 1;
cos_add_kalman = cos_sum_delay_kalman./cos_cnt_add_kalman;
cos_mdr_kalman = (no_episodes - cos_cnt_falarm_kalman - cos_detected_kalman)/no_episodes;
cos_precision_kalman = cos_detected_kalman./(cos_detected_kalman + cos_cnt_falarm_kalman);
cos_recall_kalman = cos_detected_kalman./(no_episodes - cos_cnt_falarm_kalman);
cos_F_kalman = (2*cos_precision_kalman.*cos_recall_kalman)./(cos_precision_kalman + cos_recall_kalman);

euc_pfa_kalman = euc_cnt_falarm_kalman/no_episodes;   % probability of false alarm for all thresholds
euc_cnt_add_kalman(euc_cnt_add_kalman==0) = 1;
euc_add_kalman = euc_sum_delay_kalman./euc_cnt_add_kalman;   % average detection delay
euc_mdr_kalman = (no_episodes- euc_detected_kalman - euc_cnt_falarm_kalman)/no_episodes; % miss detection ratio
euc_precision_kalman = euc_detected_kalman./(euc_detected_kalman + euc_cnt_falarm_kalman);
euc_recall_kalman = euc_detected_kalman./(no_episodes - euc_cnt_falarm_kalman);
euc_F_kalman = (2*euc_precision_kalman.*euc_recall_kalman)./(euc_precision_kalman + euc_recall_kalman);

disp('============================================================')
tmp = length(cos_thresh_all);
fmt=['kalman Cos Average detection delay:     ' repmat('%1.3f ',1,tmp) '\n'];
fprintf(fmt,cos_add_kalman); % average detection delay
fmt=['kalman Cos Probability of false alarm: ' repmat('%1.3f ',1,tmp) '
\n'];
fprintf(fmt,cos_pfa_kalman);
fmt=['kalman Cos Miss detection ratio:          ' repmat('%1.3f ',1,tmp) '\n'];
fprintf(fmt,cos_mdr_kalman);
fmt=['kalman Cos Precision:                     ' repmat('%1.3f ',1,tmp) '\n'];
fprintf(fmt,cos_precision_kalman);
fmt=['kalman Cos Recall:                        ' repmat('%1.3f ',1,tmp) '\n'];
fprintf(fmt,cos_recall_kalman);
fmt=['kalman Cos F-score:                       ' repmat('%1.3f ',1,tmp) '\n'];
fprintf(fmt,cos_F_kalman);

disp('============================================================')
tmp = length(euc_thresh_all);
fmt=['kalman Euc Average detection delay:       ' repmat('%1.3f ',1,tmp) '\n'];
fprintf(fmt,euc_add_kalman);% average detection delay
fmt=['kalman Euc Probability of false alarm:    ' repmat('%1.3f ',1,tmp) '\n'];
fprintf(fmt,euc_pfa_kalman);
fmt=['kalman Euc Miss detection ratio:          ' repmat('%1.3f ',1,tmp) '\n'];
fprintf(fmt,euc_mdr_kalman);
fmt=['kalman Euc Precision:                     ' repmat('%1.3f ',1,tmp) '\n'];
fprintf(fmt,euc_precision_kalman);
fmt=['kalman Euc Recall:                        ' repmat('%1.3f ',1,tmp) '\n'];
fprintf(fmt,euc_recall_kalman);
fmt=['kalman Euc F-score:                       ' repmat('%1.3f ',1,tmp) '\n'];
fprintf(fmt,euc_F_kalman);

TestResult_euc_F_kalman=[TestResult_euc_F_kalman;euc_F_kalman];
TestResult_euc_F_pca=[TestResult_euc_F_pca;euc_F_pca];
TestResult_cos_F_kalman=[TestResult_cos_F_kalman;cos_F_kalman];
TestResult_cos_F_pca=[TestResult_cos_F_pca;cos_F_pca];

TestResult_euc_precision_kalman=[TestResult_euc_precision_kalman;
    euc_precision_kalman];
```

```
TestResult_euc_precision_pca=[TestResult_euc_precision_pca;
    euc_precision_pca];
TestResult_cos_precision_kalman=[TestResult_cos_precision_kalman;
    cos_precision_kalman];
TestResult_cos_precision_pca=[TestResult_cos_precision_pca;
    cos_precision_pca];

end
%% Plots
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Plot Average Detection delay
figure('Units','Centimeters', 'position' , [ 2 5 7.5 9]);
subplot(211), hold on, box on , grid off
yyaxis left
plot(euc_thresh_all,[euc_add_pca;euc_add_kalman],'b')
ylabel('Average Detection delay')
yyaxis right
plot(euc_thresh_all,[euc_recall_pca;euc_recall_kalman],'r')
ylabel('Recall')
title('Euclidean detector')
xlabel('Thresholds')
xlim([euc_thresh_all(1) euc_thresh_all(end)])

subplot(212), hold on, box on , grid off
yyaxis left
plot(cos_thresh_all,[cos_add_pca;cos_add_kalman],'b')
ylabel('Average Detection delay')
yyaxis right
plot(cos_thresh_all,[cos_recall_pca;cos_recall_kalman],'r')
ylabel('Recall')
ylim([0 1.05])
title('Cosine similarity detector')
xlabel('Thresholds')
xlim([cos_thresh_all(1) cos_thresh_all(end)])
%%%%%%%%%%%%%%%%%%%%%%%%%%%% Plot F-score & Miss detection ratio
figure('Units','Centimeters', 'position' , [ 10 5 7.5 9]);
subplot(211), hold on, box on , grid off
yyaxis left
plot(euc_thresh_all,[euc_F_pca;euc_F_kalman],'b')
ylabel('F-Score')
ylim([0 1.05])
yyaxis right
plot(euc_thresh_all,[euc_mdr_pca;euc_mdr_kalman],'r')
ylabel('Miss detection ratio')
```

```
ylim([0 1.05])
xlabel('Thresholds')
title('Euclidean detector')
xlim([euc_thresh_all(1) euc_thresh_all(end)])

subplot(212),hold on, box on , grid off
yyaxis left
plot(cos_thresh_all,[cos_F_pca;cos_F_kalman],'b')
ylabel('F-Score')
ylim([0 1.05])
yyaxis right
plot(cos_thresh_all,[cos_mdr_pca;cos_mdr_kalman],'r')
ylabel('Miss detection ratio')
ylim([0 1.05])
xlabel('Thresholds')
title('Cosine-similarity detector')
xlim([cos_thresh_all(1) cos_thresh_all(end)])

%%%%%%%%%%%%%%%%%%%%%% Plot Precision & Probability of false alarm
figure('Units','Centimeters', 'position' , [ 18 5 7.5 9]);
subplot(211), hold on, box on , grid off
yyaxis left
plot(euc_thresh_all,[euc_precision_pca;euc_precision_kalman],'b')
ylabel('Precision')
ylim([-0.05 1.05])
yyaxis right
plot(euc_thresh_all,[euc_pfa_pca;euc_pfa_kalman],'r')
ylabel('Prob. of false alarm')
xlabel('Thresholds')
title('Euclidean detector')
ylim([-0.05 1.05])
xlim([euc_thresh_all(1) euc_thresh_all(end)])

subplot(212), hold on, box on , grid off
yyaxis left
plot(cos_thresh_all,[cos_precision_pca;cos_precision_kalman],'b')
ylabel('Precision')
ylim([-0.05 1.05])
yyaxis right
plot(cos_thresh_all,[cos_pfa_pca;cos_pfa_kalman],'r')
ylabel('Prob. of false alarm')
xlabel('Thresholds')
title('Cosine similarity detector')
```

```
ylim([-0.05 1.05])
xlim([cos_thresh_all(1) cos_thresh_all(end)])
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%window size
Win_Data=10:5:40
fontSize=12

figure('Units','Centimeters', 'position' , [ 18 5 7.5 9]);
box on
ax1 = nexttile;
n1=contourf(euc_thresh_all,Win_Data,TestResult_euc_F_pca);
colormap ("default");
colorbar
xlabel(sprintf('Euclidean Thresholds'));
ylabel(sprintf('PCA window length'));


figure('Units','Centimeters', 'position' , [ 18 5 7.5 9]);
box on
ax2 = nexttile;
m1=contourf(cos_thresh_all,Win_Data,TestResult_cos_F_pca);
colormap ("default");
colorbar
xlabel(sprintf('Cos-Sim Thresholds'));
ylabel(sprintf('PCA window length'));
```

Acknowledgements

I would like to express my gratitude to my co-supervisor, Professor Abolghasem Daeichian, who guided me throughout this project. I would also like to show my deep appreciation to Andrea Tortorelli, Francesco Liberati and Professor Francesco Delli Priscoli who helped me finalize my project. I wish to extend my special thanks to my friends and family who supported me and offered deep insight into the study.

Elham Honarvar, Rome, October 2022