Ministry of Higher Education and Scientific Research

وزارة التعليم العالي والبحث العلمي

Badji Mokhtar Annaba University

Faculty of Technology

Department electronics

جامعة باجي مختار – عنابـــة

كـليـة التكنـولـوجيـا

قســم الكترونيك

Thesis

Submitted to obtain the diploma of

Doctorate Third Cycle

Field: Automatic

Specialty: Automatic and Signals

By:

BOUGHELOUM Wafa

Title :

Diagnostic de fonctionnement des systèmes dynamiques par analyse en composantes principales non linéaires

Thesis defended on March,10, 2024 in front of the jury composed of:

N°	Name and Surname	Grade	Institution	Quality
01	BAHI TAHAR	Prof.	University Badji Mokhtar -Annaba	President
02	RAMDANI MESSAOUD	Prof.	University Badji Mokhtar -Annaba	Supervisor
03	CHENIKHER SALAH	Prof.	University of Tebessa	Examiner
04	BENSAOULA SALAH	MCA	University Badji Mokhtar -Annaba	Examiner

It is with deep gratitude that I dedicate this humble work:

To my dear parents who have always encouraged and supported me from my early days of study until this day. For their constant sacrifices, wise advice, and unwavering support. May God bless them with good health and long life.

To my sister Loubna and my brothers Nabil and Imed, for their affection, understanding, and patience. For the good times we have shared together.

To my sister Mouna and my brother-in-law Karim, To my brother Amine, my sister-in-law Amira, and the little princess Miral, for their infinite support and presence.

To my dear heart sisters Sonia and Anissa, for their tenderness, companionship, and presence despite the distance.

To my friends Besma, Chahra, Asma, Asma and Asma for their supports.

To my dear friend Moh, for his help and support. For the courage he has given me during these recent times.

To my close friend Imed.

To all my family, all my friends, and all those who are close to my heart and whose names I have not mentioned.

Remerciements

I would like to express my deep gratitude and sincere thanks to all the people who have contributed directly or indirectly to the completion of my doctoral thesis.

First and foremost, I would like to thank my thesis advisor, **Professor Messaoud RAMDANI**, for his guidance, expertise, and support throughout this research journey. His valuable advice, availability, and patience have been instrumental in the success of this work.

I thank **Professor CHENIKHER Salah** from Tebessa University for his interest in examining this thesis and for the honor of being part of the jury.

I would also like to express my gratitude to **Professor Tahar BAHI** from Badji Mokhtar Annaba University for presiding over the thesis jury. I extend my appreciation to **Doctor Salah BENSAOULA** from Badji Mokhtar Annaba University for accepting to be the reviewer of my work.

My thanks also go to Doctor **BEKAIK Mounir** for his collaboration, insightful discussions, and valuable ideas. His encouragement has greatly contributed to improving the quality of my research.

I would like to express my gratitude to my family and friends for their unwavering support, constant encouragement, and understanding throughout this journey. Their presence and moral support have been sources of comfort and motivation during challenging times.

Finally, I would like to extend my heartfelt thanks to all the individuals who participated in my research as participants or provided necessary resources, data, or information for my work.

I am aware that this thesis would not have been possible without the support and contribution of each and every one of you. Your encouragement, trust, and support have been essential in the realization of this research project.

In conclusion, I am deeply grateful to all the individuals who have played a role in the completion of my doctoral thesis. Your unwavering support, expertise, and encouragement have contributed to my academic journey and shaped my professional trajectory.

Thank you infinitely to all those who have been by my side throughout this experience. Your contribution will forever be engraved in my memory and in the accomplishment of this research work.

ملخص

المتناثر المكدس البيانات مشفر على يعتمد مبتكر نهج باستخدام الديناميكية الأنظمة وتشغيل تشخيص حول العمل هذا يتمحور وتشخيص مراقبة إلى البيولوجية والأنظمة الصناعية والعمليات الكهربائية الشبكات مثل الديناميكية الأنظمة تحتاج .(SSAE) البُعد لتقليل تستخدم شائعة إحصائية تقنية (PCA) الرئيسية المكونات تحليل يُعتبر .أدائها وتحسين سلامتها لضمان دقيق وليس فقط الخطية العلاقات لالتقاط تُستخدمان المتناثر PCA و PCA فإن ،ذلك ومع .أهمية الأكثر المعلومات واستخلاص .الديناميكية الأنظمة في الموجودة الخطية غير العلاقات

اصطناعية عصبية شبكة وهو ،(SSAE) المتناثر المكدس البيانات مشفر استخدام على يعتمد الرسالة هذه في المقترح النهج لنمذجة أساسية تعد والتي ،للبيانات الخطية وغير التسلسلية التمثيلات تعلم من المتناثر المكدس البيانات مشفر يتمكن .عميقة مما ،التجريد تتزايد ميزات استخراج من المتناثر المكدس البيانات لمشفر المكدسة الطبقات تمكن .الديناميكية الأنظمة تعقيد .البيانات في المعنوية والتصاميم الأنماط اكتشاف يسهل

يتم ،أولاً .الديناميكية الأنظمة تشخيص في (SSAE) المتناثر المكدس البيانات مشفر لتطبيق شاملة منهجية الرسالة هذه تقدم البيانات مشفر تدريب يتم ثم .الشاذة والقيم الضوضاء لإزالة مسبقًا معالجتها يتم ثم ،مناسبة حساسات باستخدام البيانات جمع المتن المكدس

مشفر بواسطة عليها الحصول تم التي النتائج استخدام يتم .والمفيدة المدمجة التمثيلات لتعلم مسبقًا المعالجة البيانات على اثر يتم .الأداء وتحسين ،المحتمل الفشل وتوقع ،الأعطال اكتشاف مثل ،الديناميكية الأنظمة لتشخيص المتناثر المكدس البيانات المكدس البيانات مشفر وقوة فعالية النتائج تظهر .المياه معالجة ونظام تخيلي مثال خلال من المقترحة المنهجية من التحقق صيانة بشأن مستنيرة قرارات واتخاذ المشكلات عن المبكر الكشف من يمكن مما ،الديناميكية الأنظمة تشخيص في المتناثر .الأنظمة وتحسين

وتشغيل تشخيص في (SSAE) المتناثر المكدس البيانات لمشفر الواعد الاستخدام على الضوء العمل هذا يسلط ،الختام في لتحليل قوية طريقة يوفر مما ،البيانات في الخطية وغير المعقدة العلاقات بالتقاط المستخدم النهج يسمح .الديناميكية الأنظمة .المعقدة الديناميكية الأنظمة وإدارة مراقبة لتحسين جديدة آفاقًا المتحققة النتائج تفتح .الديناميكية الأنظمة ومراقبة

الدالة الكلمات :مراقبة العمليات ،التحكم الإحصائي لعمليات متعددة متغيرات ،التشخيص ،الرئيسية المكونات تحليل ،الرئيسية المكونات ،المتناثرة الرئيسية المكونات ،مشفر البيانات مكدس المتناثر ،بناء إعادة ،البيانات

ABSTRACT

This work focused on the diagnosis of dynamic systems based on multivariate statistical process monitoring (MSPM) approach, namely the Stacked Sparse Autoencoders (SSAE). Dynamic systems, such as electrical networks, industrial processes and biological systems, require accurate monitoring and diagnosis to ensure correct operation and safety. Principal Component Analysis (PCA) is a commonly used statistical technique to reduce the dimensionality of data and extract the most significant and relevent information. However, global linear PCA and Sparse PCA are only able to capture linear relationships and not take into account the nonlinear relationships and correlation present in dynamic systems which can be interpreted as non linear principal component analysis.

The proposed approach in this thesis is based on the use of Stacked Sparse Autoencoders (SSAE), which are deep artificial neural networks. SSAE is capable of learning hierarchical and nonlinear representations of data, which is essential for modeling the complexity of dynamic systems. The stacked layers of SSAE enable the progressive extraction of increasingly abstracted features, facilitating the detection of significant patterns and motifs in the data.

This work presents a comprehensive methodology for the application of SSAE in the diagnosis of dynamic systems. Firstly, data is collected using appropriate sensors, and then it is preprocessed to remove noise and outliers. Next, SSAE is trained on the preprocessed data to learn compact and informative representations. The results obtained by SSAE are then used for the diagnosis of dynamic systems, including anomaly detection, prediction of potential failures, and performance optimization. The proposed methodology is validated through synthetic examples and a water treatment process. The results demonstrate the effectiveness and robustness of SSAE for the diagnosis of dynamic systems, enabling early detection of problems and informed decision-making for maintenance and system optimization.

In conclusion, this work highlights the promising use of Stacked Sparse Autoencoders (SSAE) for the diagnosis and operation of dynamic systems. The SSAE approach allows capturing complex and nonlinear relationships in the data, offering a powerful method for the analysis and monitoring of dynamic systems. The obtained results open up new perspectives for enhancing the monitoring of complex dynamic systems.

Keywords: *Process Monitoring, Multivariate Statistical Process Control, Diagnosis, Principal Component Analysis, Sparse Principal Component Analysis, Stacked Sparse Autoencoder, Data Reconstruction, Dynamic systems.*

Résumé

Ce travail s'est concentré sur le diagnostic des systèmes dynamiques en utilisant une approche de surveillance de processus statistique multivarié (MSPM), plus précisément les Autoencodeurs Empilés et Épars (SSAE). Les systèmes dynamiques tels que les centrales électriques, les procédés industriels et les systèmes biologiques nécessitent une surveillance et un diagnostic précis afin d'assurer un fonctionnement correct et sûr. L'Analyse en Composantes Principales (ACP) est une technique statistique couramment utilisée pour réduire la dimensionnalité des données et extraire les informations les plus significatives et pertinentes. Cependant, l'ACP linéaire globale et l'ACP éparse ne sont capables de capturer que les relations linéaires et ne tiennent pas compte des relations non linéaires et des corrélations présentes dans les systèmes dynamiques, ce qui peut être interprété comme une analyse en composantes principales non linéaire. L'approche proposée dans cette thèse est basée sur l'utilisation des Autoencodeurs Empilés et Épars (SSAE), qui sont des réseaux de neurones artificiels profonds. Les SSAE sont capables d'apprendre des représentations hiérarchiques et non linéaires des données, ce qui est essentiel pour modéliser la complexité des systèmes dynamiques. Les couches empilées des SSAE permettent l'extraction progressive de caractéristiques de plus en plus abstraites, facilitant la détection de motifs et de motifs significatifs dans les données. Ce travail présente une méthodologie complète pour l'application des SSAE dans le diagnostic des systèmes dynamiques. Tout d'abord, les données sont collectées à l'aide de capteurs appropriés, puis elles sont prétraitées pour éliminer le bruit et les valeurs aberrantes. Ensuite, les SSAE sont entraînés sur les données prétraitées afin d'apprendre des représentations compactes et informatives. Les résultats obtenus par les SSAE sont ensuite utilisés pour le diagnostic des systèmes dynamiques, y compris la détection d'anomalies, la prédiction de pannes potentielles et l'optimisation des performances. La méthodologie proposée est validée à l'aide d'exemples synthétiques et d'un processus de traitement de l'eau. Les résultats démontrent l'efficacité et la robustesse des SSAE pour le diagnostic des systèmes dynamiques, permettant une détection précoce des problèmes et une prise de décision éclairée pour la maintenance et l'optimisation du système. En conclusion, ce travail met en évidence l'utilisation prometteuse des Autoencodeurs Empilés et Épars (SSAE) pour le diagnostic et le fonctionnement des systèmes dynamiques. L'approche SSAE permet de capturer des relations complexes et non linéaires dans les données, offrant ainsi une méthode puissante pour l'analyse et la surveillance des systèmes dynamiques. Les résultats obtenus ouvrent de nouvelles perspectives pour améliorer la surveillance des systèmes dynamiques complexes.

Mots-clés: *Surveillance des processus, Contrôle de processus statistique multivarié, Diagnostic, Analyse des composants principales, Analyse en composantes principales parcimonieuses, Autoencodeur clairsemé empilé, Reconstruction des données, Systémes dynamiques*.

LIST OF ABBREVIATIONS

IoT	Internet of Things
SCADA	Supervisory Control and Data Acquisition
DCS	Distributed Control System
PLC	Programmable Logic Controller
AI	Artificial Intelligence
PCA	Principal Component analysis
CPs	Principal Components
PVC	Cumulative Percentage of Variance
VRE	Variance of Reconstruction Error
SPE	Statistical Process Control Exponentially
SWE	Squared Weighted Euclidean
SPE	Squared Prediction Error
EWMA	Exponentially Weighted Moving Average
SPC	Statistical Process Control
KDE	Kernel Density Estimation
PDF	Probability Density Function
SVI	Sensor Validity Index
VNR	Variance Non-Reconstructed
SPCA	Sparse Principal Component Analysis
NLPCA	NonLinear Principal Component Analysis
LLE	Locally Linear Embedding
CNN	Convolutional Neural Networks
RNN	Recurrent Neural Networks
SSAE	Stacked Sparse Autoencoders
MSE	Mean Squared Error
SPM	Statistical Process Monitoring
AE	Autoencoder
UCLkmeans	Upper Conrol Limit based Kmeans

FCM	Fuzzy C-means
UCLFCM	Upper Conrol Limit based FCM
AUCLkmeans	Adaptif Upper Conrol Limit based Kmeans
AUCLFCM	Adaptif Upper Conrol Limit based FCM

Table of contents

List of figures **vii**

General Introduction **1**

1 STATE OF ART ON DIAGNOSIS OF INDUSTRIAL PROCESSES **3**

1.1 Introduction 3
1.2 Purpose of monitoring 4
1.3 Definitions and Concepts 5
1.4 Monitoring and diagnosis 6
1.5 Fault detection and isolation procedure 7
1.5.1 Different types of fault structures 8
1.5.2 Classification of diagnosic methods 9
1.5.3 Fault diagnosis methods 10
1.6 The performance of a diagnosis system in industry 13
1.7 Conclusion 14

2 Nonlinear Multivariate Statistical Process Monitoring **15**

2.1 Introduction 15
2.2 Principal component analysis 16
2.3 Identification of PCA model 17
2.4 Choice of the dimension of the reduced space and the number of principal components (CPs) 20
2.4.1 The cumulative percentage of variance (PCV) 20
2.4.2 The variance of reconstruction error (VRE) 21
2.4.3 Detection and localization of faults by PCA 22
2.5 Illustrative example 33
2.6 Linear extension of PCA 43
2.7 Nonlinear extension of pca 46

2.8 Conclusion 50

3 CONTRIBUTION TO STATISTICAL CONTROL OF NONLINEAR PROCESSES 51

3.1 Introduction 51
3.2 Neural networks 52
3.2.1 Activation funtiocn 54
3.2.2 Gradient descent 57
3.3 Autoencoder 58
3.3.1 The relationship between antoencoder and PCA 60
3.3.2 The parameters to define for training an autoencoder 60
3.3.3 Types of autoencoders 61
3.4 Statistical Process Monitoring (SPM) Method: Stacked Sparse Autoencoders (SSAE) 63
3.4.1 Anomaly Detection 65
3.4.2 Control Limits 66
3.4.3 Fault identification 70
3.5 Conclusion 73

4 EXPERIMENTAL RESULTS AND DISCUSSION 75

4.1 Introduction 75
4.2 Application on synthetic data 76
4.3 Case study: Application to drinking Water Treatment Plant 94
4.4 Discussion and Conclusion 107

General Conclusion 109

Bibliography 111

List of figures

1.1 The general structure of a monitoring system 7
1.2 Concept of model based fault detection and isolation 8
1.3 Classification of different fault diagnosis methods 10
1.4 Fault diagnosis methods 11

2.1 Evolution of the reconstruction variance VNR as a function of the number of components l 35
2.2 Evolution of the different measures and their estimates 37
2.3 Modeling by SPE 38
2.4 Modeling by T2 38
2.5 Modeling by SWE 39
2.6 Modeling by Combined Index 39
2.7 Fault detection by SPE 40
2.8 Fault detection by T2 40
2.9 Fault detection by SWE 41
2.10 Fault detection by Combined Index 41
2.11 Calculation of contributions for the moment (k=850) 42
2.12 Calculation of contributions for the moment (k=900) 42
2.13 PCA and NLPCA Models 47

3.1 The biological neuron 53
3.2 The artificiel neuron 54
3.3 The threshold function 55
3.4 The linear function 55
3.5 The sigmoid function 56
3.6 The Rectified Linear Unit (ReLU) function 56
3.7 The gradient descent 57
3.8 Autoencoder architecture 58

3.9 Structure of the proposed SSAE model 64
3.10 Reconstruction Principle 72

4.1 SPE: data in normal state for PCA δ_α^2 threshold 77
4.2 SPE: data in normal state for SPCA with δ_α^2 threshold 77
4.3 SPE: data in normal state for SSAE with δ_α^2 threshold 78
4.4 SPE: data in normal state for PCA with KDE threshold 78
4.5 SPE: data in normal state for SPCA with KDE threshold 79
4.6 SPE: data in normal state for PCA, SPCA and SSAE with KDE threshold . 79
4.7 SPE: data in normal state for PCA with AUCL threshold using K-means clustering 80
4.8 SPE: data in normal state for SPCA with AUCL threshold using K-means clustering 80
4.9 SPE: data in normal state for SSAE with AUCL threshold using K-means clustering 81
4.10 SPE: data in normal state for PCA with AUCL threshold using FCM 81
4.11 SPE: data in normal state for SPCA with AUCL threshold using FCM . . . 82
4.12 SPE: data in normal state for SSAE with AUCL threshold using FCM . . . 82
4.13 SPE: data in faulty state for PCA with δ_α^2 threshold 83
4.14 SPE: data in faulty state for SPCA with δ_α^2 threshold 84
4.15 SPE: data in faulty state for SSAE with δ_α^2 threshold 84
4.16 SPE: data in faulty state for PCA with KDE threshold 85
4.17 SPE: data in faulty state for SPCA with KDE threshold 85
4.18 SPE: data in faulty state for SSAE with KDE threshold 86
4.19 SPE: data in faulty state for PCA with AUCL threshold using K-means clustering 86
4.20 SPE: data in faulty state for SPCA with AUCL threshold using K-means clustering 87
4.21 SPE: data in faulty state for SSAE with AUCL threshold using K-means clustering 87
4.22 SPE:data in faulty state for PCA with AUCL threshold using FCM 88
4.23 SPE:data in faulty state for SPCA with AUCL threshold using FCM 88
4.24 SPE:data in faulty state for SSAE with AUCL threshold using FCM 89
4.25 Fault isolation using normalized contribution plots for PCA (fault in the 3rd sensor) 89
4.26 Fault isolation using normalized contribution plots for SPCA (fault in the 3rd sensor) 90

4.27 Fault isolation using normalized contribution plots for SSAE (fault in the 3rd sensor) . . . 90
4.28 Fault isolation using reconsctruction principle for SSAE using δ_α^2 (fault in the 3rd sensor) . . . 91
4.29 Fault isolation using reconsctruction principle for SSAE using KDE(fault in the 3rd sensor) . . . 92
4.30 Fault isolation using reconsctruction principle for SSAE using AUCL Kmeans(fault in the 3rd sensor) . . . 93
4.32 The typical drinking water processing units . . . 95
4.33 SPE: data in normal state for SPCA with δ_α^2 threshold . . . 97
4.34 SPE: data in normal state for SSAE with δ_α^2 threshold . . . 98
4.35 SPE: data in normal state for SPCA with KDE threshold . . . 98
4.36 SPE: data in normal state for SSAE with KDE threshold . . . 99
4.37 SPE: data in normal state for SPCA with AUCL threshold using K-means clustering . . . 99
4.38 SPE: data in normal state for SSAE with AUCL threshold using K-means clustering . . . 100
4.39 SPE: data in normal state for SPCA with AUCL threshold using FCM . . . 100
4.40 SPE: data in normal state for SSAE with AUCL threshold using FCM . . . 101
4.41 SPE: data in faulty state for SPCA with δ_α^2 threshold . . . 101
4.42 SPE: data in faulty state for SSAE with δ_α^2 threshold . . . 102
4.43 SPE: data in faulty state for SPCA with KDE threshold . . . 102
4.44 SPE: data in faulty state for SSAE with KDE threshold . . . 103
4.45 SPE: data in normal state for SPCA with AUCL threshold using K-means clustering . . . 103
4.46 SPE: data in normal state for SSAE with AUCL threshold using K-means clustering . . . 104
4.47 SPE: data in normal state for SPCA with AUCL threshold using FCM . . . 104
4.48 SPE: data in normal state for SSAE with AUCL threshold using FCM . . . 105
4.49 Fault identification using normalized contribution plots for SPAC . . . 105
4.50 Fault identification using normalized contribution plots for SSAE (Fault in the 8 sensor) . . . 106

General Introduction

The diagnosis and operation of dynamic systems are crucial areas to ensure their proper functioning and optimal performance. An innovative approach that has recently emerged in this field is the use of Stacked Sparse Autoencoders (SSAE). Before exploring SSAE, it is essential to understand two preliminary techniques: Principal Component Analysis (PCA) and Sparse PCA. Then, we will delve into SSAE and discuss its specific application to a water treatment system. Finally, we will detail the methods for fault detection and localization.

Principal Component Analysis (PCA) is a commonly used method for dimensionality reduction and extracting the most significant information from data. It transforms a set of correlated variables into a new set of uncorrelated variables called principal components. However, PCA has a limitation in capturing complex nonlinear relationships present in dynamic systems.

To overcome this limitation, Sparse PCA has been developed. This variant of PCA assumes that linear relationships are often sparse in real-world data and that only a few linear combinations are needed to represent the data meaningfully. Thus, Sparse PCA allows capturing complex and nonlinear relationships, improving data representation, and facilitating fault detection.

Stacked Sparse Autoencoders (SSAE) are deep artificial neural networks capable of learning hierarchical and nonlinear representations of data. Unlike PCA and Sparse PCA, SSAE is specifically designed to capture complex and nonlinear relationships present in dynamic systems. They consist of multiple layers of autoencoders, which are models capable of reconstructing data using an intermediate representation (encoding) of features. SSAE can extract increasingly abstract features at each layer, facilitating the detection of significant patterns and motifs in dynamic systems.

In the context of a water treatment system, applying the method based on SSAE is highly relevant. Water treatment systems are essential for providing safe and quality drinking water. However, these systems are prone to various faults such as leaks, equipment malfunctions, variations in water quality, etc. The SSAE-based method can be applied to these systems to extract relevant features from data collected using appropriate sensors. These features learned by SSAE are then used to effectively detect and localize faults in the water treatment system.

Regarding fault detection and localization methods, the use of SSAE allows leveraging the hierarchical and nonlinear representations learned by the network. These methods include analyzing neuron activations in the hidden layers of SSAE, comparing extracted features with reference models, using clustering algorithms to group similar data, and more. These approaches offer precise fault detection and efficient localization in the water treatment system.

In conclusion, the use of Stacked Sparse Autoencoders (SSAE) represents an innovative and powerful approach for diagnosing and operating dynamic systems. This method offers a promising solution to enhance the management of water treatment systems. By leveraging hierarchical and nonlinear data representations, SSAE enables accurate fault detection and efficient localization, contributing to the maintenance and optimization of dynamic system performance.

This work is structured as follow:

- **Chapter 1**: discusses the state-of-the-art in diagnostic techniques for dynamic systems, focusing on existing methods and the challenges encountered.
- **Chapter 2**: provides a detailed presentation of Principal Component Analysis (PCA), Sparse PCA, and modeling and diagnostics based on multivariate statistical data analysis.
- **Chapter 3**: examines the contribution of statistical control methods for nonlinear processes, highlighting the advantages and limitations of these approaches.
- **Chapter 4**: presents a concrete application of the SSAE-based method on a potable water treatment system, detailing the methods used for fault detection and localization.

Chapter 1

STATE OF ART ON DIAGNOSIS OF INDUSTRIAL PROCESSES

1.1 Introduction

Diagnosis is an essential practice in many fields such as medicine, psychology, engineering, and computer science, among others. The state of the art in diagnosis varies depending on the field, but some common points can be identified.

In all fields, diagnosis involves collecting and analyzing data to identify the causes of a problem. The methods used to collect and analyze this data can vary depending on the specific problem and field.
For example, in medicine, diagnosis can be based on blood tests, X-rays, physical evaluations, and more. In psychology, diagnosis can be based on psychometric tests, clinical interviews, behavioral observations, and others. In engineering, diagnosis can be based on sensor data analysis, simulation models, prototype testing, and more.

In computer science, diagnosis can be used to identify problems in computer systems, networks, software, and more. Diagnostic methods in computer science may include file log analysis, load testing, performance analysis, and others.

In all fields, diagnosis is often an iterative process that may require multiple steps to identify the underlying cause of a problem. Problem-solving techniques such as brainstorming, cause-and-effect analysis, and the scientific method may be used to help identify possible causes of a problem.

Finally, diagnosis can be aided by computer tools such as modeling software, machine learning algorithms, and databases. These tools can help collect, store, and analyze data to facilitate diagnosis and problem resolution.

In industry, diagnosis is often performed to identify the causes of failures or breakdowns in equipment and machinery. The diagnostic process may involve data analysis, modeling, and simulation techniques to identify the root causes of problems.

Recent advances in the Internet of Things (IoT) and real-time data analytics have led to the development of predictive diagnostic systems that can predict failures before they occur. These systems use sensors to collect real-time data on machines and transmit it to data analytics algorithms to detect anomalies and predict failures.

In addition, artificial intelligence and machine learning have also enabled the development of automated diagnostic systems that can identify problems and failures more quickly and accurately than traditional manual diagnostic methods. These systems use machine learning algorithms to analyze diagnostic data and provide recommendations for problem resolution.

In summary, industrial diagnosis relies on data analysis, modeling, and simulation techniques, as well as technological advances such as IoT, real-time data analytics, artificial intelligence, and machine learning. These advances have enabled the development of predictive and automated diagnostic systems that can identify problems more quickly and accurately than traditional manual diagnostic methods.

1.2 Purpose of monitoring

The objective of industrial supervision is to centrally monitor and control production processes and equipment [**?**]. This allows real-time data to be collected on equipment and process performance, enabling quick problem detection and preventive measures to avoid breakdowns and unexpected downtime.

Industrial supervision can be used to monitor various parameters such as temperature, pressure, flow rate, level, quality, and energy consumption [**?**] [**?**] [**?**] [**?**]. The collected data can be analyzed to detect anomalies, trends, and abnormal behavior, which can help identify potential problems before they turn into breakdowns or failures.

Industrial supervision also provides real-time information on production, quality, and equipment performance, which can help optimize production processes, improve product quality, reduce production costs, and maximize equipment availability and

reliability. In summary, the main objective of industrial supervision is to ensure that production processes and equipment operate reliably, efficiently, and safely, while minimizing maintenance costs and unexpected downtime.

1.3 Definitions and Concepts

Diagnosis in industry can be defined as the process of collecting and analyzing data to identify underlying causes of failures or breakdowns in equipment and machines. The results of the diagnosis are used to determine the necessary repair and maintenance measures to restore the normal operation of the equipment or machines[**?**] [**?**] [**?**].

Here are some key concepts associated with diagnostic in industry [**?**] [**?**] [**?**] [**?**] [**?**] [**?**]:

Data analysis: a technique used to identify patterns and trends in data collected from equipment and machines. Data analysis helps to detect anomalies and abnormal trends, which can help identify potential problems before they turn into failures or breakdowns.

Modeling and simulation: tools used to simulate the behavior of equipment and machines under normal and abnormal conditions. Modeling and simulation enable engineers to identify potential causes of failures and breakdowns, and propose solutions to prevent or resolve them.

Predictive diagnosis: : a diagnosis method that uses data analysis algorithms [**?**] to detect anomalies and predict failures before they occur. Predictive diagnostics rely on real-time data collected from sensors installed on equipment and machines.

Automated diagnosis: a diagnosis method that uses machine learning algorithms [**?**] to identify problems and breakdowns more quickly and accurately than traditional manual diagnostic methods. Automated diagnostic systems can provide recommendations for resolving detected problems.

Preventive maintenance: a maintenance strategy that involves performing regular maintenance operations on equipment and machines to prevent failures and breakdowns [**?**]. Preventive maintenance is based on data analysis and modeling to determine optimal maintenance intervals.

By using these diagnostic concepts and techniques, companies can improve the availability and reliability of their equipment and machines, reduce maintenance costs, and avoid unexpected downtime.

1.4 Monitoring and diagnosis

Monitoring and diagnosis play a critical role in the industrial sector [**?**], ensuring the efficient operation, safety, and reliability of industrial processes and equipment [**?**]. Here are some key aspects of monitoring and diagnostics in the industry:

Real-time monitoring: Industrial systems rely on real-time monitoring to continuously collect data from various sensors and devices. This data provides insights into the performance, condition, and operation of equipment, allowing operators to detect deviations and anomalies promptly.

Condition monitoring: Condition monitoring involves monitoring specific parameters, such as temperature, pressure, vibration, or fluid levels, to assess the health and performance of equipment. By continuously monitoring these parameters, abnormalities or early signs of deterioration can be detected, enabling proactive maintenance actions.

Remote monitoring and control: Many industrial systems utilize remote monitoring and control capabilities, allowing operators to monitor and control equipment from a central control room or remote location. This enables real-time observation of critical parameters, facilitates rapid response to issues, and allows for efficient management of multiple assets.

Predictive maintenance: Monitoring and diagnostics support predictive maintenance strategies, where the condition of equipment is continuously assessed to predict potential failures or performance degradation. By analyzing data patterns and trends, maintenance activities can be scheduled proactively, minimizing unplanned downtime and optimizing maintenance costs.

Fault detection and diagnosis: Monitoring systems employ algorithms and analytics to detect faults and anomalies in industrial equipment. By analyzing sensor data, deviations from normal behavior can be identified, triggering alarms or notifications. Diagnostic techniques are then applied to pinpoint the root causes of faults, aiding in effective troubleshooting and timely repairs.

Data-driven decision-making: Monitoring and diagnostics generate a wealth of data that can be analyzed to derive insights and support decision-making processes. By leveraging advanced analytics and machine learning algorithms, operators and maintenance personnel can make informed decisions regarding equipment performance, maintenance strategies, and process optimization.

Integration with automation systems: Monitoring and diagnostic systems are often integrated with industrial automation systems, such as SCADA (Supervisory Control

and Data Acquisition) or DCS (Distributed Control System). This integration allows for seamless data exchange and coordinated actions between the monitoring and control systems, enhancing operational efficiency and response capabilities.

Overall, monitoring and diagnostics in the industry enable proactive maintenance, enhance operational efficiency, improve safety, and reduce downtime. By leveraging real-time data analysis and predictive capabilities, industrial processes can be optimized, ensuring the smooth operation of equipment and systems. The figure 1.1 shows the general structure of a monitoring system.

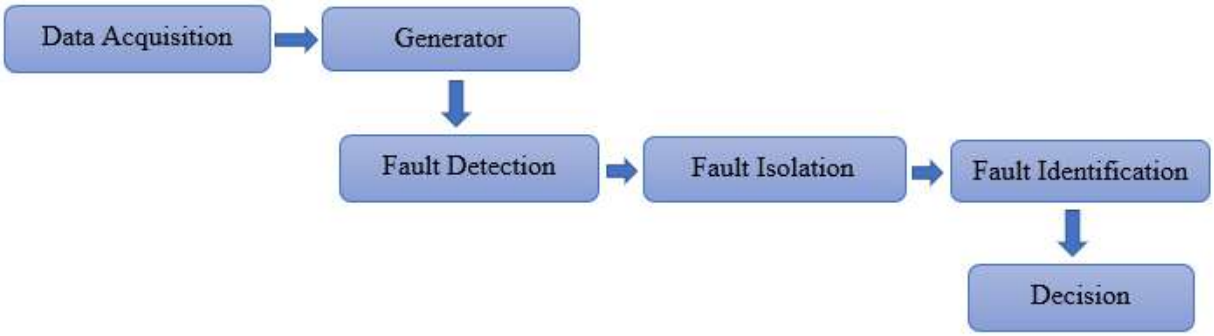

Figure 1.1: The general structure of a monitoring system

1.5 Fault detection and isolation procedure

Fault detection and isolation procedure is a systematic method for identifying problems and malfunctions in industrial equipment or systems [**?**], and for isolating the underlying causes of the problem. This procedure can be divided into several key steps, including:

Data collection: collecting data on the symptoms of the problem and on the performance of the system in general.

Data analysis: analyzing the data to detect anomalies and abnormal behaviors that may indicate a failure or problem.

Fault isolation: once the problem is detected, it is important to isolate it by identifying the underlying causes. This may involve examining individual components of the

system, checking connections, or analyzing performance data to identify patterns of abnormal behavior.

Fault correction: after isolating the fault, appropriate repair and maintenance measures can be taken to correct the problem. This may include replacing faulty components, repairing connections, or reprogramming the programmable logic controller (PLC) in charge of the system.

Verification: verifying that the correction has solved the problem and that the system is operating optimally.

In summary, the procedure for fault detection and isolation (Figure 1.2) is an important process for maintaining the performance [**?**] and reliability of industrial equipment and systems, and for minimizing unexpected downtime and maintenance costs.

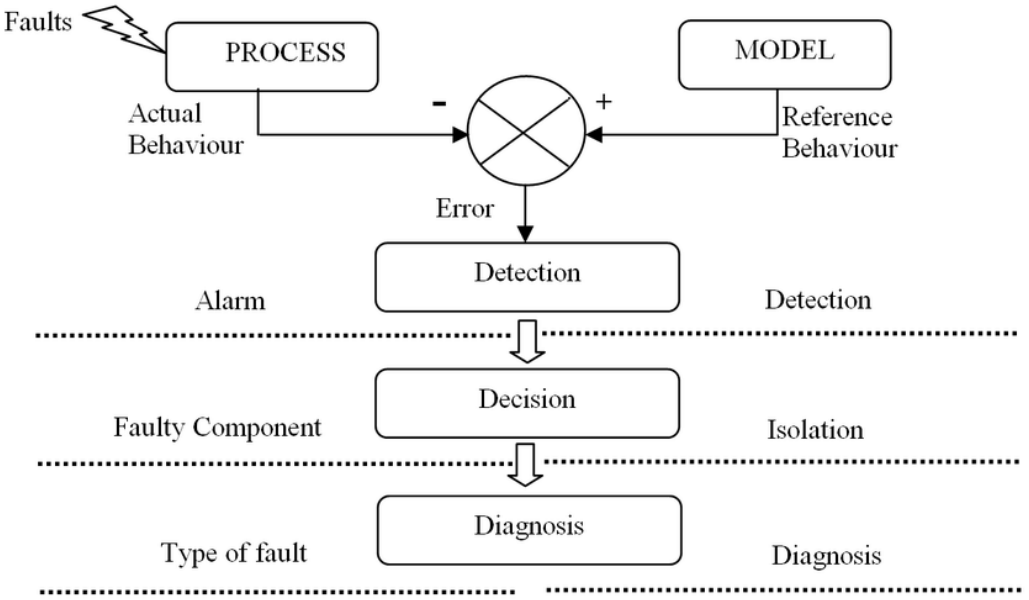

Figure 1.2: Concept of model based fault detection and isolation

1.5.1 Different types of fault structures

There are different types of fault structures that can occur in industrial equipment or systems [**?**] [**?**]. Here are some examples:

Partial failure: a partial failure occurs when a component is not functioning at its optimal capacity but continues to work at a lower level.
Total failure: a total failure occurs when a component completely stops working.
Intermittent failure: an intermittent failure occurs when a component does not work consistently and may cause intermittent issues.
System failure: a system failure occurs when a set of components is not working properly and affects the overall functioning of the system.
Design failure: a design failure occurs when the component or system was designed improperly or inadequately, leading to performance issues.
Human error: human error occurs when a person makes a mistake in the design, commissioning, or maintenance of the system, which can lead to a failure or problem.
Component degradation: component degradation occurs when a component deteriorates over time and loses performance, which can eventually lead to failure.
Wear: wear occurs when component experiences wear and tear due to normal use of the system, which can lead to long-term failure.
Understanding these different fault structures can help identify problems more quickly and take preventive measures to avoid unexpected downtime and failures.

1.5.2 Classification of diagnosic methods

The classification of diagnostic methods [**?**] [**?**] [**?**] (Figure 1.3) can be grouped into several categories, including:

Model-based methods: Model-based methods use mathematical models of the system to diagnose problems. This can include techniques such as system identification, state estimation, and parameter estimation.

Signal-based methods: Signal-based methods analyze signals from sensors or other measurements to diagnose problems. This can include techniques such as statistical process control, time-series analysis, and signal processing.

Artificial intelligence (AI)-based methods: AI-based methods use machine learning algorithms [**?**] to learn patterns in data and diagnose problems. This can include techniques such as neural networks, decision trees, and support vector machines.

Hybrid methods: Hybrid methods combine two or more of the above methods to diagnose problems. For example, a hybrid method might use expert knowledge to select the appropriate model-based or AI-based method for a given problem.

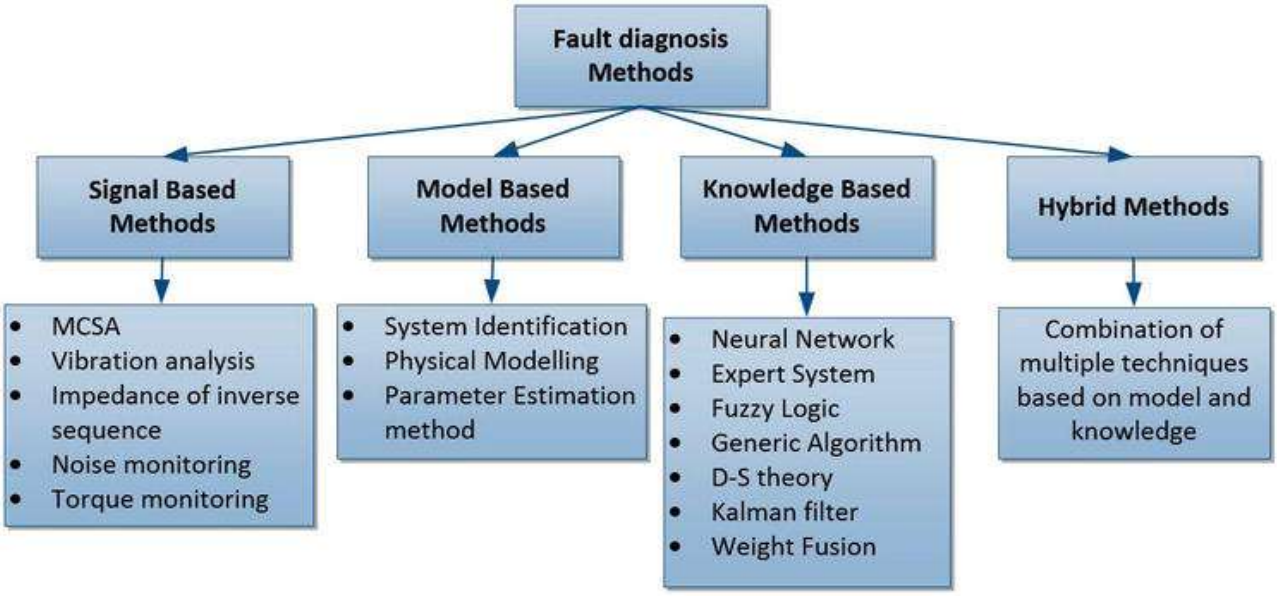

Figure 1.3: Classification of different fault diagnosis methods

The choice of diagnostic method will depend on the nature of the system, the available data and sensors, and the expertise of the diagnostician.

1.5.3 Fault diagnosis methods

There are various fault diagnosis methods [**?**] employed in different industries as shown in Figure to 1.4 identify and classify faults or anomalies in equipment and systems. Here are some commonly used fault diagnosis methods:

Model-based diagnostic methods

Model-based diagnostic methods [**?**] involve the use of a model or a set of models to compare the expected behavior of a system or component with its actual behavior in order to identify faults or anomalies. These methods rely on mathematical or physical models that describe the behavior of the system under normal conditions,

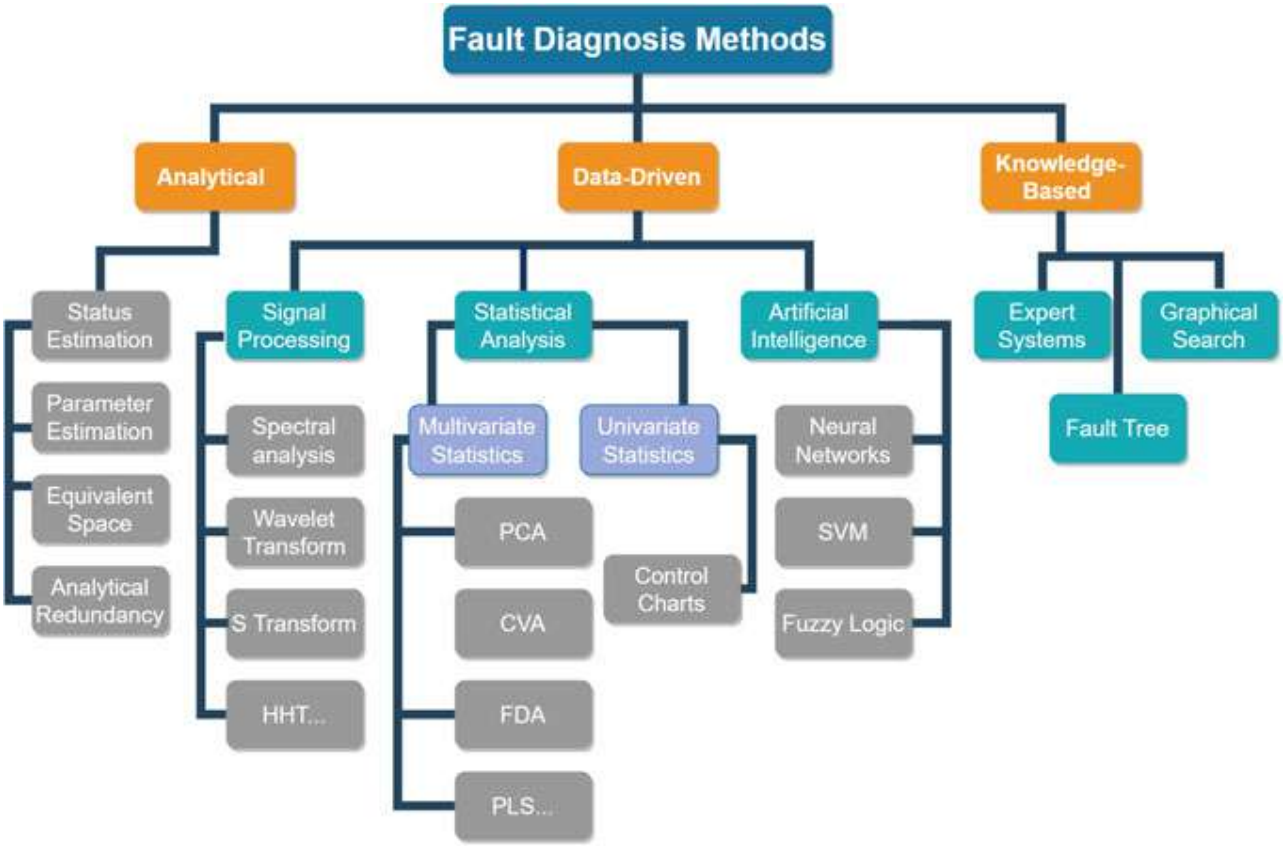

Figure 1.4: Fault diagnosis methods

which are then used to detect deviations or discrepancies from the expected behavior. Model-based methods can be further classified into several categories, including:

Analytical redundancy: This method [**?**] [**?**] involves comparing the measurements of a system with its mathematical model to detect any discrepancies or deviations. This approach can be used to detect and isolate faults, as well as to estimate the magnitude of the fault.

Knowledge-based reasoning: This method involves using expert knowledge or rules to diagnose faults. These rules are typically based on the experience of experts in the field and can be used to identify common patterns of behavior that indicate a fault.

Artificial intelligence (AI): AI-based diagnostic methods involve the use of machine learning algorithms or neural networks to learn patterns of behavior in the system and to identify anomalies or faults. These methods can be used to identify complex patterns of behavior that may be difficult to detect using traditional methods.

Hybrid methods: Hybrid methods combine different diagnostic techniques, such as analytical redundancy and knowledge-based reasoning, to improve the accuracy and reliability of the diagnostic process.

Overall, model-based diagnostic methods are widely used in industrial and engineering applications due to their ability to provide accurate and reliable diagnoses of faults and anomalies in complex systems.

Methods without models

Methods without models in diagnostics are approaches that do not use pre-established models to detect faults in a system. Instead, these methods rely on real-time data collected from the system to detect anomalies and abnormal behavior. Here are some examples of methods without models in diagnostics:

Analysis of variance (ANOVA): ANOVA is a statistical method that compares the means of multiple groups of data. This method can be used to detect differences between normal and abnormal data.

Principal component analysis (PCA): PCA [**?**] is a statistical method that reduces the dimensionality of data by finding the principal directions of the data variance. This method can be used to detect anomalies in the data.

Neural networks: Neural networks are machine learning models that can be used to detect anomalies in data. Neural networks can be trained to recognize normal patterns in the data and alert when abnormal patterns are detected.

Threshold-based methods: Threshold-based methods [**?**] are approaches that compare real-time data to pre-defined thresholds to detect anomalies. If the data exceeds the threshold, a warning is generated.

These methods without models in diagnostics may be less accurate than methods with models, but they can be useful in detecting anomalies in systems where models are not available or difficult to establish.

1.6 The performance of a diagnosis system in industry

In industry, the performance of a diagnostic system is of paramount importance as it directly impacts the reliability, safety, and efficiency of industrial equipment and systems [**?**]. Here are some key considerations for evaluating the performance of a diagnostic system in an industrial setting:

Accuracy and reliability: The diagnostic system should exhibit a high level of accuracy in detecting faults and anomalies to minimize false positives and false negatives. Reliability is crucial to ensure that the system consistently provides accurate diagnoses over time.

Speed and real-time analysis: Industrial processes often operate in real-time, so the diagnostic system should be capable of analyzing data and providing timely results. Swift detection and diagnosis of faults can prevent downtime, minimize disruptions, and reduce maintenance costs.

Scalability and adaptability: Industrial environments can involve complex and diverse systems. A diagnostic system should be scalable to handle large volumes of data and adaptable to various types of equipment and processes. It should be able to accommodate different industrial domains and easily integrate with existing infrastructure.

Fault classification and severity assessment: The diagnostic system should not only detect faults but also accurately classify them based on their type and severity. This information helps prioritize maintenance actions and allocate resources effectively.

Proactive maintenance: An effective diagnostic system can enable predictive and proactive maintenance strategies. By identifying potential issues before they cause major failures, it allows for scheduled maintenance, reducing unplanned downtime and optimizing equipment performance.

Data availability and training: The diagnostic system's performance can be enhanced by having access to high-quality training data that represents the range of faults and

anomalies in the industrial system. Sufficient and diverse data enables the system to learn and improve its diagnostic capabilities.

Integration with monitoring systems: Integration with existing monitoring systems, such as sensors and data acquisition systems, can provide continuous data streams for analysis and enable real-time monitoring of equipment health.

Ultimately, the performance of a diagnostic system in industry is measured by its ability to accurately and efficiently identify faults and anomalies, facilitate proactive maintenance, and contribute to the overall reliability and productivity of industrial processes.

1.7 Conclusion

In conclusion, Chapter 1 has provided an overview of the current state of diagnostic techniques in dynamic systems. We have explored various existing methods and highlighted the challenges faced in achieving accurate and efficient diagnostics. It is evident that diagnostic procedures play a crucial role in ensuring the proper functioning and optimal performance of dynamic systems. However, the complexity and nonlinearity of these systems pose significant challenges for traditional diagnostic approaches.

To address these challenges, Chapter 2 will delve into the detailed exploration of Principal Component Analysis (PCA), Sparse PCA, and modeling and diagnostics based on multivariate statistical data analysis. These advanced techniques offer promising solutions for capturing the complex relationships and patterns within dynamic systems, allowing for more accurate and effective diagnostics. By leveraging these methods, we aim to enhance our understanding of the underlying dynamics and improve the diagnosis of faults in dynamic systems.

Chapter 2

Nonlinear Multivariate Statistical Process Monitoring

2.1 Introduction

Chapter 2 focuses on exploring two fundamental techniques in the field of diagnostic and control of dynamic systems: Principal Component Analysis (PCA) and Sparse PCA. These techniques play a crucial role in dimensionality reduction, feature extraction, and modeling of complex systems.

Firstly, we delve into Principal Component Analysis (PCA), which is a widely used method for dimensionality reduction and feature extraction. PCA allows us to transform a set of correlated variables into a new set of uncorrelated variables, known as principal components. However, the traditional PCA approach has limitations in capturing non-linear and complex relationships present in dynamic systems.

To address this limitation, Sparse PCA has been developed as a variant of PCA. Sparse PCA assumes that linear relationships are often sparse in real-world data, meaning that only a few linear combinations are necessary to represent the data effectively. By leveraging sparsity, Sparse PCA enables the capture of complex and non-linear relationships, enhancing data representation and facilitating fault detection.

In this chapter, we will delve into the theoretical foundations of PCA and Sparse PCA, discussing their mathematical formulations, underlying assumptions, and implementation techniques. We will explore how these techniques can be applied to dynamic systems to extract meaningful features, identify patterns, and aid in the diagnostic process.

Furthermore, we will investigate the integration of PCA and Sparse PCA into the modeling and diagnostic process based on multivariate statistical data analysis. This approach combines the power of dimensionality reduction and feature extraction with statistical analysis, enabling a comprehensive understanding of system behavior and fault detection.

The chapter aims to provide a solid theoretical understanding of PCA and Sparse PCA, along with practical insights into their application in diagnosing and monitoring dynamic systems. By leveraging these techniques, engineers and researchers can effectively analyze large-scale datasets, identify critical features, and develop accurate models for fault detection and control.

With this foundation in place, we can now proceed to Chapter 3, where we will explore the contribution of statistical control methods for nonlinear processes, building upon the concepts and techniques discussed in Chapter 2.

2.2 Principal component analysis

Principal component analysis (PCA)[**?**] [**?**] is a multivariate statistical analysis technique that aims to reduce the dimensionality of a dataset by creating new variables called principal components. This technique is used to identify the variables that contribute the most to the variation in a dataset and to determine how these variables are related to each other.

PCA is often used to explore and visualize complex data by reducing the number of variables to consider. This technique is also used to facilitate statistical modeling by reducing the dimensionality of input data and eliminating redundant variables.

The process of PCA involves the linear transformation of the original data into a new space of orthogonal variables. Each new variable represents a linear combination of the original variables, weighted by coefficients called "factor loadings". The first principal components represent the highest variance in the original data, while the subsequent components represent a decreasing proportion of the variance.

PCA can be used in a variety of fields, including economics, finance, biology, psychology, and engineering. This technique is particularly useful in fields where data is complex and where it is difficult to effectively visualize or model the relationship between variables.

2.3 Identification of PCA model

Model identification in PCA [**?**] is a statistical analysis process that involves determining the optimal number of principal components to use in representing a dataset. It is important to choose the right number of principal components as this can affect the accuracy and interpretability of PCA results.

There are several methods for identifying the optimal number of principal components. The most common method is the "elbow" method, which involves plotting the percentage of variance explained by each principal component against the number of principal components and identifying the point where the curve starts to flatten or form an elbow. This point is considered as the optimal number of principal components to use.

Another common method is the "scree plot" method, which also involves plotting the percentage of variance explained by each principal component against the number of principal components, but identifies the point where the curve starts to flatten or "drop" sharply. This point is also considered as the optimal number of principal components to use. Generally, it is recommended to use enough principal components to explain at least 70% of the total variance in the data. However, the optimal number of principal components may vary depending on the context and purpose of the analysis. In summary, model identification in PCA is an important step in data analysis using PCA, which involves determining the optimal number of principal components to use in representing a dataset accurately and interpretable.

Mathematically, model identification in PCA involves calculating the eigenvalues and eigenvectors of the covariance or correlation matrix of the original variables. Eigenvectors are also called factor axes or principal components.

For each eigenvector, the corresponding eigenvalue indicates the proportion of the total variance of the data that is explained by that eigenvector. The eigenvectors are ranked in descending order of eigenvalues, so that the first principal component explains the largest proportion of the total variance, followed by the second principal component, and so on. Thus, to identify the optimal number of principal components, one can examine the eigenvalues in descending order and identify the point where the eigenvalue curve starts to flatten or form an elbow, or use the "scree plot" method to identify the point where the curve starts to flatten or "drop" sharply. Finally, to determine the coefficients of each variable in each principal component, one can calculate the factor loadings by multiplying the matrix of eigenvectors by the matrix of

centered and standardized (or normalized) data, and then normalizing the results to obtain loadings with unit variance.

In summary, mathematically identifying the PCA model involves calculating the eigenvectors and eigenvalues of the covariance or correlation matrix of the original variables, using this information to determine the optimal number of principal components to use, and calculating the coefficients of each variable in each principal component.

The purpose of Principal Component Analysis (PCA) is to identify the existing linear relationships among the different variables of the system using input and output data of the system.

Considering the vector $x(K) = [x_1(K)x_2(K)....x_n(K)]^T$ with n variables associated with an observation K, the matrix X is then composed of N observations of the vector $x(K)$.
The data matrix X:

$$T = X^T P \tag{2.1}$$

$$X = PT^T \tag{2.2}$$

With $T \in R^{N\times n}$ and $P \in R^{n\times n}$ are the matrices of the principal components and the corresponding eigenvectors are obtained from the spectral decomposition of the covariance matrix Σ of X:

$$\Sigma = \frac{1}{N-1} XX^T \tag{2.3}$$

$$\Sigma = P\Lambda P^T \qquad with \qquad PP^T = P^T P = I_n \tag{2.4}$$

With Λ the diagonal matrix of eigenvalues, where the diagonal terms are arranged in descending order of magnitudes, is represented as: $\lambda_1 \geq \lambda_2 ... \geq \lambda_n$.
Let us partition the matrices of eigenvalues, eigenvectors, and principal components:

$$\Lambda = \begin{bmatrix} \underset{\ell\times\ell}{\hat{\Lambda}} & 0 \\ 0 & \underset{(n-\ell)\times(n-\ell)}{\tilde{\Lambda}} \end{bmatrix}$$

$$P = [\underset{n\times\ell}{\hat{P}} \quad \underset{n\times(n-\ell)}{\tilde{P}}] \tag{2.5}$$

$$T = [\underset{N\times\ell}{\hat{P}} \quad \underset{N\times(n-\ell)}{\tilde{P}}] \tag{2.6}$$

Where the index ℓ corresponds to the number of "retained" principal components, it is associated with the largest eigenvalues.

$$X = \hat{P}\hat{T}^T + \tilde{P}\tilde{T}^T \tag{2.7}$$

By setting :

$$\hat{X} = \hat{P}\hat{T}^T = \sum_{i=1}^{\ell} p_i t_i^T \tag{2.8}$$

$$E = \tilde{P}\tilde{T}^T = \sum_{i=\ell+1}^{n} p_i t_i^T \tag{2.9}$$

By performing the following decomposition of the data matrix, we obtain: :

$$X = \hat{X} + E \tag{2.10}$$

Where the matrices $\hat{X}$ and E represent, respectively, the modeled variations from ℓ components ($\ell < n$) and the residual (unmodeled) variations of X.
It can also be shown that the matrices $\hat{X}$ and E are given by:

$$\hat{X} = \hat{C}X \tag{2.11}$$

And

$$E = \tilde{C}X \tag{2.12}$$

Where $\hat{C} = \hat{P}\hat{P}^T$ and $\tilde{C} = I_n - \hat{C}$ constitute the PCA model of the system.
The ℓ first eigenvectors $\hat{P} \in R^{n\times\ell}$ constitute the principal space of the data, while the last $(n-l)$ eigenvectors $\tilde{P} \in R^{n\times(n-\ell)}$ constitute the residual space.

2.4 Choice of the dimension of the reduced space and the number of principal components (CPs)

The choice of the dimension of the reduced space and the number of principal components [**?**] depends on several factors, including the objective of the analysis and the number of initial variables. A common approach is to retain a number of principal components that explain a sufficient proportion of the total variance of the data, typically 70% or more. This can be determined by examining the eigenvalues, which represent the amount of variance explained by each principal component.

Another approach is to consider the Kaiser-Guttman rule, which recommends retaining only the principal components with an eigenvalue greater than 1. This rule is based on the idea that each principal component should explain more variance than a standardized initial variable. A third approach is the Anderson's break method, which involves identifying the point where the eigenvalue curve flattens or forms an elbow by comparing the observed eigenvalues to the theoretical eigenvalues calculated from a random distribution.

It is important to note that the final choice of the dimension of the reduced space and the number of principal components can be influenced by additional considerations such as simplicity of interpretation, stability of results, or performance of predictive models based on the principal components.

2.4.1 The cumulative percentage of variance (PCV)

The cumulative percentage of variance (PCV) [**?**] is a measure that indicates the proportion of the total variance of the data that is explained by a given number of principal components.

The PCV can be calculated [**?**] by cumulating the proportions of variance explained by each principal component, starting with the principal component that explains the largest proportion of variance. For example, if the first two principal components explain 30% and 20% of the total variance, respectively, the PCV for these two principal components would be 50%.

The PCV is useful for determining the number of principal components to retain in principal component analysis, as it allows for choosing a sufficient number of

components to explain a satisfactory proportion of the total variance. A high PCV indicates that the retained principal components are able to explain a large portion of the variability in the data, while a low PCV may indicate that the retained principal components are not sufficiently representative of the initial data.

The cumulative percentage of variance (PCV) for l principal components can be calculated from the eigenvalues Λ_1, Λ_2, ..., Λ_l of the first l principal components as follows:

$$PCV_l = 100 \left(\frac{\sum_{j=1}^{l} (\Lambda_j)}{\sum_{j=1}^{m} (\Lambda_j)} \right) \% \tag{2.13}$$

where m is the total number of initial variables. The PCV_l represents the cumulative proportion of the total variance explained by the first l principal components.

2.4.2 The variance of reconstruction error (VRE)

The variance of reconstruction error (VRE) [**?**] is a measure of the quality of reconstructing a dataset from its principal components. Using principal component analysis (PCA), data can be projected onto a reduced-dimensional space using a limited number of principal components, which can significantly reduce the complexity of the data while retaining the most important information.

The VRE [**?**] represents the amount of residual variance that is not explained by the principal components retained in the data reconstruction. A low VRE indicates high-quality data reconstruction, meaning that the retained principal components explain a large proportion of the variance in the data. Conversely, a high VRE indicates lower quality reconstruction, meaning that the retained principal components fail to explain a large proportion of the variance in the data. The VRE can be calculated using the following formula:

$$VRE = \frac{1}{n} \sum (x - \hat{x})^2 \tag{2.14}$$

Where n is the total number of observations, x is the original data vector, and $\hat{x}$ is the reconstructed data vector from the retained principal components.

2.4.3 Detection and localization of faults by PCA

Principal component analysis (PCA) is a commonly used method for detecting and localizing defects [**?**] in data. In the case of multivariate data, defects can be caused by outliers, measurement errors, or measurement noise. Defects can also be caused by errors in input data, defective manufacturing processes, or faulty equipment.
Defect detection by PCA [**?**] involves creating a reference model from the original data. This model is then used to evaluate new data as it is collected. If a new observation is far from the reference model, it may indicate the presence of a defect. Defects can be localized by identifying the variables that contribute the most to the deviation of the new observation from the reference model.

PCA can be used for defect detection and localization in various applications, such as monitoring industrial processes, monitoring product quality, detecting financial fraud, and monitoring equipment health.

Novelty detection

Novelty detection [**?**] focuses on identifying previously unseen or unknown patterns, anomalies, or deviations in data. It aims to detect instances that significantly differ from the normal behavior of a system or process. By using machine learning or statistical techniques, novelty detection algorithms can learn from historical data to distinguish between normal and abnormal instances, without prior knowledge of the abnormal instances during training.

Diagnosis [**?**], on the other hand, involves the identification and classification of faults or malfunctions in industrial systems. It aims to pinpoint the specific cause or root of the problem based on observed symptoms or indicators. Diagnosis techniques often rely on analyzing sensor data, system parameters, or expert knowledge to determine the underlying issue and provide actionable insights for maintenance or corrective actions.

In practice, novelty detection [**?**] can be a valuable tool within the diagnostic process. It helps to flag unusual or unexpected events that may indicate a fault or anomaly in the system. By incorporating novelty detection into the diagnostic framework, it becomes possible to proactively detect and diagnose previously unseen or unknown faults, leading to improved system reliability, efficiency, and maintenance planning.

Several approaches exist for detecting errors or process aggravations. Limited research has been conducted on fault detection processing to diagnose the primary causes of

malfunctions. Among these approaches, residue analysis is commonly used, which involves performing a similarity test between measured and estimated data.

The Statistical Process Control Exponentially (SPE): The Statistical Process Control - Exponentially Weighted Moving Average (SPE)[**?**] [**?**] [**?**] detection index is a process monitoring method used to detect quality or performance deviations in a manufacturing or production process. The SPE detection index uses a mathematical formula to calculate a real-time monitoring value for each unit produced. This monitoring value is then compared to a control limit to determine whether the process is in control or out of control.

If the monitoring value exceeds the control limit, it indicates that there is a deviation in the process that needs to be examined and corrected to avoid failures or quality issues. The SPE detection index is widely used in the manufacturing, electronics, aerospace, and automotive industries to monitor production processes and improve product quality. The mathematical formula for calculating the SPE detection index is as follows: At k moment, we have :

$$SPE(k) = \sum_{i=1}^{m} (e_i(k))^2 \tag{2.15}$$

Where $e_i(k)$ is the i^{th} residual gived by:

$$e_i(k) = x_i(k) - \hat{x}_i(k) \tag{2.16}$$

Where m is the number of input variables, $x_i(k)$ is the i^{th} element of the measurement vector $x(k)$, $\hat{x}$ is the estimation of x using the PCA model, which is defined as:

$$\hat{x} = \hat{C}x \tag{2.17}$$

The process is considered to be in abnormal operation (presence of a fault) at time instant k if :

$$SPE(k) > \delta_\alpha^2 \tag{2.18}$$

Where δ_α^2 is the detection threshold of $SPE(k)$. Assuming that $\theta_i = \sum_{j=\ell+1}^{m} \lambda_j^i$ where

λ_j^i is the j^{th} eigenvalue of the matrix Σ to the power of i :

$$\delta_\alpha^2 = g\chi_{h,\alpha}^2 \tag{2.19}$$

Where $g = \theta_2/\theta_1$,$h = integer(\theta_1^2/\theta_2)$,$integer(o)$ is the integer value of o and $\chi_{h,\alpha}^2$ is the distribution of χ^2 with h degrees of freedom. The SPE monitoring value is then compared to a control limit to determine whether the process is in control or out of control.

Hotelling's T^2 Statistic: Hotelling's T^2 statistic [**?**] [**?**] is a multivariate statistical measure used to detect deviations or anomalies in a dataset. It assesses the distance of a data point from the center of a distribution in a multivariate space. It is based on the principle of comparing the observed data point to the expected distribution and quantifying the degree of deviation. The larger the value of Hotelling's T^2 statistic, the further the data point is from the center of the distribution, indicating a potential anomaly. At time k, we have:

$$T^2(k) = \sum_{i=1}^{\ell} \frac{t_i^2(k)}{\lambda_i} \tag{2.20}$$

where t_i is the i^{th} principal component and λ_i is the i^{th} eigenvalue of the covariance matrix Σ.
The process is considered to be in abnormal operation (presence of a fault) at time k if:

$$T^2(k) > T_\alpha^2 \tag{2.21}$$

Where T_α^2 is the detection threshold for $T^2(k)$ given by :

$$T_\alpha^2 = \frac{\ell(N-1)(N+1)}{N(N-\ell)} F_{\ell,N-\ell,\alpha} \tag{2.22}$$

Where $F_{\ell,N-\ell,\alpha}$ is the Fisher distribution with ℓ and $N-\ell$ degrees of freedom.

Hawkins' T_H^2 Statistic or SWE: Hawkins' T_H^2 statistic, also known as the Squared Weighted Euclidean (SWE) statistic, is a multivariate statistical measure used for anomaly detection. It is an extension of Hotelling's T^2 statistic that focuses on the residual space or the subspace orthogonal to the principal components.

The T_H^2 statistic measures the distance of a data point from the subspace spanned by the residual components. It quantifies the deviation of the data point from the expected behavior based on the residual variation. By considering the residual space, it provides additional information about anomalies that may not be captured by the principal components.

A high value of Hawkins' T_H^2 statistic suggests that the data point deviates significantly from the expected behavior in the residual subspace, indicating a potential anomaly or unusual pattern in the data.
At time k, we have:

$$T_H^2(k) = \sum_{i=\ell+1}^{m} \frac{t_i^2(k)}{\lambda_i} \tag{2.23}$$

Where t_i is the i^{th} principal component and λ_i is the i^{th} eigenvalue of the covariance matrix Σ.
The process is considered to be in abnormal operation (presence of a fault) at time k if:

$$T_H^2(k) > T_{H\alpha}^2 \tag{2.24}$$

Where T_α^2 is the detection threshold for $T_H^2(k)$ given by :

$$T_{H\alpha}^2 = \frac{(m-\ell)(N-1)(N+1)}{N(N-m+\ell)} F_{m-\ell,N-m+\ell,\alpha} \tag{2.25}$$

Where $F_{m-\ell,N-m+\ell,\alpha}$ is the Fisher distribution with $m-\ell$ and $N-m+\ell$ degrees of freedom.

Combined Index: The combined index [**?**] is a statistical measure used for anomaly detection that combines multiple indicators or statistics into a single value. It is designed to provide a comprehensive assessment of the abnormality or deviation of a data point from the expected behavior.

The combined index typically integrates different measures, such as the Hotelling's T^2 statistic, the squared prediction error (SPE), or other relevant indicators. These individual measures capture different aspects of the data and may have different sensitivities to anomalies.

By combining multiple indicators, the combined index aims to improve the detection accuracy and robustness by considering various aspects of the data. It allows for a more

comprehensive evaluation of the anomaly likelihood and provides a single threshold or criterion for determining whether a data point is abnormal or not.

The specific formulation of the combined index may vary depending on the application and the chosen indicators. It is often defined as a weighted sum or a function of the individual indicators, where the weights or the function parameters are determined based on the specific requirements and characteristics of the problem at hand.

Yue et Qin [**?**] propose the use of a combined index for fault detection that combines the *SPE* and the T^2 statistics as follows:

$$\varphi(k) = \frac{T^2(k)}{T_\alpha^2} + \frac{SPE(k)}{\delta_\alpha^2} = X^T(k)\Phi X(k) \tag{2.26}$$

avec

$$\Phi = \frac{\hat{P}\hat{\Lambda}^{-1}\hat{P}^T}{T_\alpha^2} + \frac{I_m - \hat{P}\hat{P}^T}{\delta_\alpha{}^2} \tag{2.27}$$

Where T_α^2 is the detection threshold for the T^2 and δ_α^2 is the detection threshold for the *SPE* index. The process is considered to be in abnormal operation (presence of a fault) at time k if:

$$\varphi(k) > g\chi_{h,\alpha}^2 \tag{2.28}$$

With

$$g = \frac{\sum_{j=1}^{m} \lambda_j^2}{\sum_{j=1}^{m} \lambda_j} = \frac{trace(\Sigma\Phi)^2}{trace(\Sigma\Phi)} \tag{2.29}$$

And h is the number of degrees of freedom of the χ^2 distribution:

$$h = \frac{\left(\sum_{j=1}^{m} \lambda_j\right)^2}{\sum_{j=1}^{m} \lambda_j^2} = \frac{[trace(\Sigma\Phi)]^2}{trace(\Sigma\Phi)^2} \tag{2.30}$$

EWMA filter: The Exponentially Weighted Moving Average (EWMA)[**?**] filter is a commonly used data filtering technique in statistics and signal processing for smoothing time series. The EWMA filter calculates an exponentially weighted moving average of the data by assigning a decreasing weight to the historical data. Exponential weights give more importance to the most recent data, making the EWMA filter more sensitive to recent trends than other moving average methods. The mathematical formula for the EWMA filter [**?**] is as follows:

$$\overline{\mathrm{SPE}}(\mathrm{k}) = \|\bar{\mathrm{e}}(\mathrm{k})\|^2 \tag{2.31}$$

With

$$\bar{e}(k) = (I-\beta)\bar{e}(k-1) + \beta e(k) \tag{2.32}$$

Where β is a diagonal matrix whose elements are the forgetting factors for the residuals, I is an identity matrix, and $e(k) = 0$.
With $\bar{e}(k)$ and $\bar{S}PE(k)$ being the filtered vectors of the residuals $e(k)$ and squared prediction error (SPE) respectively. It should be noted that $\bar{S}PE(k)$ helps reduce false alarms but introduces some delay in the detection.
To simplify, we assume in the following that the matrix of forgetting factors is given by:

$$\beta = \gamma I \tag{2.33}$$

In this case, the process is considered to be in abnormal operation (presence of a defect) at time k if:
$\bar{SPE}(k) > \bar{\delta}^2_\alpha$ where $\bar{\delta}^2_\alpha$ is the confidence threshold for this filtered indicator, which is given by the following formula:

$$\bar{\delta}^2 = \frac{\gamma}{2-\gamma}\delta^2_\alpha \tag{2.34}$$

Where γ is a forgetting factor.
The EWMA filter can also be used in change detection to monitor variations in a process and signal significant changes in the data. By using a control limit based on the standard deviation of the filtered data, the EWMA filter can detect significant drifts

in the data and signal changes that require further investigation.

Control limits: Control limits, also known as specification limits or control bounds, are thresholds used in statistical process control (SPC) to determine if a process is operating within acceptable limits. They are defined based on the desired quality or performance standards for a particular process.

In SPC, control limits are typically set at specific values that indicate the upper and lower bounds within which a process should operate to meet the desired specifications. These limits are often determined by analyzing historical data or through statistical methods such as calculating the mean and standard deviation of the process data.

The control limits serve as reference boundaries to monitor the process over time. If the observed data points fall within the control limits, the process is considered to be in control and operating within acceptable parameters. However, if the data points exceed the control limits, it suggests the presence of special causes or variations in the process, indicating a potential issue or problem that requires investigation and corrective action.

Control limits are an essential tool in SPC as they provide a clear indication of when a process is deviating from the desired specifications, allowing for timely intervention and adjustments to maintain the process within acceptable limits.

Parametric χ^2 Distribution (δ^2_α) The parametric χ^2 distribution, denoted as δ^2_α, is a probability distribution that arises in statistical inference. It is commonly used for hypothesis testing and constructing confidence intervals.

The χ^2 distribution is characterized by a single parameter known as the degrees of freedom (ν). The degrees of freedom determine the shape of the distribution. The δ^2_α notation indicates the specific χ^2 distribution with cumulative probability α.

In hypothesis testing, the χ^2 distribution is used to calculate the critical values or p-values associated with test statistics, such as the chi-square test. These critical values or p-values help determine if the observed data is statistically significant or falls within the expected range of variability.

When constructing confidence intervals, the χ^2 distribution is utilized to determine the critical values that define the interval. These intervals provide an estimate of the population parameter with a certain level of confidence.

Overall, the parametric χ^2 distribution (δ_α^2) plays a fundamental role in statistical inference, allowing for hypothesis testing and estimation in a variety of applications. The system is considered in his normal operating conditions if $SPE \leq \delta_\alpha^2$. On the other side, if $SPE > \delta_\alpha^2$, the system is considered in a faulty state where δ denoted for the control limit of SPE [**?**], which can be calculated by using a weighted χ^2 distribution:

$$\begin{aligned} \delta &= g\chi^2_{h,\alpha} \\ g &= \frac{v}{2m} \\ h &= \frac{2m^2}{v} \end{aligned} \tag{2.35}$$

Where m and v are respectively the estimated mean and variance of SPE.

Kernel Density Estimation (KDE): Kernel Density Estimation (KDE)[**?**] is a non-parametric technique used to estimate the probability density function (PDF) of a random variable based on a set of observed data points. It provides a smooth estimate of the underlying distribution, even when the distribution shape is unknown or complex.

The basic idea behind KDE is to place a kernel (a smooth, symmetric function) on each observed data point and then sum up these kernels to obtain a smoothed estimate of the PDF. The choice of kernel function, such as the Gaussian (normal) kernel, determines the shape and smoothness of the estimated density.

To estimate the density at a particular point, the kernels are centered at each data point, and their contributions are weighted by their proximity to the point of interest. The bandwidth parameter controls the width of the kernels and affects the smoothness of the estimated density. A smaller bandwidth produces a more detailed but potentially noisy estimate, while a larger bandwidth leads to a smoother but potentially oversmoothed estimate.

KDE has various applications, including data visualization, outlier detection, and statistical inference. It allows for flexible and data-driven estimation of the underlying distribution, making it a versatile tool in exploratory data analysis and hypothesis testing. However, it should be noted that KDE performance can be influenced by the choice of kernel function and bandwidth selection, which need to be carefully considered based on the specific dataset and analysis goals. Let $X_i (i = 1, 2, ..., n)$ be a

sample taken from a continuous distribution. The KDE of the density function $f(x)$ at any point x is defined as:

$$f(x) = \frac{1}{mh} \sum_{j=1}^{m} K\left(\frac{x - x_j}{h}\right) \tag{2.36}$$

Where K is a Kernel function that integrates to one and has zero mean and $h > 0$ is a bandwidth parameter. The function $K(.)$ is defined as follow:

$$K(x) = (2\pi)^{-m/2} \exp\left(-\frac{1}{2} x^T x\right) \tag{2.37}$$

The KDE approach is capable of estimating the Probability Density Function PDF function underlying the SPEj indices. Then, the local control limits of KDE_j can be obtained from the probability density function SPE_j with an α confidence level by solving the following equation:

$$\int_{-\infty}^{SPE_\alpha} f(SPE) dSPE = \alpha \tag{2.38}$$

As a result, KDE is a well-established approach to estimating PDF, especially for a univariate random variable like SPE.

Principal Component Analysis (PCA) based localization

PCA-based localization (Principal Component Analysis)[**?**] is a data processing method that reduces the dimensionality of data by projecting variables into a lower-dimensional space. Using PCA, variables are transformed into new, uncorrelated variables called principal components. These components are ordered by their ability to explain the variance in the data. The first principal components are the most important for explaining the variance in the data.

PCA-based localization can be used to locate outliers in multivariate data by calculating ***the Mahalanobis distance*** between each data point and the center of gravity of the data. The Mahalanobis distance takes into account the correlation between variables, which is important for multivariate data. Data points that are at a Mahalanobis distance

greater than a predefined threshold can be considered outliers. These points may require further investigation to determine whether they are truly abnormal or simply extreme points in the data. The formula for calculating the Mahalanobis distance is as follows: At time k, we have:

$$D(k) = T^2(k) + T_H^2(k) = \sum_{i=1}^{m} \frac{t_i^2(k)}{\lambda_i} \tag{2.39}$$

The process is considered to be in abnormal operation (presence of a fault) at time k if:

$$D(k) > D_\alpha \tag{2.40}$$

Where D_α is the detection threshold for $D(k)$ given by :

$$D_\alpha = \frac{m(N-1)(N+1)}{N(N-m)} F_{m,N-m,\alpha} \tag{2.41}$$

Where $F_{m,N-m,\alpha}$ is the Fisher distribution with m and $N-m$ degrees of freedom.

Localization by contribution plots: The defect localization by contribution [**?**] [**?**] calculation is a data analysis method used to detect anomalies in complex systems. This method is often used in industry to detect defects in manufacturing processes.

The method involves modeling the complex system as a set of interdependent variables. Then, the contributions of each variable to the overall variation of the system are calculated. The variables that contribute the most to the variation are identified as the most important.

Defect localization by contribution calculation can also be used for fault diagnosis by identifying the variables that contribute the most to the variation in abnormal data. By identifying the most important variables, it is possible to locate the source of the fault.

Contribution plots and SPE (Sum of Squares of Prediction Errors) can be used together to identify the variables or factors that have the greatest impact on process variability.

To create a contribution plot based on SPE, first we need to calculate the SPE for each variable or factor in the process. This can be done using a statistical model that relates the variables to the process output. The SPE value for each variable indicates how much of the overall process variability is explained by that variable. Once we have calculated the SPE values for each variable, we can create a contribution plot that shows the contribution of each variable to the overall process variability. The variables

with the highest SPE values will have the greatest contribution to the process variability, and therefore should be the focus of process improvement efforts. Contribution of variable j to the Q-statistic is calculated as :

$$C_{ijk}^{Q} = e_{ijk}^{2} \tag{2.42}$$

Where $e = (x_i - \hat{x}_i)$.
Using contribution plots based on SPE can help identify the root causes of process variability and guide process improvement efforts. By focusing on the variables with the highest SPE values, process engineers can develop targeted solutions to improve process performance and reduce defects or failures.

Sensor Validity Index (SVI): The Sensor Validity Index (SVI)[**?**] is an index used to evaluate the quality of data provided by a sensor in a measurement or control system.

The SVI is usually calculated by comparing the data provided by a sensor with the data from a reference sensor or with the expected theoretical data under normal operating conditions. The SVI is then defined as the proportion of valid data, i.e., the proportion of data that corresponds to the expected data. A high SVI indicates that the sensor provides reliable and accurate data, while a low SVI indicates that the data provided by the sensor is not reliable and must be treated with caution. The SVI can also be used to detect errors or malfunctions in the measurement or control system by identifying sensors that provide non-valid data.

The calculation of the SVI can be performed in real-time in a measurement or control system, allowing for continuous monitoring of the quality of data provided by sensors and quickly detecting any problems or anomalies. The SVI is therefore an important tool for ensuring the reliability and accuracy of measurement and control systems, especially in critical applications where errors or malfunctions can have significant consequences.
SVI is defined as follows:

$$\eta_j^2(k) = \frac{SPE_j(k)}{SPE(k)} \tag{2.43}$$

Where *SPE* is the quadratic global prediction error computed before reconstruction and SPE_j is the jth quadratic prediction error computed after reconstruction [**?**] [**?**]. The validity index of a faulty sensor must converge towards zero.

2.5 Illustrative example

In this section, we present the application of the PCA method on a synthetic example. The various principles discussed earlier are applied to demonstrate their significance and to explicitly explain their implementation.
We consider a static system governed by 5 variables x_j, where $j = 1,...,5$, and described at different instants k by the following equations:

$$\begin{cases} x_1(k) = u_1(k) + 0.05 \times v_1(k) & v_2 \sim N(0,\sigma^2) \\ x_2(k) = u_2(k) + 0.06 \times v_2(k) & v_1 \sim N(0,\sigma^2) \\ x_3(k) = x_1(k) + 2 \times u_1(k-1) + 0.02 \times v_2(k) & \\ x_4(k) = x_1(k) + 0.7 \times x_2(k) & \\ x_5(k) = u_1(k-1) + 3 \times u_2(k-2) + 0.04 \times v_2(k) & \end{cases}$$

Where u_1 and u_2 are input signals of the system in the form of square waves, whose amplitude and durations vary randomly, and v_1 and v_2 are white noise with variance equal to 1. The data matrix $X = [x_1(k) \quad x_2(k) \quad x_3(k) \quad x_4(k) \quad x_5(k)]^T$ consists of N=1000 observations.

The PCA method studies the correlation between different variables that define a system. The best tool to characterize it is the correlation or covariance matrix. For this example, the correlation matrix is given by:

$$\Sigma = \begin{pmatrix} 1.0000 & 0.0021 & 0.9622 & 0.9296 & -0.0024 \\ 0.0021 & 1.0000 & -0.0094 & 0.0012 & 0.9227 \\ 0.9622 & -0.0094 & 1.0000 & 0.8986 & -0.0139 \\ 0.9296 & 0.0012 & 0.8986 & 1.0000 & 0.0034 \\ -0.0024 & 0.9227 & -0.0139 & 0.0034 & 1.0000 \end{pmatrix} \tag{2.44}$$

Now, the task is to identify the eigenvectors and eigenvalues of this matrix, which correspond, respectively, to the directions of the new orthonormal space and the variances associated with the projection of the data X onto these directions. The matrices of eigenvalues and eigenvectors are given, respectively, by:

$$\Lambda = \begin{pmatrix} 2.8605 & 0 & 0 & 0 & 0 \\ 0 & 1.9228 & 0 & 0 & 0 \\ 0 & 0 & 0.330 & 0 & 0 \\ 0 & 0 & 0 & 0.1069 & 0 \\ 0 & 0 & 0 & 0 & 0.0768 \end{pmatrix} \tag{2.45}$$

$$P = \begin{pmatrix} -0.5838 & -0.0071 & 0.7836 & -0.2123 & 0.0094 \\ 0.0052 & -0.7071 & -0.0192 & -0.0927 & -0.7008 \\ -0.5776 & 0.0017 & -0.5841 & -0.5640 & 0.0847 \\ -0.5706 & -0.0090 & -0.2105 & 0.7882 & -0.0937 \\ 0.0066 & -0.7071 & 0.0127 & 0.0835 & 0.7021 \end{pmatrix} \tag{2.46}$$

Now, once the directions of the new orthonormal subspace are defined, a change of basis is performed to project the correlated variables $(x_1, x_2, ..., x_5)$ onto the directions P. A reduced number of variables that best explain the variability of the original data is obtained. These new variables are called principal components t_i.

After the diagonalization phase of the correlation matrix, the next step is to determine the principal components. This phase is crucial as it is directly linked to the development of the optimal structure of the model.

After presenting the diagonalization phase of the correlation matrix, determining the number of principal components is a crucial step in the PCA method, as it helps identify the optimal structure of the model.

While respecting the basic idea of the PCA method, which is to have a reduced and optimal representation of the information, the retained principal components correspond only to the directions of the greatest dispersion, satisfying the maximum variation of the initial data. The VNR (Variance Non-Reconstructed)(Figure 2.1) criterion is more relevant for fault diagnosis objectives as it takes into account the existing redundancy among the different variables using the principle of reconstruction.

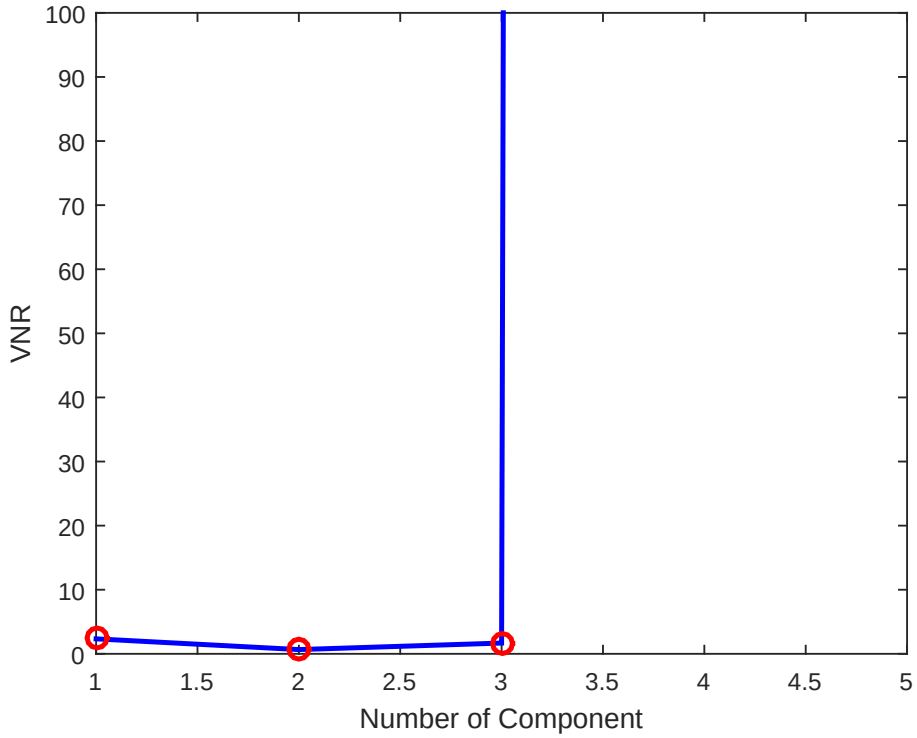

Figure 2.1: Evolution of the reconstruction variance VNR as a function of the number of components l

The PCA model is identified once the optimal number of principal components l is determined. The matrix that characterizes this model is given as follows:

$$\hat{C} = \begin{pmatrix} 0.6588 & -0.0036 & -0.3379 & -0.3326 & -0.0039 \\ -0.0036 & 0.5001 & -0.0022 & 0.0064 & -0.4999 \\ -0.3379 & -0.0022 & 0.6654 & -0.3294 & -0.0025 \\ -0.3326 & 0.0064 & -0.3294 & 0.6756 & 0.0061 \\ -0.0039 & -0.4999 & -0.0025 & 0.0061 & 0.5000 \end{pmatrix} \tag{2.47}$$

Actually, the matrix of observations X can be estimated from the selected principal components corresponding to the l largest eigenvalues (carrying the information) of the covariance matrix. The evolution of the data and their estimation is illustrated in Figure 2.2.

It can be seen in Figure 2.2 that the estimates are consistent with the initial data as well as the estimation errors of all the variables are almost zero.

Using the two indices SPE, SWE, $T2$ and Combined Index for the modelling, we obtain figures 2.3, 2.4, 2.5 and 2.6. We notice threshold overruns this is due to the white noise that we call (false alarms), the difference between a noise and a fault is that a fault is permanent as show in figure 2.7, 2.8, 2.9 and 2.10.

A defect is injected into variable x_3 between instants (800 and 1000) with a bias of 1.2 (30% of the range of variation of variable x_3). The indices SPE, $T2$, SWE, and Combined Index are used to detect the defect. To locate the faulty variable, the contribution calculation method is employed for localization.

As a principle, the highest contribution corresponds to the faulty variable. Figure 2.7 and 2.8 represent the calculation of the SPE contributions at instants $k = 850$ and $k = 900$, respectively, where the defect was detected. It can be observed that the highest contribution is obtained for variable x_3, indicating that it is indeed the faulty variable.

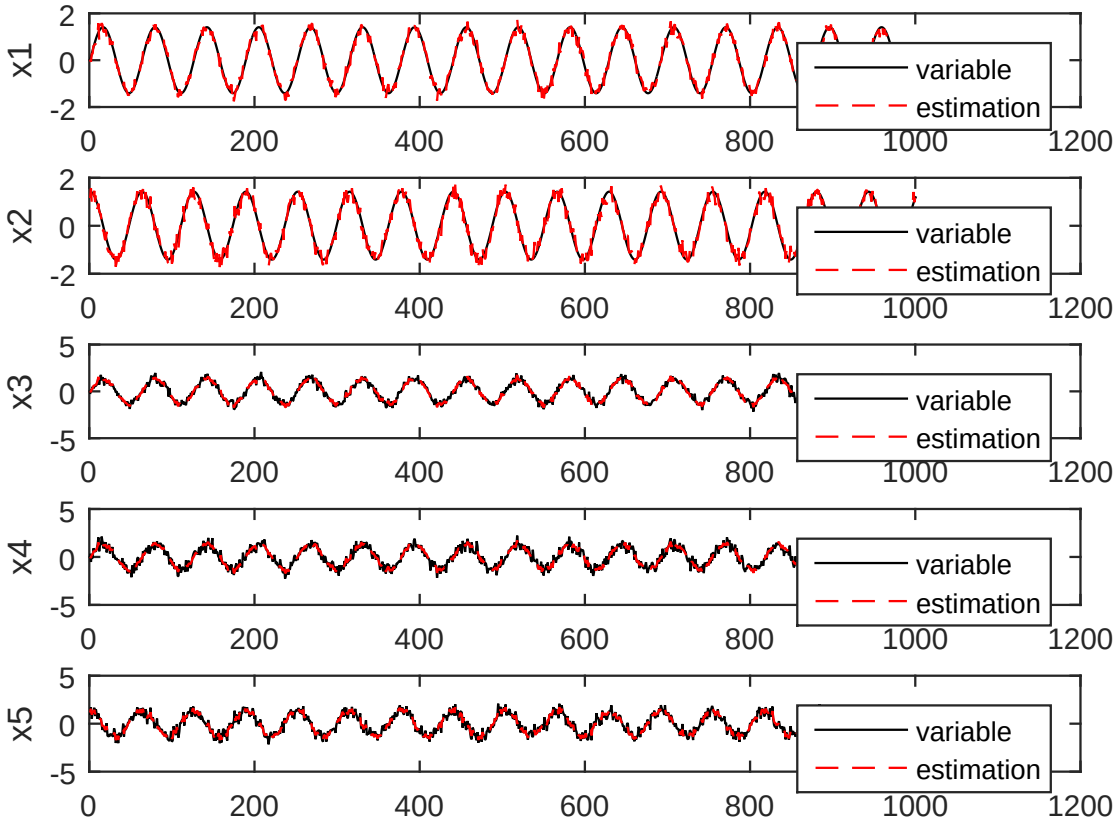

Figure 2.2: Evolution of the different measures and their estimates

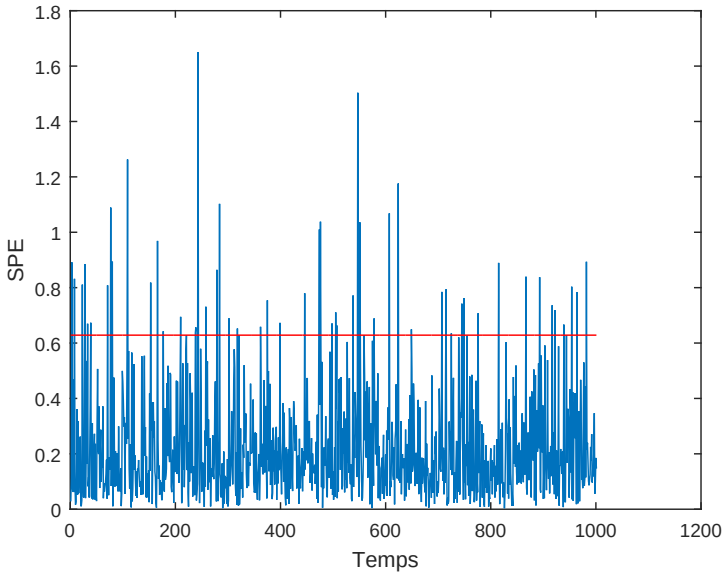

Figure 2.3: Modeling by SPE

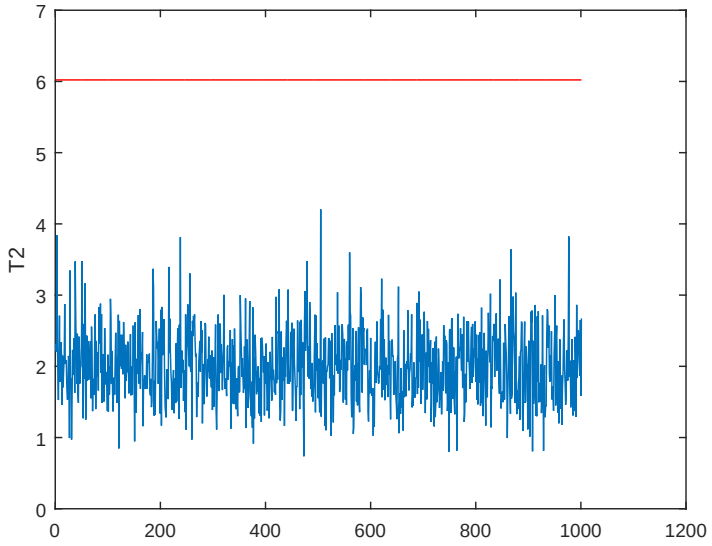

Figure 2.4: Modeling by T2

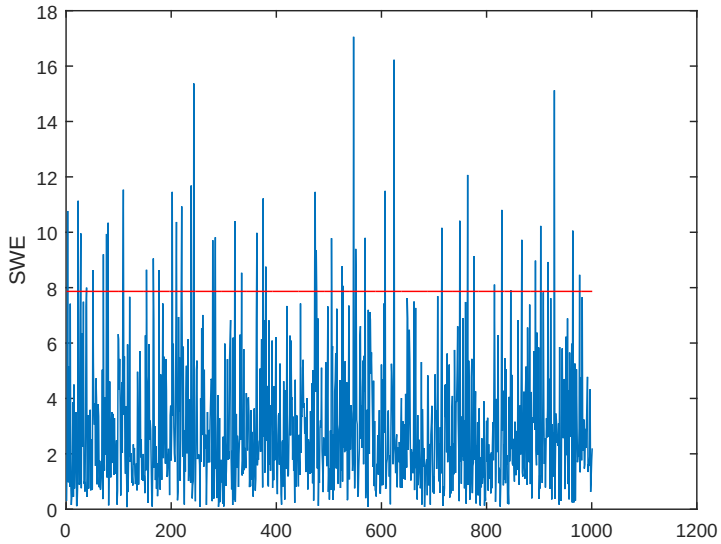

Figure 2.5: Modeling by SWE

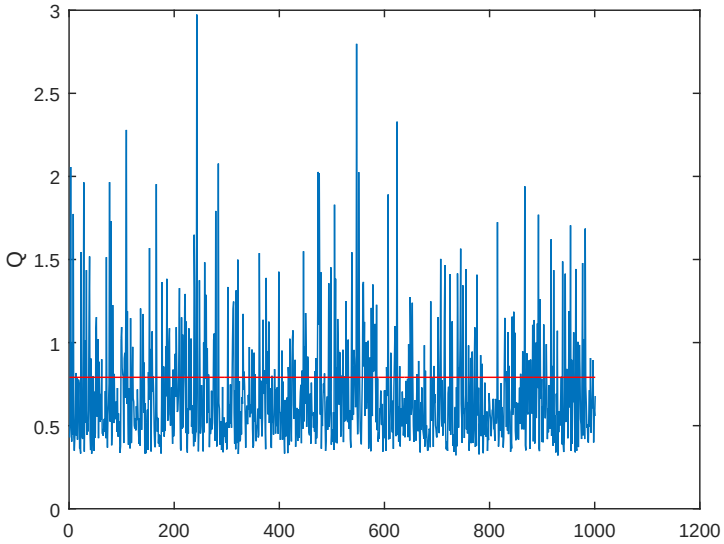

Figure 2.6: Modeling by Combined Index

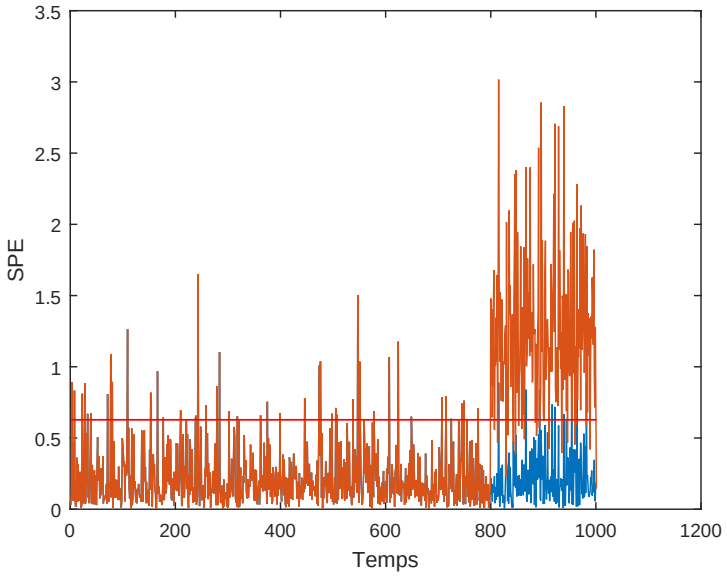

Figure 2.7: Fault detection by SPE

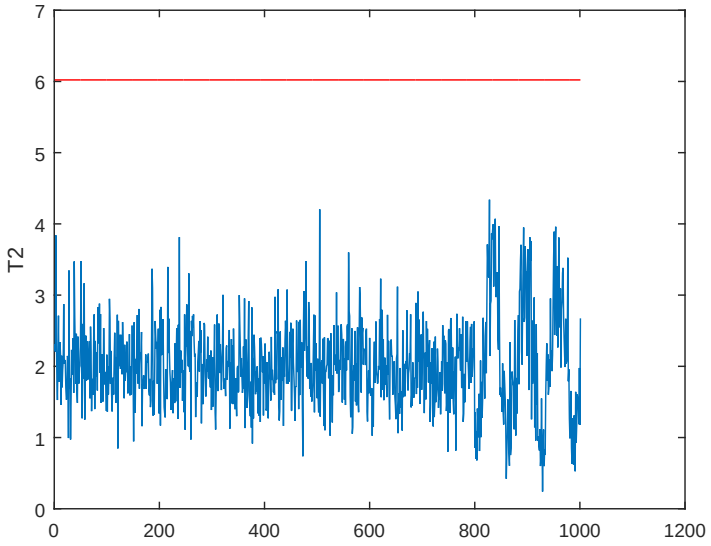

Figure 2.8: Fault detection by T2

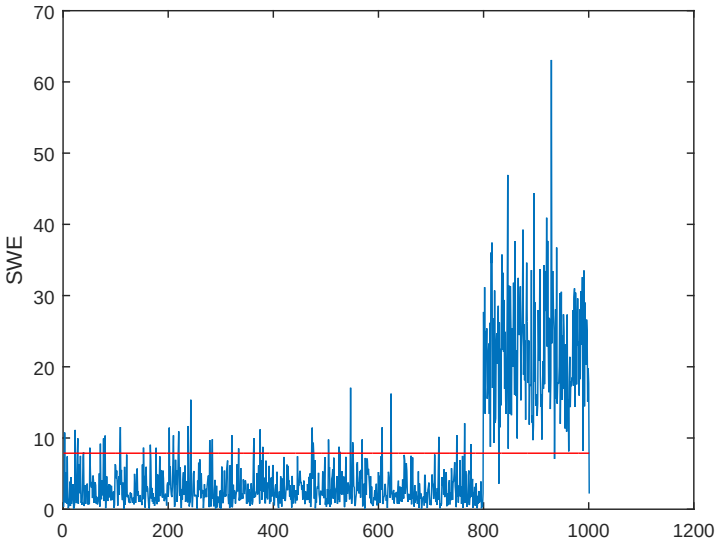

Figure 2.9: Fault detection by SWE

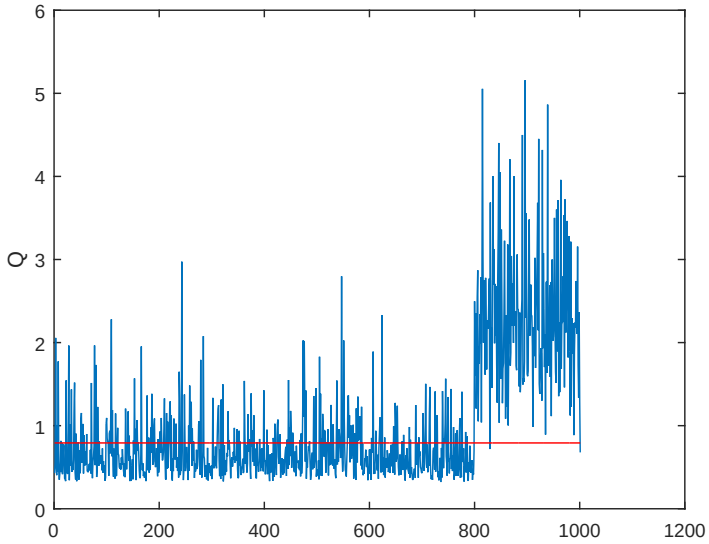

Figure 2.10: Fault detection by Combined Index

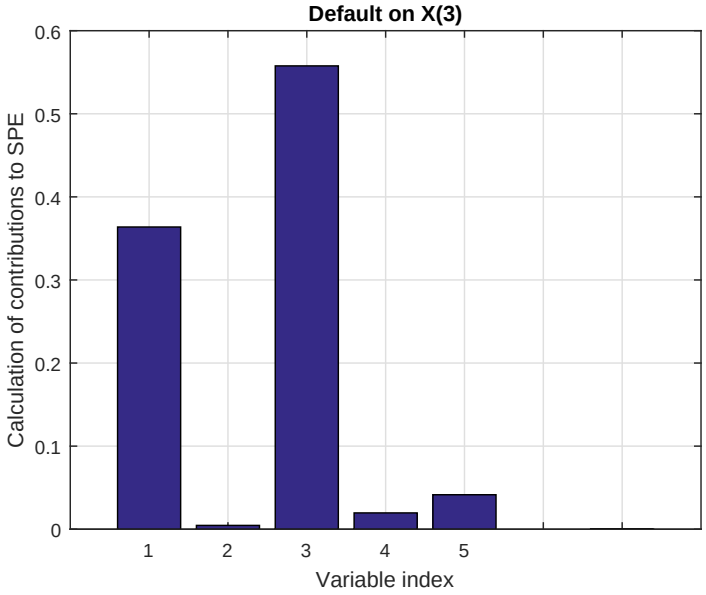

Figure 2.11: Calculation of contributions for the moment (k=850)

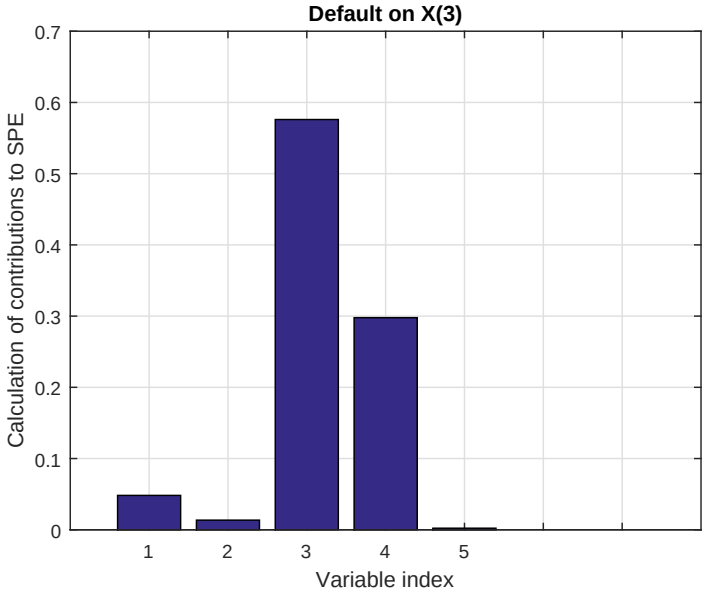

Figure 2.12: Calculation of contributions for the moment (k=900)

2.6 Linear extension of PCA

Principal Component Analysis (PCA) is a multivariate analysis technique that reduces the dimensionality of data by identifying the most important variables and combining them into new variables, called principal components. However, PCA is based on a linear relationship between variables, which means that nonlinear relationships cannot be directly accounted for.

To address this, linear extensions of PCA can be used. Here are some examples:
Centered-reduced PCA:
This extension [**?**] is used to standardize variables before analysis so that they all have zero mean and unit variance. This puts variables on the same scale and avoids variables with different scales dominating the analysis.
Kernel PCA: This extension [**?**] is used to transform data into a higher-dimensional space using a nonlinear kernel function so that nonlinear relationships between variables can be captured. This allows for capturing complex structures in data that cannot be represented in a lower-dimensional space.

Generalized PCA:
This extension [**?**] is used to deal with data with different covariances or non-symmetric covariance matrices. Generalized PCA can be used to identify the principal directions of variation in data and determine the weights of each variable in constructing the principal components.
These extensions allow for taking additional aspects into account in data analysis and improving the performance of PCA for data with more complex relationships between variables.

However, ***Sparse PCA:***
Is a variant of Principal Component Analysis (PCA) that can be used when analyzing high-dimensional data [**?**] where only a small number of variables are expected to be important for capturing the underlying structure.

In traditional PCA, all variables are considered and combined into new principal components. However, in Sparse PCA, a penalty term is added to the optimization objective to encourage the principal components to have many zero weights, effectively removing unimportant variables from consideration. This sparsity constraint can be achieved through various techniques, such as L1 regularization or the use of optimiza-

tion algorithms that promote sparsity. The resulting sparse principal components can be easier to interpret than the dense principal components in traditional PCA, as they only involve a subset of the original variables.

Sparse PCA can be useful in various applications, such as genetics, where many genetic markers are measured but only a small subset are expected to be relevant for a particular trait or disease. There are several types of Sparse PCA. Here are some of the most common:

L1-norm Sparse PCA: [**?**] is a variant of Principal Component Analysis (PCA) that incorporates sparsity regularization. In traditional PCA, the principal components are obtained as linear combinations of the original variables, with coefficients that are not necessarily sparse. However, in L1-norm Sparse PCA, an additional penalty term is introduced to promote sparsity in the coefficients.

The L1-norm penalty, also known as the Lasso penalty, encourages many of the coefficients to be exactly zero, resulting in a sparse representation of the principal components. This sparsity property allows for a more interpretable and concise representation of the data, as only a subset of variables contribute significantly to each principal component.

The L1-norm Sparse PCA formulation involves solving an optimization problem that balances the fit to the data with the sparsity of the coefficients. Various algorithms, such as iterative thresholding or convex optimization methods, can be used to solve this problem efficiently.

L1-norm Sparse PCA finds applications in various fields, including signal processing, image analysis, and feature selection, where identifying a sparse set of important features or components is desired. By promoting sparsity, L1-norm Sparse PCA can enhance interpretability, reduce dimensionality, and improve the efficiency of subsequent analysis tasks.

L0-norm Sparse PCA:

Also known as Compressed Sensing PCA [**?**], is a variant of Principal Component Analysis (PCA) that incorporates a sparsity constraint on the number of non-zero coefficients in the principal components.

In traditional PCA, the principal components are obtained as linear combinations of the original variables, without any constraint on the number of non-zero coefficients. However, in L0-norm Sparse PCA, the goal is to find sparse representations of the principal components, where only a limited number of variables contribute significantly to each component.

The L0-norm, also known as the "pseudo-norm," counts the number of non-zero elements in a vector. In L0-norm Sparse PCA, the objective is to minimize the L0-norm of the coefficients while maintaining a good fit to the data. This sparsity constraint forces the algorithm to identify a small subset of relevant variables that have a significant impact on each principal component.

Finding the exact solution to the L0-norm Sparse PCA problem is computationally intractable, as it involves an exhaustive search over all possible combinations of non-zero coefficients. Therefore, approximate methods are used, such as greedy algorithms or convex relaxations, to find a suboptimal sparse representation.

L0-norm Sparse PCA has applications in various domains, including signal processing, image analysis, and feature selection, where identifying a small set of influential variables or components is crucial. By enforcing sparsity at the level of the number of non-zero coefficients, L0-norm Sparse PCA can provide a highly interpretable representation of the data while reducing the dimensionality and enhancing computational efficiency.

Group Sparse PCA:

Group Sparse PCA [**?**] is a variant of Principal Component Analysis (PCA) that encourages sparsity at the group level rather than at the individual variable level. It aims to identify groups of variables that collectively contribute to the principal components while promoting sparsity within each group.

In traditional PCA, the principal components are obtained as linear combinations of the original variables, with no explicit consideration of group structure. However, in Group Sparse PCA, the data variables are organized into pre-defined groups or clusters based on prior knowledge or domain expertise.

The goal of Group Sparse PCA is to identify a sparse set of groups of variables that have a significant impact on each principal component. This encourages the selection of whole groups of variables instead of individual variables, allowing for a more interpretable representation of the data.

To achieve group sparsity, regularization techniques are employed. These techniques penalize the L1-norm (sum of absolute values) of the coefficients at the group level. By applying this penalty, the algorithm encourages many groups to have zero coefficients, effectively selecting a subset of groups that contribute most significantly to each principal component.

Solving the Group Sparse PCA problem involves optimization techniques such as iterative algorithms or convex optimization methods. These methods find the optimal

combination of groups and coefficients that best approximate the data while enforcing group sparsity.

Group Sparse PCA has applications in various fields, including genetics, neuroscience, and image analysis, where variables naturally group together based on their characteristics or relationships. It provides a more structured and interpretable representation of the data by identifying relevant groups of variables that collectively contribute to the principal components.

These different methods of Sparse PCA can be used depending on the specific needs of the data analysis, but they all have the common goal of reducing the complexity of the analysis by selecting the most important variables.

2.7 Nonlinear extension of pca

Linear Principal Component Analysis (PCA) and non-linear PCA are two methods of data analysis that differ in their approach to finding relationships between variables.

Linear PCA assumes that the relationships between variables are linear. It seeks to reduce the dimensionality of data by finding linear combinations of variables called principal components. These principal components are ordered based on their contribution to the total variance of the data. They allow for the representation of the data in a lower-dimensional space while retaining a maximum amount of information.

On the other hand, non-linear PCA allows for the processing of complex data that are not linearly separable. It uses data transformation techniques to find non-linear relationships between variables. It can be performed using different methods, such as kernel-based non-linear PCA, neural network-based non-linear PCA, or non-linear factor analysis-based PCA.

In summary, linear PCA is effective when the relationships between variables are linear, while non-linear PCA allows for the processing of more complex data by finding non-linear relationships between variables. The non-linear extension of Principal Component Analysis (PCA) is a method used to process complex data that are not linearly separable. Unlike linear PCA, which assumes that the relationships between variables are linear, non-linear PCA uses data transformation techniques to find non-linear relationships between variables.

The principle of nonlinear PCA is to extend the traditional PCA method to capture nonlinear structures in the data. While linear PCA relies on assumptions of linearity between variables, nonlinear PCA allows for modeling more complex and nonlinear relationships.

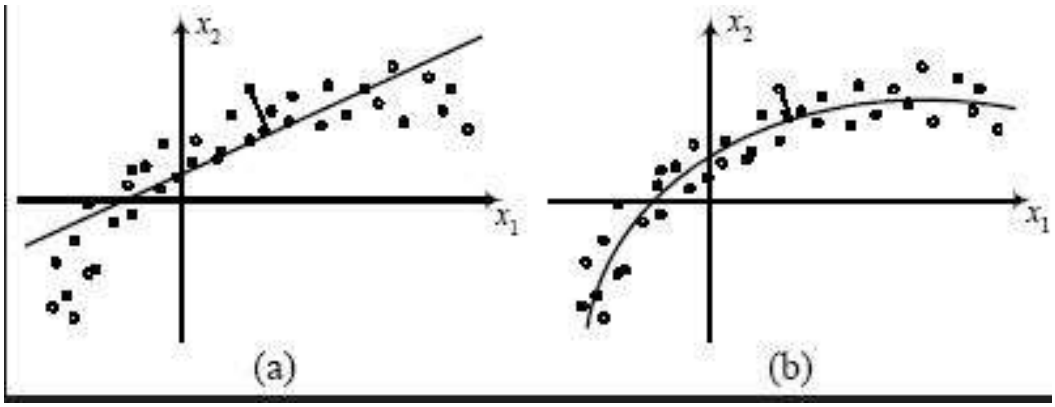

Figure 2.13: PCA and NLPCA Models

The central idea of nonlinear PCA is to transform the original data using nonlinear functions in order to project them into a higher-dimensional space where nonlinear structures can be better represented. This transformation can be achieved through various techniques such as kernel methods, neural networks, or nonlinear dimensionality reduction methods.

Once the data is transformed into the higher-dimensional space, PCA is applied to extract the nonlinear principal components. These components are linear combinations of the original variables in the transformed space and represent the directions of maximum variance in that space.

The nonlinear principal components are ordered based on their contribution to the total variance of the data. The first principal components will capture the most important nonlinear structures, while subsequent components will explain less important variations.

The advantage of nonlinear PCA is that it allows for a better representation and understanding of complex data with nonlinear relationships. It is useful when the data cannot be simply explained by linear relationships and when more complex interactions between variables need to be taken into account.

In summary, nonlinear PCA extends the traditional PCA method to accommodate nonlinear structures in the data. It enables the capture and analysis of nonlinear relationships between variables, providing a better understanding of the data and underlying models. There are different methods to perform Nonlinear Principal Component Analysis (NLPCA). Here are some commonly used methods:

1. **Kernel PCA:**
   Kernel PCA is a method that extends traditional PCA to capture nonlinear structures in the data. It utilizes kernel functions to perform a nonlinear mapping of the data into a higher-dimensional feature space where linear PCA can be applied.

   In Kernel PCA, the data is first mapped into a higher-dimensional space using a kernel function such as the Gaussian (RBF) kernel or polynomial kernel. This transformation allows for capturing nonlinear relationships and structures that may not be evident in the original feature space.

   Once the data is mapped into the higher-dimensional space, linear PCA is performed to extract the principal components. These components represent the directions of maximum variance in the transformed feature space and correspond to the most important nonlinear patterns or structures in the data.

   The choice of the kernel function is crucial in nonlinear Kernel PCA, as it determines the type of nonlinear mappings applied to the data. Different kernel functions have different properties and are suitable for capturing different types of nonlinearities in the data.

   Nonlinear Kernel PCA can be a powerful tool for dimensionality reduction and nonlinear feature extraction. It has applications in various fields, including computer vision, pattern recognition, and bioinformatics, where nonlinear relationships are common.

   It is worth noting that nonlinear Kernel PCA is computationally more demanding than traditional linear PCA, especially for large datasets, as it involves the computation of kernel matrices. However, it provides a valuable approach to explore and analyze nonlinear structures in the data.

2. **Autoencoders:**
   An autoencoder is a type of artificial neural network used for unsupervised learning and dimensionality reduction. It is designed to learn a compressed representation of the input data, called the latent space or encoding, and then reconstruct the original input data from this latent representation.

The structure of an autoencoder consists of an encoder and a decoder. The encoder network takes the input data and gradually reduces its dimensionality, compressing it into a lower-dimensional latent space. The decoder network then takes this compressed representation and attempts to reconstruct the original input data.

During the training process, the autoencoder aims to minimize the reconstruction error, which is the difference between the original input data and the reconstructed output. By doing so, the autoencoder learns to extract the most important features and patterns in the data, effectively capturing its underlying structure.

The compressed latent space learned by the autoencoder can be used for various purposes, such as data compression, denoising, anomaly detection, and dimensionality reduction. By discarding or manipulating certain components of the latent space, it is possible to generate new data samples or perform data generation tasks.

Autoencoders can be shallow with only one hidden layer, or they can be deep with multiple hidden layers, forming a stacked autoencoder. Deep autoencoders are capable of learning more complex representations and capturing hierarchical structures in the data.

Overall, autoencoders provide a powerful tool for learning compact representations of complex data and have found applications in various domains, including image and text analysis, feature extraction, and anomaly detection.

3. **Manifold Learning:** These methods aim to model the intrinsic structure of the data by considering it as sampled from a manifold or sub-space. Techniques like Isomap, Locally Linear Embedding (LLE), and t-Distributed Stochastic Neighbor Embedding (t-SNE) are examples of Manifold Learning methods that can be used for NLPCA.

4. **Neural Network-based Methods:**
   Neural networks, especially deep architectures like convolutional neural networks (CNN) or recurrent neural networks (RNN), can be used to perform a nonlinear transformation of the data and extract the principal features.

5. **Nonlinear Extensions of PCA Algorithms:**
   Some methods directly extend traditional PCA algorithms to handle nonlinear

data. For example, Kernel PCA is an extension of traditional PCA that uses kernel functions to handle nonlinear data.

These methods vary in approach and complexity, and it is important to choose the most suitable method based on the data and the specific objective of the analysis.

2.8 Conclusion

In conclusion, we have provided a comprehensive overview of Principal Component Analysis (PCA), Sparse PCA, and modeling and diagnostics based on multivariate statistical data analysis. We have explored the underlying principles and methodologies of these techniques, highlighting their ability to capture complex relationships and patterns within dynamic systems. These advanced approaches offer valuable insights into the underlying dynamics and enable more accurate and effective diagnostics.
As we move forward, Chapter 3 will focus on the contribution of statistical control methods for nonlinear processes. We will delve into the advantages and limitations of these approaches in maintaining the stability and performance of dynamic systems. By combining the knowledge gained from Chapter 2 with the statistical control techniques discussed in Chapter 3, we aim to develop a comprehensive framework for the diagnosis and operation of dynamic systems.

Chapter 3

CONTRIBUTION TO STATISTICAL CONTROL OF NONLINEAR PROCESSES

3.1 Introduction

Chapter 3 focuses on the contribution of statistical control methods in the context of nonlinear processes, with a specific emphasis on the utilization of Stacked Sparse Autoencoders (SSAE). Nonlinear processes are widely prevalent in various fields, including engineering, biology, finance, and many others. However, their intricate dynamics and complex interactions pose significant challenges in terms of modeling, control, and fault detection.

Traditional statistical control methods, such as control charts and regression analysis, are primarily designed for linear systems and may not effectively capture the nonlinear behavior of complex processes. Therefore, there is a growing need for advanced techniques that can address the nonlinear nature of these processes and provide more accurate and reliable diagnostic capabilities.

In this chapter, we explore the application of Stacked Sparse Autoencoders (SSAE) as a powerful tool for statistical control of nonlinear processes. SSAE is an artificial neural network model that has gained significant attention in recent years due to its ability to learn hierarchical and nonlinear representations of data. By leveraging the inherent structure of SSAE, we can extract meaningful features and patterns from

complex and high-dimensional data, enabling us to detect anomalies and identify faults in nonlinear processes more effectively.

We will delve into the theoretical foundations of SSAE and its training algorithms, emphasizing its ability to learn hierarchical representations through unsupervised learning. We will also discuss the advantages of SSAE over traditional linear methods, highlighting its capacity to capture nonlinear relationships, handle high-dimensional data, and adapt to varying operating conditions.

Furthermore, we will explore different approaches to integrating SSAE into the statistical control framework. This includes methods for feature extraction, fault detection, and fault localization using the learned representations from SSAE. We will discuss the application of various statistical techniques, such as clustering, classification, and anomaly detection, to leverage the extracted features for effective diagnostic and control of nonlinear processes.

The chapter will also address practical considerations, such as data preprocessing, model validation, and interpretation of results, to ensure the robustness and reliability of the proposed approach. Additionally, we will examine the computational aspects of SSAE and discuss strategies for efficient implementation in real-time systems.

Through this chapter, we aim to demonstrate the potential of Stacked Sparse Autoencoders (SSAE) as a valuable tool in the statistical control of nonlinear processes. By harnessing the power of SSAE, we can enhance our understanding of complex system behavior, improve fault detection capabilities, and facilitate the control and optimization of nonlinear processes.

3.2 Neural networks

An artificial neural network [**?**] can be defined as a model that is inspired by the functioning of the biological neuron. The biological neuron consists of three essential components (See Figure 3.1):

Dendrites: They serve as the inputs of the neuron and are responsible for transmitting electrical signals from other neurons to the neuron's cell body.

Cell body: It accumulates electrical charges. If the neuron becomes sufficiently excited (i.e., the electrical charge exceeds a certain threshold), it generates an electrical

potential that propagates through its axon to excite other neurons.

Axon: It represents the output of the neuron to other neurons. The point of contact between the axon of one neuron and the dendrite of another neuron is called a synapse.

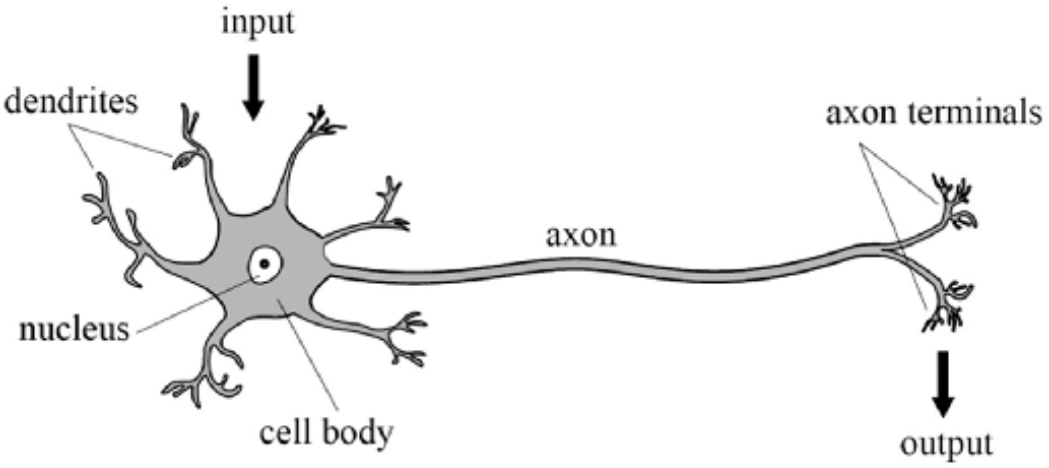

Figure 3.1: The biological neuron

In parallel to the biological neuron, as illustrated in Figure 3.2, an artificial neuron receives inputs $x_1, x_2, ..., x_p$ in vector form, and then performs an affine combination of its inputs minus the neuron's activation threshold or bias $b(w_{i,1}x_1 + w_{i,2}x_2 + ... + w_{i,p}x_p - b)$. Then, an activation function f is applied to this output:

$f(w_{i,1}x_1 + w_{i,2}x_2 + ... + w_{i,p}x_p - b) = f(\sum_{k=1}^{p} w_{i,p}x_i - b)$ to create an output y. The parameters $w_{i,1}x_1, w_{i,2}, ..., w_{i,p}$ and b respectively represent the weights and bias of the neuron. A set of interconnected neurons forms what is called a neural network. The weights and biases are estimated during the learning phase of the neural network.

There are two types of neural network learning: supervised learning and unsupervised learning. Supervised learning is characterized by the presence of a teacher who has knowledge about the environment in which the network operates. However, unsupervised learning is performed on data where the labels of observations are unknown. It is characterized by the absence of a teacher. The goal is to group statistical individuals that share common characteristics.

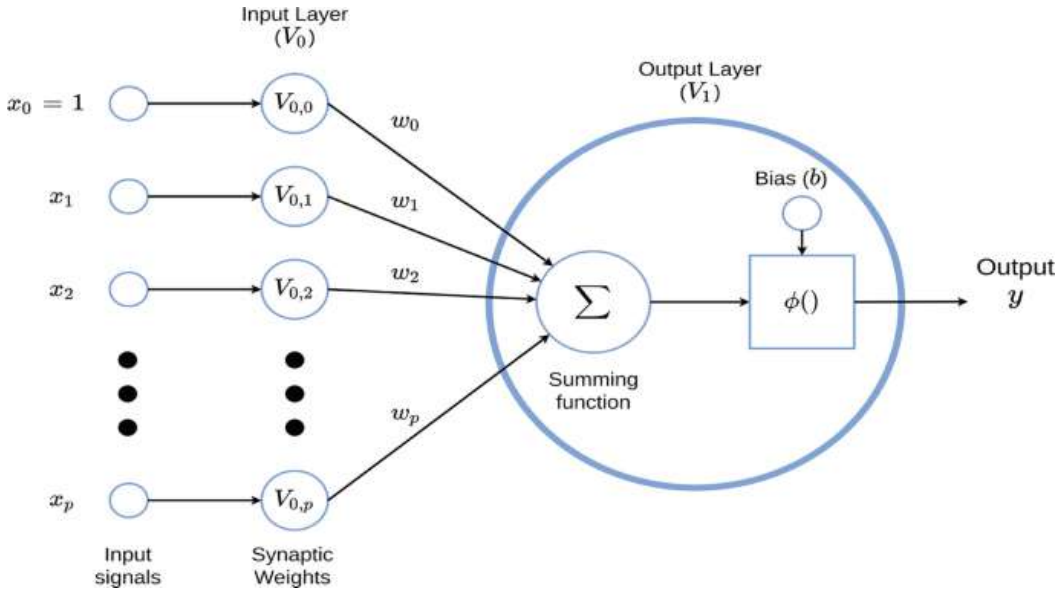

Figure 3.2: The artificiel neuron

3.2.1 Activation funtiocn

An activation function [**?**] is a mathematical function used in neural networks to introduce non-linearity into the network's output. It is applied to the weighted sum of inputs and bias in each neuron, producing the output of the neuron. Activation functions play a crucial role in determining the overall behavior and learning capability of a neural network. They can help the network learn complex patterns, make decisions, and model non-linear relationships in the data.

Several activation functions are used in artificial neural networks. Among them, we can mention:
The threshold function: $f(x) = 1_{[0,+\infty]}$ (Figure 3.3)
The linear function: $f(x) = ax, a \in \Re^*, x \in \Re$ (Figure 3.4)
•

The sigmoid function: $f(x) = \frac{1}{1+e^{-x}}, x \in \Re$ (Figure 3.5)

The Rectified Linear Unit (ReLU) function: $f(x) = \max(0,x), x \in \Re$ (Figure3.6)

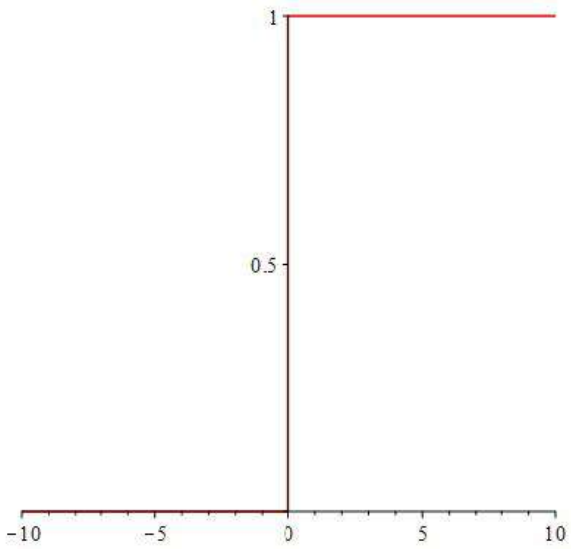

Figure 3.3: The threshold function

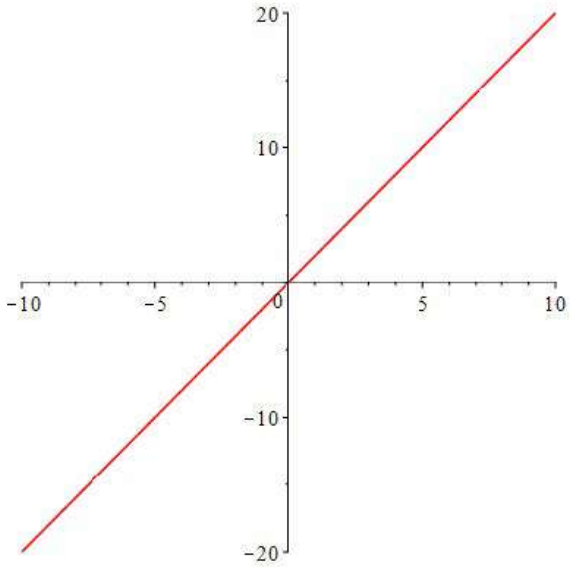

Figure 3.4: The linear function

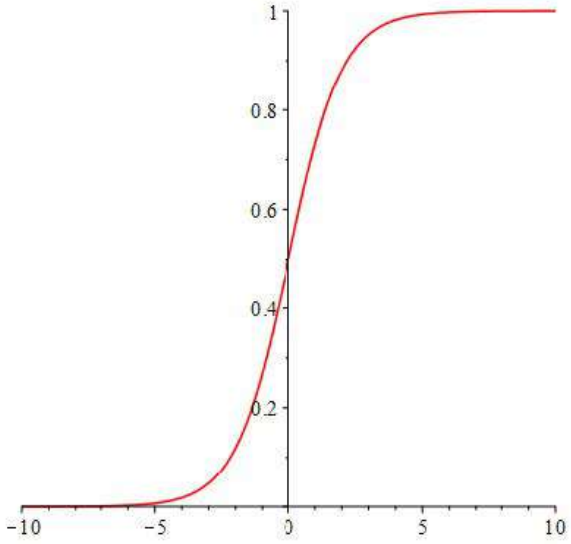

Figure 3.5: The sigmoid function

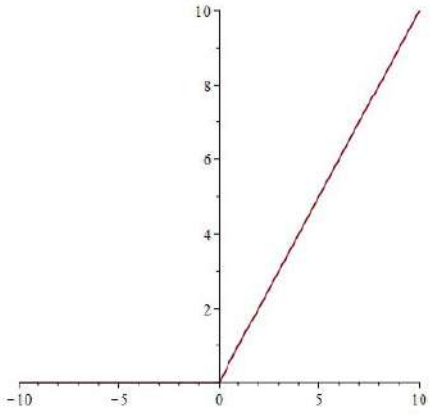

Figure 3.6: The Rectified Linear Unit (ReLU) function

3.2.2 Gradient descent

The training of a neural network is typically performed using the method of gradient descent. As illustrated in Figure 3.7, gradient descent aims to improve the network by iteratively updating the weights to minimize the objective function. The objective function is typically the mean squared error (3.1) :

$$E = \sum_{i=1}^{n} L(x_i, y_i) = \frac{1}{n} \sum_{i=1}^{n} (x_i - y_i)^2 \tag{3.1}$$

L, being a cost function that measures the divergence between the input training sample and the outputs.

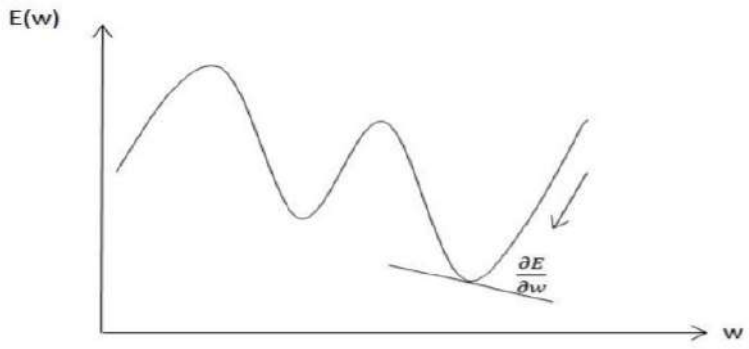

Figure 3.7: The gradient descent

More specifically, the steps of the gradient descent algorithm are as follows:

1. Initialization phase: Randomly initialize the weight matrix w.
2. Before the learning process begins, input the training data x_i to the network through forward propagation to obtain the outputs y_i.
3. Calculate the objective function between x_i and y_i and minimize it with respect to the weights w by solving the equation $\frac{\partial E}{\partial w} = 0$.
4. Perform backpropagation of the gradient of the objective function. This involves updating the weights of the neurons from the output layer to the input layer to minimize the objective function.
5. Iterate the weight update process until the objective function converges to a global minimum.

Informally, the term "backpropagation of the gradient" is also used to refer to both the gradient descent algorithm and the backpropagation process.

3.3 Autoencoder

An autoencoder [**?**] is a type of artificial neural network used for dimensionality reduction (like PCA), data compression, and data generation. It is composed of two main parts: the encoder and the decoder .

The encoder transforms the input data into a lower-dimensional latent representation, which captures the most important information of the input data. This latent representation is often called a "code" or "embedding". The decoder then takes this code and decodes it to reconstruct the input data (Figure 3.8).

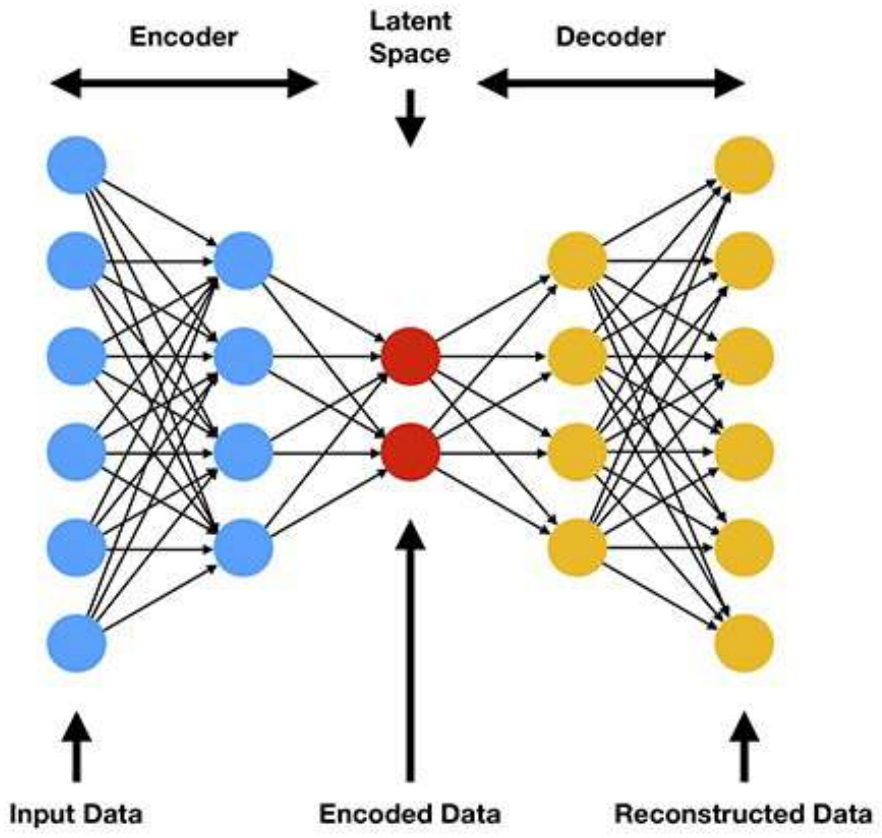

Figure 3.8: Autoencoder architecture

The goal of the autoencoder is to minimize the difference between the input data and the reconstructed output data, also called "reconstruction error". This means that the autoencoder learns to reconstruct the input data with the highest possible accuracy using the most compact latent representation possible. The autoencoder can also be used to generate new data similar to the input data, by sampling random values for the latent code and decoding these values to generate new data. This can be useful in applications such as data synthesis for machine learning.

The autoencoder is often used as an unsupervised model because it does not require labels to learn how to reconstruct the input data. However, it can also be used as a supervised model by adding a classification layer to the encoder or decoder, allowing the autoencoder to learn to reconstruct the input data while performing a classification task.

The autoencoder is a neural network model that learns a compressed representation of data using an encoding and decoding process. The model can be mathematically represented as follows:

Let X be an input vector of dimension p, and Y an output vector of the same dimension p. The autoencoder is composed of two parts: the encoder and the decoder. The encoder takes X as input and produces a compressed representation Z of dimension k, where $k << p$. This step can be mathematically represented as follows:

$$\mathrm{Z} = \mathrm{f}(\mathrm{XW} + \mathrm{b}) \tag{3.2}$$

where f is a nonlinear activation function, W is a weight matrix, and b is a bias vector. The decoder takes Z as input and produces a reconstruction Y' of X. This step can be mathematically represented as follows:

$$\mathrm{Y}' = \mathrm{g}(\mathrm{ZW}' + \mathrm{b}') \tag{3.3}$$

where g is a nonlinear activation function, W' is a weight matrix, and b' is a bias vector. The objective of the autoencoder is to minimize the difference between X and Y', which can be formulated as a cost function:

$$\mathrm{L}(\mathrm{X}, \mathrm{Y}') = ||\mathrm{X} - \mathrm{Y}'||^2 \tag{3.4}$$

Where $||.||$ is the Euclidean norm.

The cost function can be optimized using gradient descent techniques to find the parameters W, b, W', and b' that minimize the difference between X and Y'. Once the model is trained, the encoder can be used to generate compressed representations of new data, and the decoder can be used to reconstruct the original data from these representations.

3.3.1 The relationship between antoencoder and PCA

PCA can be seen as a linear autoencoder with $W \in \Re^{p*k}$, where $k \leq p$. By taking $f_\theta(X) = XW$ and $g_\psi \circ f_\theta(X) = XWW^T$, PCA aims to optimize the objective function $\left\|X - XWW^T\right\|^2$. Therefore, PCA is an autoencoder with a linear activation function and the mean squared error as the cost function.

However, an autoencoder using non-linear activations has much greater capacity and can learn more complex representations. Thus, the autoencoder is a generalization of PCA. It represents a form of non-linear PCA. This form of non-linear PCA will be further developed in this paper.

3.3.2 The parameters to define for training an autoencoder

The parameters to define for training an autoencoder include:

1. Number of hidden layers: This refers to the number of layers between the input and output layers. It determines the depth and complexity of the autoencoder.
2. Number of neurons per hidden layer: This parameter determines the size of each hidden layer and affects the capacity of the autoencoder to learn complex representations.
3. Activation functions: These functions introduce non-linearity into the autoencoder and can include options such as sigmoid, ReLU, or tanh. The choice of

activation functions can impact the learning capacity and performance of the autoencoder.

4. Loss function: This function quantifies the difference between the input and output of the autoencoder. Common loss functions used in autoencoders include mean squared error (MSE) or binary cross-entropy.
5. Optimization algorithm: This algorithm determines how the autoencoder updates its weights during training.
6. Learning rate: This parameter controls the step size at each iteration of the optimization algorithm. It influences the convergence speed and stability of the training process.
7. Regularization techniques: Regularization methods such as L1 or L2 regularization can be applied to prevent overfitting and promote generalization of the autoencoder.
8. Batch size: It specifies the number of samples processed before the weights are updated. Larger batch sizes can lead to more stable training but may require more memory.
9. Number of epochs: An epoch is a complete pass through the entire training dataset. The number of epochs determines how many times the autoencoder will iterate over the training data during the training process.

These parameters need to be carefully tuned and optimized to ensure effective training and good performance of the autoencoder.

3.3.3 Types of autoencoders

We can distinguish two types of autoencoders based on the additional constraint imposed to limit the representation capacity of the autoencoder during the optimization of the objective function. On one hand, we have "undercomplete" autoencoders, which restrict the dimension of the latent code. On the other hand, we have "overcomplete" autoencoders, which add a regularization term to the objective function.

Undercomplete autoencoders Undercomplete autoencoders [**?**] are a type of autoencoder where a constraint is imposed on the code layer to have useful representations. This constraint involves limiting the size of the code layer. In this case, similar to

PCA, the autoencoder extracts the primary characteristics of the data by reducing the dimensionality of the dataset. Thus, PCA can be considered as an undercomplete autoencoder that uses a linear activation function.

Sparse autoencoders Sparse autoencoders [**?**] use an alternative method to introduce the bottleneck without requiring a reduction in the number of nodes in the hidden layers. Specifically, they construct an objective function by penalizing the activations in a layer based on the input observation. The neural network performs encoding and decoding based on the activation of a certain number of neurons. One strategy is to add an additional term to the objective function during training to penalize the Kullback-Leibler divergence between the hidden units' marginals $\hat{\rho}_j$and a desired sparsity rate ρ as shown in equation 3.5.

$$J_{AE}(\theta,\psi) = \sum_{i=1}^{n} L(x_i,\hat{x}_i) + \lambda \sum_{j=1}^{m} KL(\hat{\rho}_j||\rho) \tag{3.5}$$

where L is a cost function used to measure the divergence between the input training sample and the reconstructed data.
This part will be further developed in this paper.

Denoising Autoencoders Another way to limit the capacity of representation of an autoencoder is to add noise [**?**] to the input data during training in order to have outputs that are as close as possible to the inputs. The autoencoder is trained to reconstruct its input x from its corrupted version $\tilde{x}$ as shown in equation 3.6.

$$J_{AE}(\theta,\psi) = \sum_{i=1}^{n} L(x_i, g_\psi(f\theta(\tilde{x}_i)) \tag{3.6}$$

Where L is a cost function used to measure the divergence between the input training sample and the reconstructed data.

Contractive Autoencoders is an autoencoder in which we penalize the Frobenius norm of the Jacobian matrix of the encoder's activations with respect to the input by adding an additional term to the objective function in order to learn useful representa-

tions (equation 3.7) :

$$J_{AE}(\theta,\psi) = \sum_{i=1}^{n} L(x_i,\hat{x}_i) + \lambda \sum_{i=1}^{m} \|\nabla x z_i\|_F^2 \tag{3.7}$$

F, L is a cost function used to measure the divergence between the input training sample and the reconstructed data, z_i is the activation of hidden unit i, and m is the number of hidden units. The Frobenius norm is defined as follows: $\|A\|_F = tr(A*A)$ where $A \in M_{m,n}$, A^* is the adjoint matrix of A.

3.4 Statistical Process Monitoring (SPM) Method: Stacked Sparse Autoencoders (SSAE)

Deep learning is a subset of machine learning that focuses on training neural networks to learn high-level representations of data. Although these methods were initially developed in the 1980s, they were largely abandoned due to limited success. However, in recent years, deep learning has made significant advancements and achieved remarkable performance in various tasks. Despite this, its application in process control remains relatively uncommon. Deep neural network models, which include a hidden layer known as the bottleneck layer, are utilized to predict patterns and extract valuable insights. First, the input vector xi = 1, 2, 3, ..., N is transformed into a hidden part represented by the function hi, which is as follows: First, the input vector $x_i = 1,2,3,...,N$ is transformed into a hidden part represented by the function h_i, which is as follows:

$$h_i = f(x_i) = sigm(W_1 x + b_1) \tag{3.8}$$

Where W_1 and b_1 are respectively the weight and the bias between the input layer and the hidden part and $sigm(x)$ is a sigmoid function that is calculated as follows:

$$sigm(x) = (1+\exp(-x))^{-1} \tag{3.9}$$

At the decode layer, h_i gets mapped to the output enunciated by $\hat{x}$. Where we employ the activation function represented as below:

$$\hat{x}_i = g(h) = sigm(W_2 h_i + b_2) \tag{3.10}$$

Where W_2 and b_2 represent the weight and the bias from the hidden to the output layer, as well ($\hat{x}$).

An autoencoder for which the learning parameter implies a sparsity penalty [20] is simply called Stacked Sparse Autoencoder (SSAE) [**?**]. Stacked Sparse Autoencoders (SSAE) [**?**] are a type of deep learning neural network that are used for unsupervised learning and feature extraction. They are similar to the traditional Autoencoder (AE) model, but instead of using a single hidden layer, they use multiple hidden layers. Each hidden layer acts as an encoder, transforming the input data into a compressed representation, and each subsequent layer acts as a decoder, reconstructing the original data (Figure 3.9).

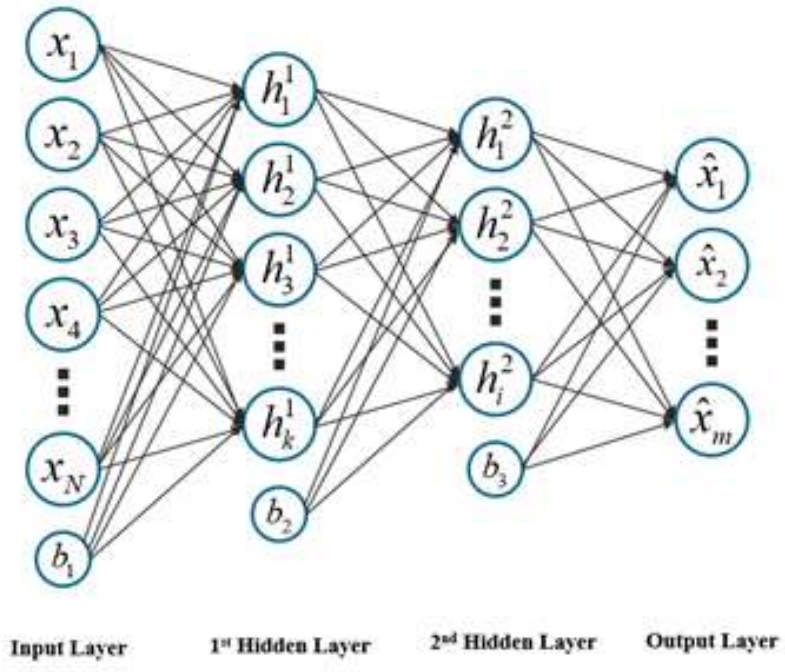

Figure 3.9: Structure of the proposed SSAE model

The main advantage of using SSAEs over traditional AEs is that they are capable of learning more complex and abstract features in the input data, as each layer learns to

extract higher-level features based on the output of the previous layer. This makes them particularly useful for tasks such as image and speech recognition, where high-level features are essential for accurate classification.

SSAEs also incorporate the concept of sparsity, where only a small number of neurons in each layer are activated at a time. This helps to reduce overfitting and improves generalization performance. Additionally, SSAEs often use dropout regularization to further improve their robustness to overfitting.

Training SSAEs can be challenging, as they require a large amount of data and computation power. However, they have been shown to outperform traditional AEs on a variety of tasks, making them a powerful tool for unsupervised feature learning and representation.
The power of the proposed network consists in predicting its output (input estimate) to be as similar as its input, by optimizing the cost function given by:

$$J = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{1}{2}\left\|\hat{x} - x_i\right\|^2\right) + \frac{\lambda}{2}\sum_{i=1}^{N}\|W_i\|^2 + \beta\sum_{j=1}^{m} KL(\rho\,\|\,\hat{\rho}_j) \tag{3.11}$$

Where m is the number of hidden nodes. λ and β are the coefficient determining the weight decay and the sparsity penalty terms respectfully. In the equation 3.11, the first term is the reconstruction error, the second is the normalization term and the final one is the Sparsity Penalty, where $KL(\rho\,\|\,\hat{\rho}_i)$ is the Kullback-Leibler divergence, is used to compute the difference between ρ and $\hat{\rho}_i$ being the constraint used during learning. $KL(\rho\,\|\,\hat{\rho}_i)$ is defined as:

$$KL(\rho\,\|\,\hat{\rho}_i) = \rho\log\frac{\rho}{\hat{\rho}_i} + (1-\rho)\log\frac{1-\rho}{1-\hat{\rho}_j} \tag{3.12}$$

The goal in the use of the SSAE is to train it using the back propagation algorithm and L-BFGS to reduce the cost function, to define the adequate parameters W_1, W_2, b_1, b_2.

3.4.1 Anomaly Detection

In the context of novelty detection, SSAE can be trained on a dataset consisting of normal or expected patterns. The model learns to reconstruct these patterns accurately, and any deviation from the learned representations can be considered as a novelty

or anomaly. By comparing the reconstruction error of unseen data with a predefined threshold, SSAE can identify and flag novel or anomalous instances.

The advantage of using SSAE for novelty detection lies in its ability to learn intricate patterns and capture subtle deviations from the learned representations. This makes it effective in detecting anomalies or novelties that may not be easily discernible using traditional approaches.

When it comes to detection techniques like SPE the most commonly used index, it follows the same principle as the one explained in Chapter 2.

3.4.2 Control Limits

As detailed in Chapter 2, static control limits are primarily used for process monitoring and diagnosis. When the observed data or measurements fall outside the predetermined control limits, it suggests that the process or system is experiencing an abnormal condition or variation. This deviation from the expected behavior can be indicative of potential issues or problems that require attention.

The diagnosis of a process using static control limits involves analyzing the data points that fall outside the control limits and investigating the potential causes. This analysis can help identify the root cause of the deviation and guide corrective actions to bring the process back within acceptable limits.

By continuously monitoring the process using static control limits, deviations can be detected in real-time, allowing for prompt intervention and troubleshooting. This proactive approach to diagnosis helps prevent quality issues, minimize downtime, and optimize process performance.

Adaptatif threshold using K-means based on Kernel Density Estimation (KDE) (UCLkmeans)

Many clustering algorithms for different problems have been proposed, including partition-based clustering, hierarchical clustering, neural network clustering, mixture model clustering and kernel clustering. Many classification algorithms for various tasks have been developed.

Thresholds using K-means based on Kernel Density Estimation (KDE) [**?**] is a method used to determine thresholds for anomaly detection or classification tasks. It combines the K-means clustering algorithm with Kernel Density Estimation to estimate the

density distribution of data points and identify thresholds for distinguishing normal and anomalous observations. Here's an explanation of how thresholds using K-means based on KDE are determined:

1. Data preprocessing: The dataset of interest is prepared by removing any irrelevant or redundant features and normalizing the data if necessary.
2. K-means clustering: The K-means algorithm is applied to the preprocessed dataset to group similar data points into K clusters. K is a user-defined parameter representing the desired number of clusters.
3. Cluster centroids: The centroid of each cluster is calculated as the mean of the data points belonging to that cluster. These centroids represent the representative values for each cluster.
4. Kernel Density Estimation: Kernel Density Estimation is performed on each cluster to estimate the density distribution of data points within that cluster. Kernel functions, such as Gaussian or Epanechnikov, are used to model the density.
5. Threshold determination: The estimated density distributions are analyzed to determine appropriate thresholds for anomaly detection. This can be done by selecting a cutoff point in the density distribution that separates normal observations from anomalous ones. The threshold can be set based on statistical properties or domain knowledge.
6. Threshold evaluation: The determined thresholds are evaluated using validation techniques, such as cross-validation or hold-out testing, to assess their performance in detecting anomalies accurately. Adjustments to the thresholds may be made based on the evaluation results.

By combining $K-means$ clustering with Kernel Density Estimation, the thresholds obtained are tailored to the underlying density distribution of the data, allowing for more precise anomaly detection. This approach can be useful in various applications, such as fraud detection, outlier detection, or quality control, where distinguishing between normal and abnormal observations is crucial.

K-means [**?**] is one of the simplest unsupervised learning algorithms that solve the well-known clustering problem. The procedure follows a simple and easy method to partition a data set via a number of classes (let's assume k classes) fixed a priori, where each parti- tion represents a cluster containing at least one object. The objective of

k-means is to minimize the objective function which is the total distance between all objects and their respective centres. These centres have to be placed in a clever way because a different location leads to a different result. The objective function is given as follows:

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \left\| x_i^j - c_j \right\|^2 \tag{3.13}$$

Where $\left\| x_i^j - c_j \right\|^2$ is a chosen distance measure between a data point x_i^j and the cluster centre c_j , is an indicator of the distance of the n data points from their respective cluster centers.

Adaptatif threshold using Fuzzy C-means (FCM) based on Kernel Density Estimation (KDE) (UCLFCM)

Adaptive thresholding using FCM based on Kernel Density Estimation (KDE)[**?**] can also be applied in industrial diagnosis and quality control scenarios. Here's an outline of how the technique can be utilized in such contexts:

1. Data Acquisition: Collect data from industrial processes or machinery that is relevant to the diagnostic task. This could include sensor readings, measurements, or images captured from machines or production lines.
2. Preprocessing: Preprocess the data to remove noise, outliers, or any artifacts that could affect the analysis. This step may involve data cleaning, normalization, or filtering techniques specific to the type of data being processed.
3. Feature Extraction: Extract relevant features from the preprocessed data. These features could be statistical measures, time-domain or frequency-domain characteristics, or any other domain-specific features that are informative for the diagnostic task at hand.
4. Kernel Density Estimation (KDE): Apply KDE on the extracted features to estimate the probability density function of the feature space. KDE captures the underlying distribution of the features, enabling a better understanding of their variations and relationships.
5. Fuzzy C-means (FCM) Clustering: Utilize FCM clustering on the KDE-estimated density function to group similar instances together. FCM assigns membership

values to each instance, indicating the degree to which it belongs to each cluster. Clustering helps in identifying different patterns or classes within the data.

6. Adaptive Thresholding: Determine an appropriate threshold to separate the different clusters obtained from FCM. This can be done using different approaches, such as maximizing inter-cluster separation, minimizing intra-cluster variance, or incorporating domain knowledge and specifications.
7. Diagnostic Decision: Make diagnostic decisions based on the assigned cluster memberships or the thresholded results. This may involve identifying faulty or anomalous instances, detecting deviations from expected behavior, or predicting failure probabilities.
8. Intervention and Maintenance: Take appropriate actions based on the diagnostic outcomes. This could involve maintenance actions, equipment calibration, process adjustments, or any necessary interventions to ensure quality control and optimize operational efficiency.
9. Evaluation and Improvement: Assess the performance of the diagnostic system by monitoring the outcomes of the interventions and validating the results. Continuously refine the system based on feedback, collected data, and domain expertise to improve diagnostic accuracy and optimize decision-making.

It's important to note that the success of this approach in industrial diagnosis depends on various factors, including the quality and representativeness of the training data, the choice of features and clustering parameters, the availability of expert knowledge, and the integration of the diagnostic system with the overall industrial processes. Collaboration with domain experts, data scientists, and industry professionals is crucial for building effective diagnostic systems in industrial settings.

Fuzzy C-means (FCM) clustering [**?** **?**] is a popular algorithm used for clustering data based on soft assignments, where each data point can belong to multiple clusters to varying degrees. FCM extends the traditional crisp or hard clustering methods, such as K-means, by allowing overlapping cluster assignments. It is particularly useful when there is ambiguity or uncertainty in the cluster membership of data points. It is based on the minimization of the following objective function:

$$J_m = \sum_{i=1}^{N} \sum_{j=1}^{C} u_{ij}^{m} \left\| x_i - c_j \right\|^2 \tag{3.14}$$

Where m is is the fuzziness parameter, u_{ij} is the degree of membership of x_i in the cluster j and x_i is the ith of the measured data and c_j is the center of the cluster.

The clustering algorithm that has been used in this paper is the Gustafson-Kessel (GK)[**?**].

3.4.3 Fault identification

Fault identification is the process of detecting and recognizing faults or abnormalities in industrial systems. It involves analyzing data collected from sensors, equipment, or other sources to identify deviations from expected behavior. Fault identification methods can include statistical analysis, machine learning algorithms, rule-based approaches, or expert knowledge. The goal is to detect and flag instances that indicate the presence of a fault or anomaly within the system.

Contribution Plots

There are various approaches of fault isolation, for which contribution diagrams can be used. The contribution of variable j to the Q statistic is calculated as follows:

$$C_{ijk}^{Q} = e_{ijk}^{2} \tag{3.15}$$

Where $e = (x_i - \hat{x}_i)$.

Nonlinear Reconstruction Principle

The nonlinear reconstruction principle can also be applied to fault identification in industrial systems. It provides a framework for detecting and diagnosing faults based on lower-dimensional measurements or observations. Here's how the nonlinear reconstruction principle can be utilized for fault identification:

1. Data Collection: Collect data from sensors, equipment logs, or other sources that provide measurements related to the industrial system's behavior. This data represents the system's operating conditions and can include sensor readings, process variables, or performance metrics.

2. Dimensionality Reduction: Apply dimensionality reduction techniques to reduce the dimensionality of the collected data. This step is crucial for dealing with high-dimensional data and extracting essential features while removing noise or irrelevant information. Nonlinear dimensionality reduction methods, such as manifold learning algorithms or autoencoders, can be employed to capture the underlying structure of the data.

3. Nonlinear Embedding: Embed the reduced-dimensional data into a lower-dimensional space using nonlinear embedding techniques. This mapping captures the nonlinear relationships and patterns in the data, facilitating fault detection. The embedded space represents the essential characteristics of the original high-dimensional data in a more manageable form.

4. Reconstruction: Use inverse mapping or reconstruction techniques to estimate the high-dimensional data from the embedded space. This step involves mapping the lower-dimensional observations back to the original high-dimensional space. By reconstructing the high-dimensional data, the nonlinear reconstruction principle allows for a comprehensive analysis of the system's behavior and the detection of deviations or anomalies associated with faults.

5. Fault Detection: Analyze the reconstructed high-dimensional data to identify deviations or abnormalities that indicate the presence of faults. This can be done using statistical methods, machine learning algorithms, or rule-based approaches. By comparing the reconstructed data with expected behavior or predefined models, deviations that are indicative of faults can be detected.

6. Fault Diagnosis: Once a fault is detected, perform fault diagnosis to determine the root cause and nature of the fault. This step involves analyzing the reconstructed data, correlating it with other relevant information, and applying diagnostic techniques to identify the specific component or subsystem responsible for the fault.

The nonlinear reconstruction principle enhances fault identification by enabling the analysis of high-dimensional data using lower-dimensional representations. It leverages the inherent structure and patterns in the data to detect and diagnose faults accurately. By reducing the dimensionality and capturing nonlinear relationships, it provides a more manageable and informative representation of the data, leading to improved fault detection and diagnosis in industrial systems (Figure 3.10).

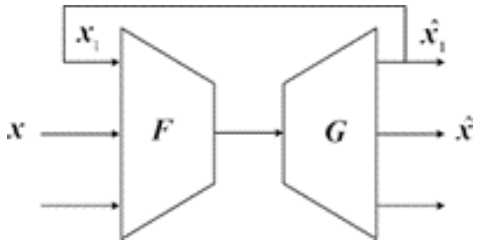

Figure 3.10: Reconstruction Principle

Using the SSAE model [**?**], the approach consists in predicting the measurement $\hat{x}_j$ of the process, by replacing the j^{th} process variable by the predicted one and repeating the operation until conervenge of the algorithm as follow:

$$\tilde{x}_i = \xi_j^T G(F(x_j)) \tag{3.16}$$

Where $\tilde{x}_i = (x_1, x_2, ..., \hat{x}_j, ..., x_l)$, ξ_j^T is the j^{th} column of the identity matrix. And for the actual repeat step i:

$$\hat{x}^{(i)} = G(F(\tilde{x})) \tag{3.17}$$

Who converges if:

$$\left\| \hat{x}_j^{(i+1)} - \hat{x}_j^{(i)} \right\| < \varepsilon \tag{3.18}$$

Where ε is the given threshold.

Sensor Validity Index (SVI)

Sensor Validity Index (SVI) is another technique for identifying the default. it is the measure of sensor performance where standard range should exist regardless of the number of principal components of the disturbances or faults [**?**], it is defined as

follows:

$$\eta_j^2(k) = \frac{SPE_j(k)}{SPE(k)} \tag{3.19}$$

Where SPE is the quadratic global prediction error computed before reconstruction and SPE_j is the jth quadratic prediction error computed after reconstruction [**?**]. The validity index of a faulty sensor must converge towards zero.

3.5 Conclusion

In conclusion, Chapter 3 has examined the contribution of statistical control methods for nonlinear processes in the context of diagnosing and managing dynamic systems. We have explored the advantages and limitations of these approaches in maintaining stability and optimizing the performance of such systems. It is clear that statistical control techniques provide valuable tools for detecting and addressing deviations from normal operating conditions in nonlinear processes.
Building upon the insights gained from Chapters 2 and 3, Chapter 4 will present a real-world application of the Stacked Sparse Autoencoders (SSAE) method in a drinking water treatment system. This chapter will focus on the practical implementation of the SSAE-based approach and will provide detailed explanations of the fault detection and localization methods employed. By leveraging the power of SSAE and combining it with the knowledge gained from the previous chapters, we aim to develop an effective diagnostic framework for the water treatment system.

Chapter 4

EXPERIMENTAL RESULTS AND DISCUSSION

4.1 Introduction

The previous chapter extensively explored various diagnostic and control techniques for dynamic systems, with a specific focus on their application in water treatment processes. Traditional methods such as Principal Component Analysis (PCA), Sparse PCA, and the innovative approach of Stacked Sparse Autoencoders (SSAE) were examined in detail for defect detection and localization.

In this chapter, we will transition from theory to practice by applying these methods to both synthetic data and a real water treatment plant system. Our objective is to analyze data collected from appropriate sensors within the water treatment system to detect potential defects and locate their sources.

We will start by introducing the chosen water treatment plant system as a case study, providing an overview of its components, key features, and performance objectives. We will then discuss the data collection and preprocessing steps, with an emphasis on selecting relevant variables and preparing the data for analysis.

Next, we will delve into the application of diagnostic methods such as PCA, Sparse PCA, and SSAE to the water treatment system data. We will detail the steps involved in the analysis, including model construction, feature learning, and the use of defect detection algorithms.

Finally, we will evaluate the performance of the defect detection and localization methods on both the synthetic data and the real water treatment plant system. We will

analyze the obtained results, discuss the advantages and limitations of each method, and provide recommendations for enhancing the diagnostic efficiency and system maintenance.

This chapter serves as a crucial step in demonstrating the applicability of diagnostic and control techniques for dynamic systems in a real-world context. The application on a water treatment plant system will showcase the effectiveness and utility of these methods in ensuring reliability, quality, and performance in water treatment processes. Additionally, the synthetic study will provide insights into the performance of these methods in controlled settings. These examples will help evaluate the effectiveness of the techniques used, identify potential improvements, and guide decisions regarding maintenance and optimization of water treatment systems.

4.2 Application on synthetic data

To demonstrate the methods and illustrate their benefits in fault detection and isolation, we use multivariate dataset containing three variables where t is uniformly distributed in the interval $[-1,1]$; ε_i denotes the Gaussian white noises with zero means and standard deviation of 0.01 and samples are collected to build PCA, SPAC and SSAE models .

$$\begin{aligned} x_1 &= t^2 + 0.3\sin(2\pi t) + \varepsilon_1 \\ x_2 &= t + \varepsilon_2 \\ x_3 &= t^3 + t + 1 + \varepsilon_3 \end{aligned} \tag{4.1}$$

After creating the models, we check the evolution of SPE under normal conditions, where its threshold was calculated with 4 techniques: the χ^2 distribution as shown in figures: 4.1, 4.2 and 4.2, Kernel Density Estimation (KDE) as shown in firues: 4.4, 4.5 and 4.6, the adaptive threshold $AUCL_{KDE}$ using K-means clustering as shown in figures: 4.7, 4.8 and 4.9 and the adaptive threshold $AUCL_{KDE}$ using Fuzzy C-means is shown in figures: 4.10, 4.11 and 4.12.

By examining the figures 4.1, 4.2, 4.2, 4.4, 4.5, 4.6, 4.7, 4.8, 4.9, 4.10, 4.11 and 4.12, we note that the best result is always obtained by the SSAE model.
In order to assess the effectiveness of our models in fault detection, we will simulate a fault occurring at one of the sensor levels. For PCA and Sparse PCA, the fault was

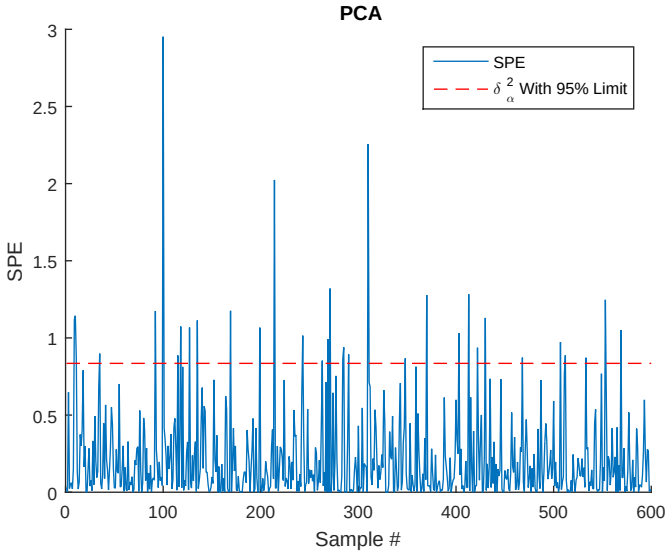

Figure 4.1: SPE: data in normal state for PCA δ_α^2 threshold

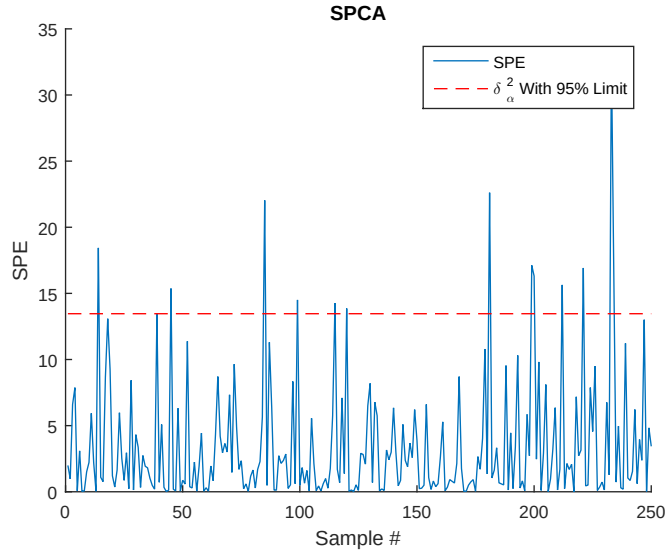

Figure 4.2: SPE: data in normal state for SPCA with δ_α^2 threshold

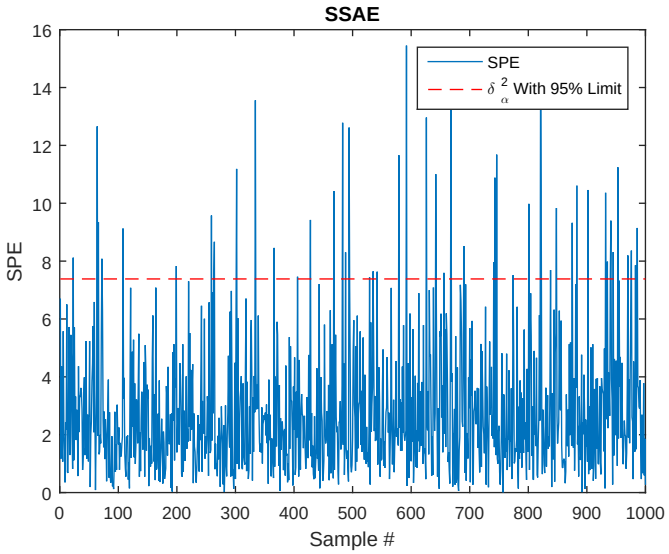

Figure 4.3: SPE: data in normal state for SSAE with δ_α^2 threshold

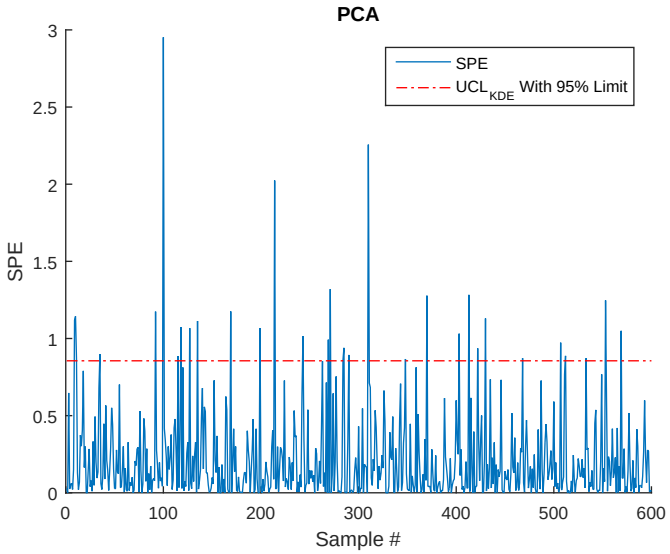

Figure 4.4: SPE: data in normal state for PCA with KDE threshold

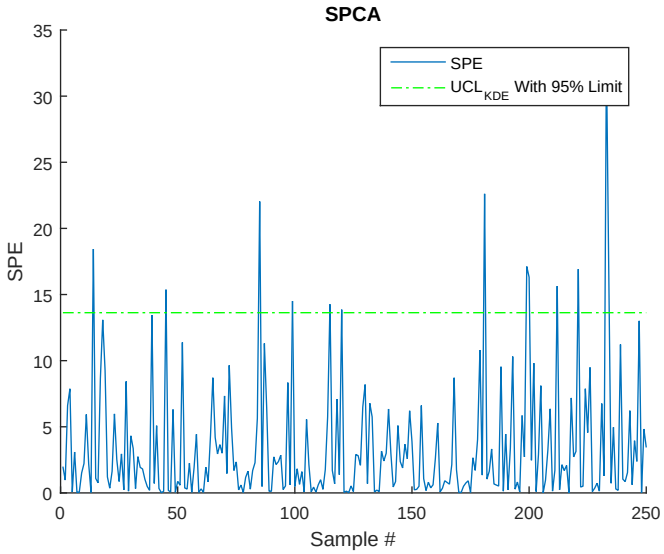

Figure 4.5: SPE: data in normal state for SPCA with KDE threshold

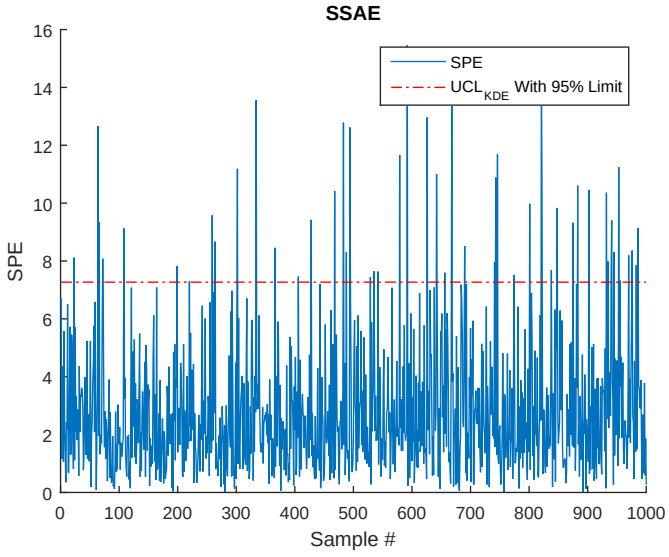

Figure 4.6: SPE: data in normal state for PCA, SPCA and SSAE with KDE threshold

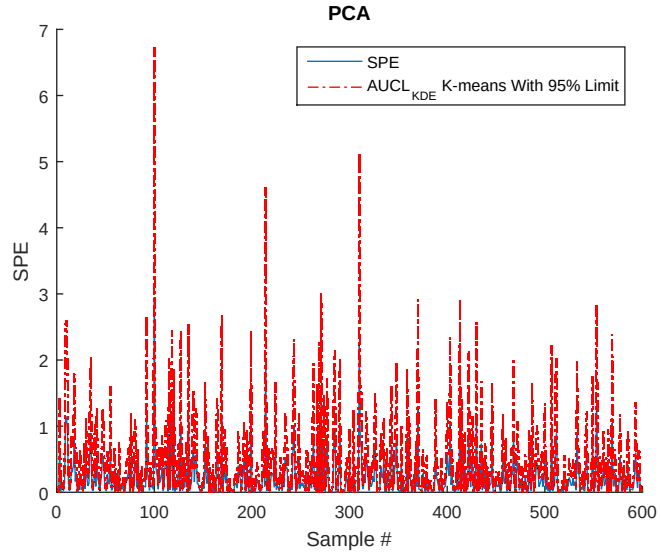

Figure 4.7: SPE: data in normal state for PCA with AUCL threshold using K-means clustering

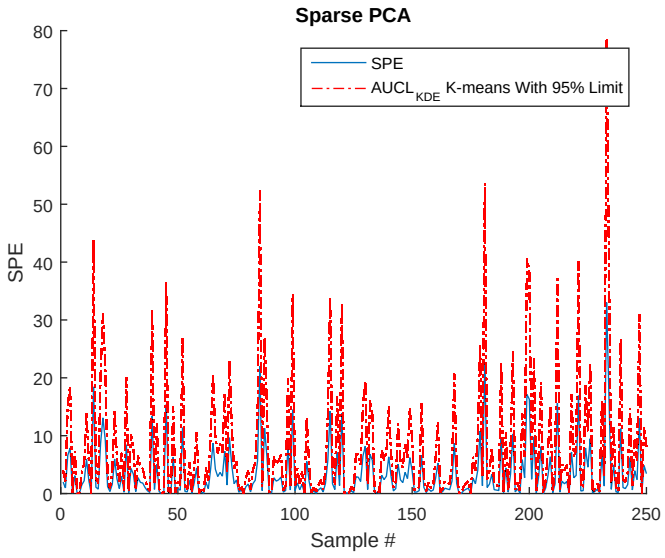

Figure 4.8: SPE: data in normal state for SPCA with AUCL threshold using K-means clustering

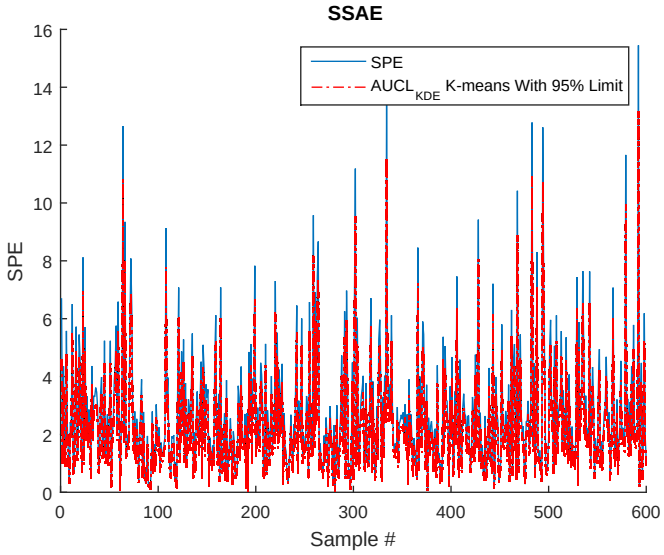

Figure 4.9: SPE: data in normal state for SSAE with AUCL threshold using K-means clustering

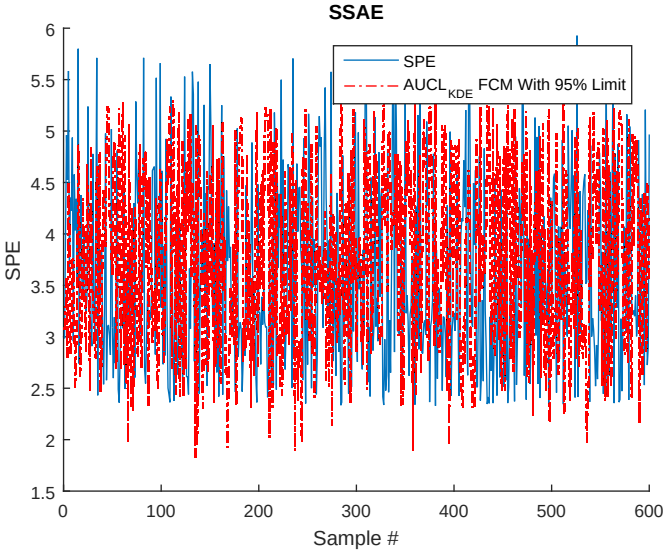

Figure 4.10: SPE: data in normal state for PCA with AUCL threshold using FCM

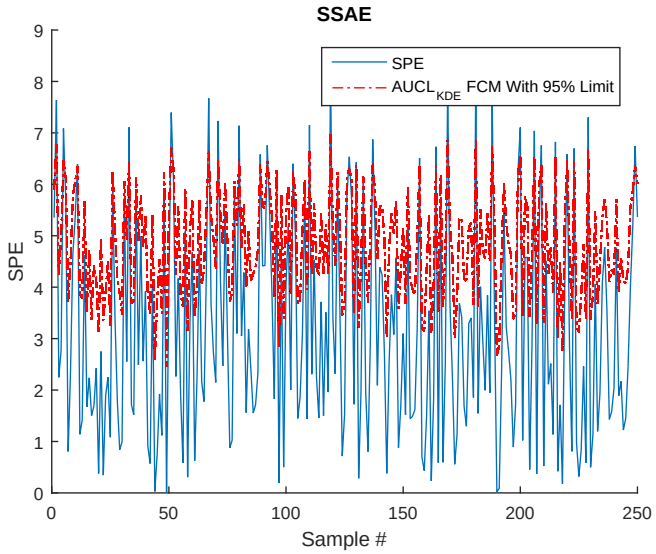

Figure 4.11: SPE: data in normal state for SPCA with AUCL threshold using FCM

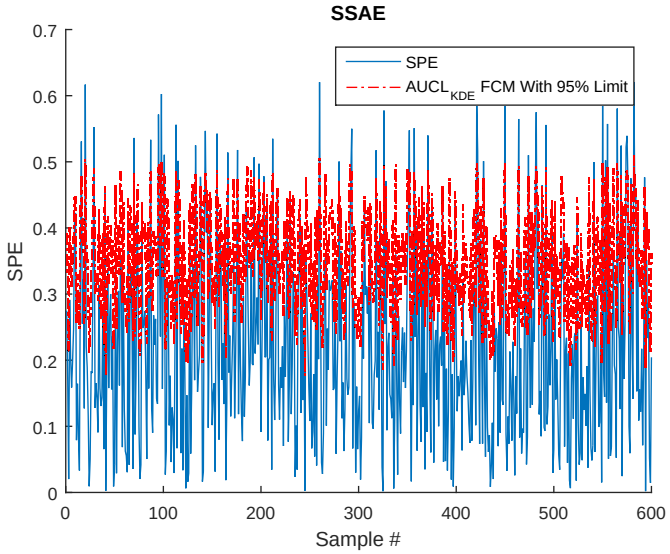

Figure 4.12: SPE: data in normal state for SSAE with AUCL threshold using FCM

injected at sample 350, while for the SSAE model, the fault was injected at sample 150.
We notice the evolution of SPE with the 4 thresholds as it is illustrated in figures 4.13, 4.14, 4.15, 4.16, 4.18, 4.18,4.19, 4.20, 4.21, 4.22, 4.23 and 4.24.

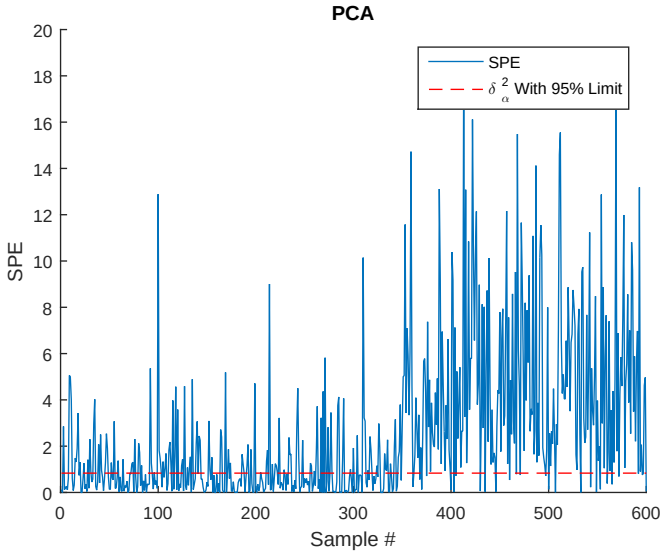

Figure 4.13: SPE: data in faulty state for PCA with δ^2_α threshold

Upon analyzing the figures in both normal and faulty states, it becomes evident that there are instances of false alarms in our data, primarily attributed to outlier measurements.

However, in our specific case, the SSAE model consistently demonstrates reliable performance by accurately detecting faults. Notably, it achieves fault detection at sample 150 consistently, ensuring consistent positive results.

Additionally, the models are capable of identifying the specific faulty sensor, as demonstrated in Figure 4.25 for the PCA model, Figure 4.26 for the Sparse PCA (SPCA) model, and Figure 4.27 for the SSAE model. In this case, it was determined that the third sensor was the one exhibiting the fault.

Furthermore, it is worth mentioning that there is another method available for localizing the faulty sensor using SVI (Sensor Validation and Identification). This method guides the process towards restoring normal operating conditions, as depicted in Figure **??**.

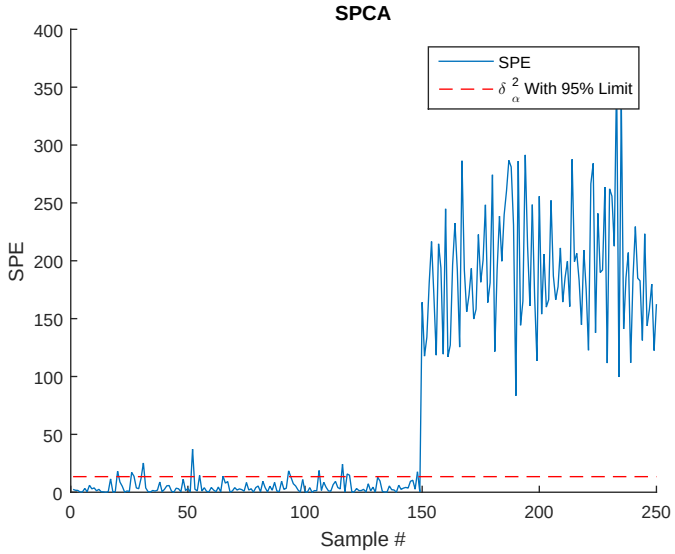

Figure 4.14: SPE: data in faulty state for SPCA with δ_α^2 threshold

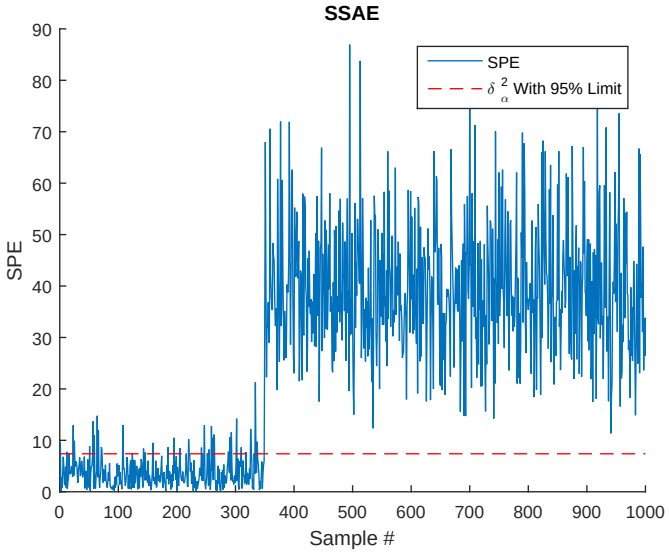

Figure 4.15: SPE: data in faulty state for SSAE with δ_α^2 threshold

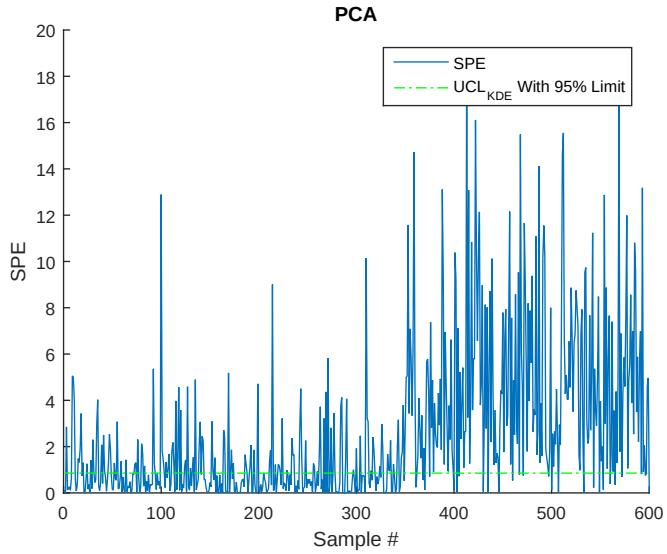

Figure 4.16: SPE: data in faulty state for PCA with KDE threshold

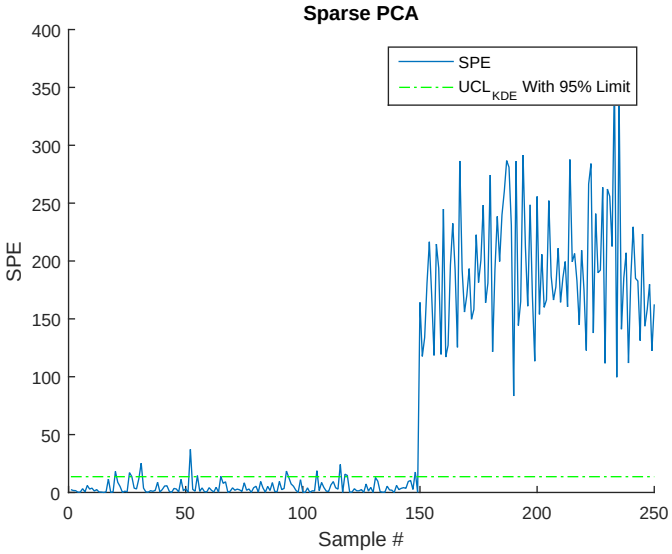

Figure 4.17: SPE: data in faulty state for SPCA with KDE threshold

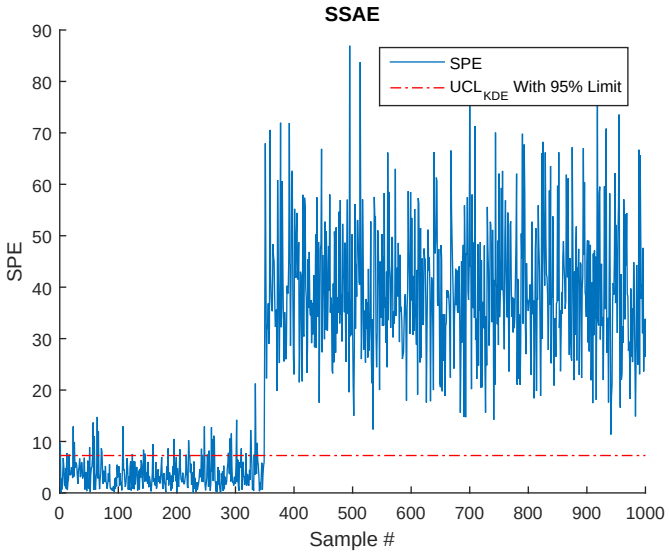

Figure 4.18: SPE: data in faulty state for SSAE with KDE threshold

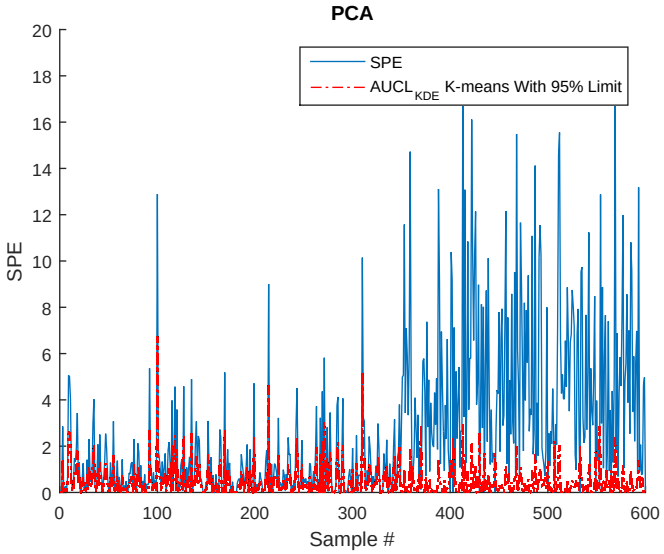

Figure 4.19: SPE: data in faulty state for PCA with AUCL threshold using K-means clustering

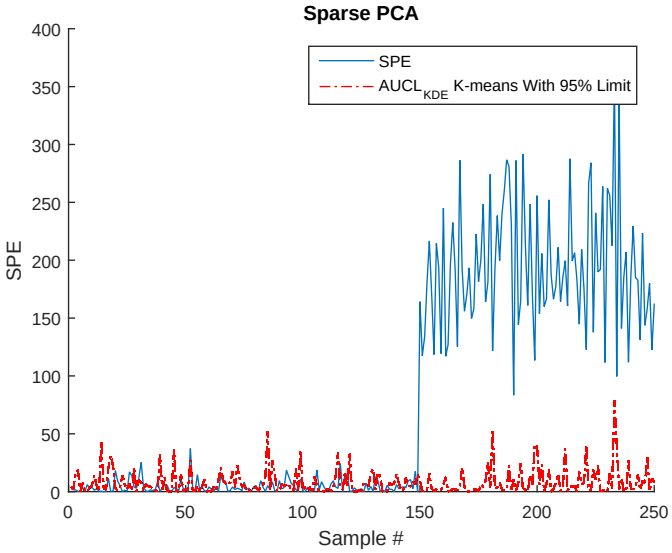

Figure 4.20: SPE: data in faulty state for SPCA with AUCL threshold using K-means clustering

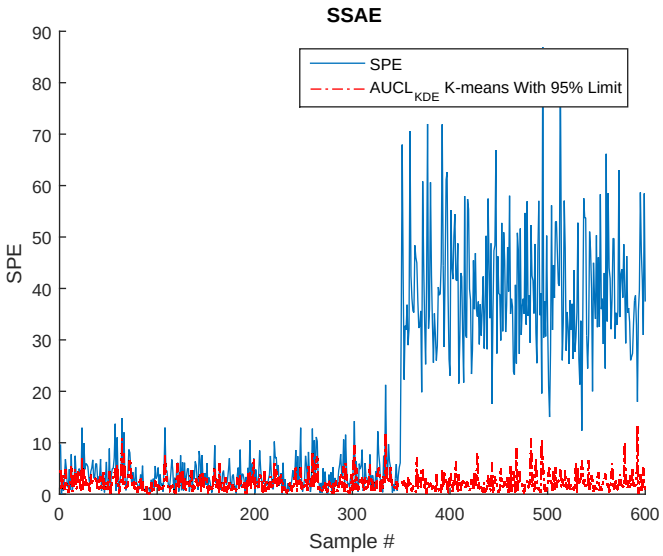

Figure 4.21: SPE: data in faulty state for SSAE with AUCL threshold using K-means clustering

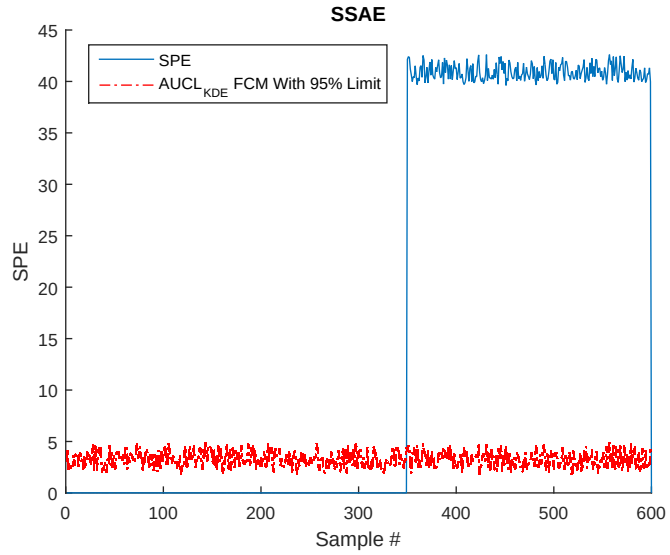

Figure 4.22: SPE:data in faulty state for PCA with AUCL threshold using FCM

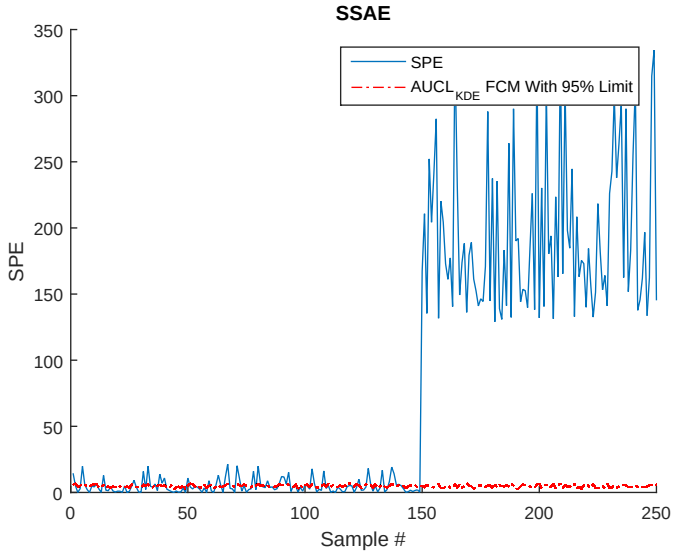

Figure 4.23: SPE:data in faulty state for SPCA with AUCL threshold using FCM

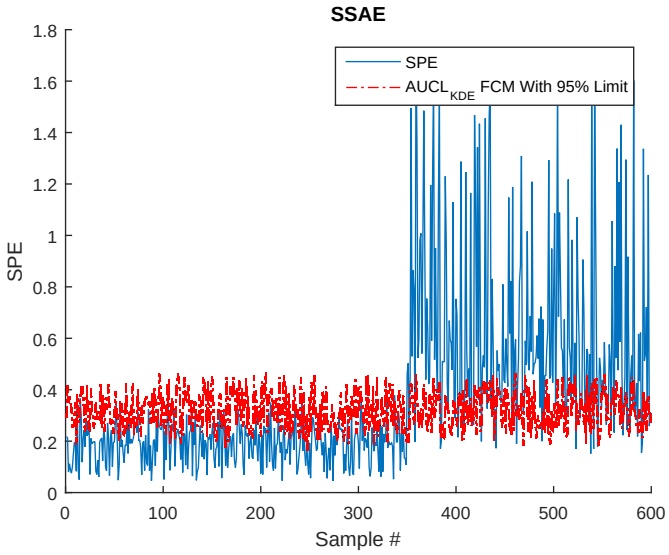

Figure 4.24: SPE:data in faulty state for SSAE with AUCL threshold using FCM

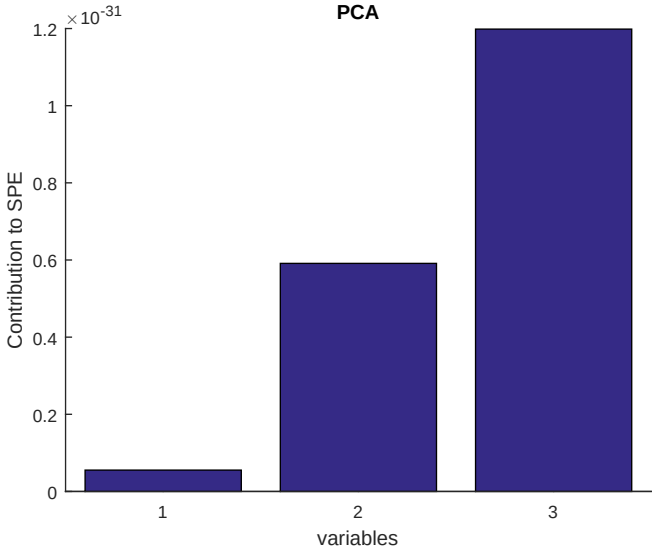

Figure 4.25: Fault isolation using normalized contribution plots for PCA (fault in the 3rd sensor)

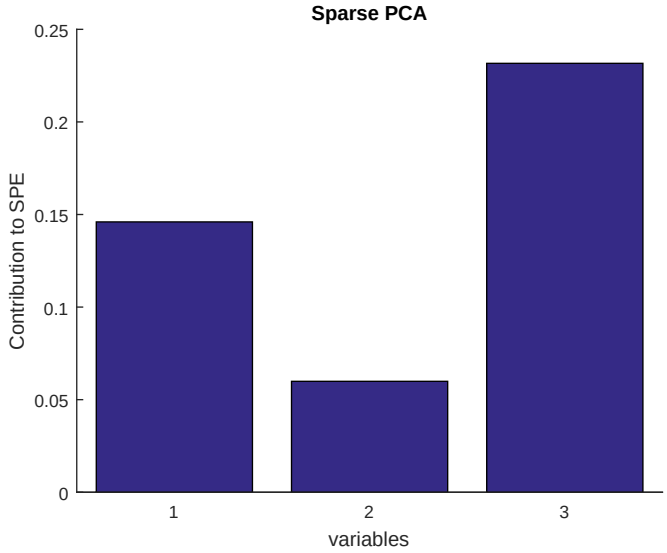

Figure 4.26: Fault isolation using normalized contribution plots for SPCA (fault in the 3rd sensor)

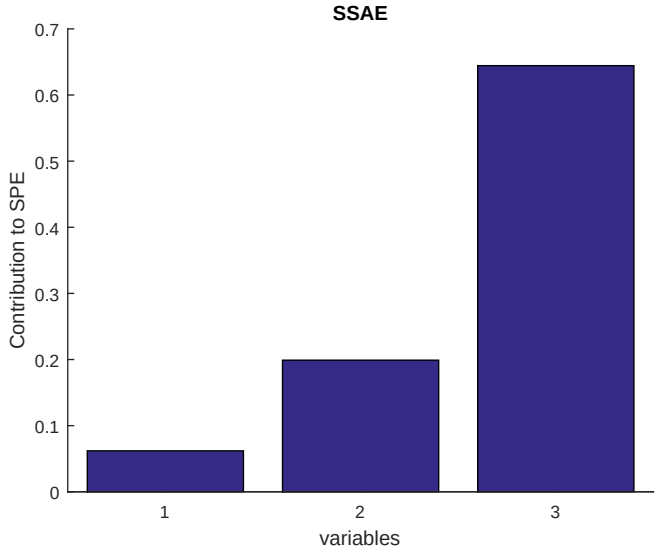

Figure 4.27: Fault isolation using normalized contribution plots for SSAE (fault in the 3rd sensor)

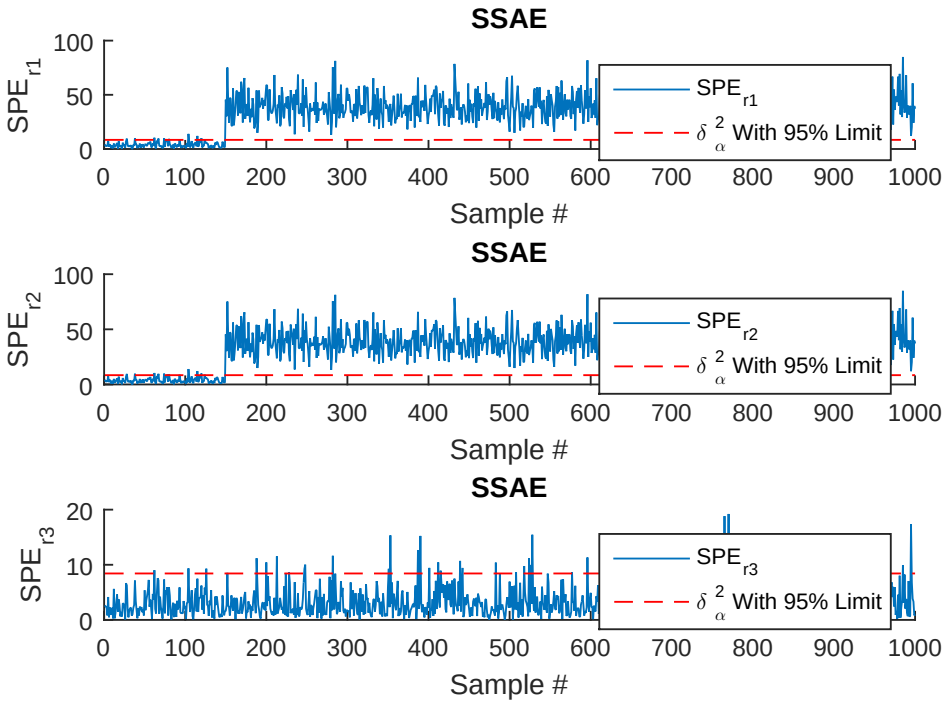

Figure 4.28: Fault isolation using reconsctruction principle for SSAE using δ_α^2 (fault in the 3rd sensor)

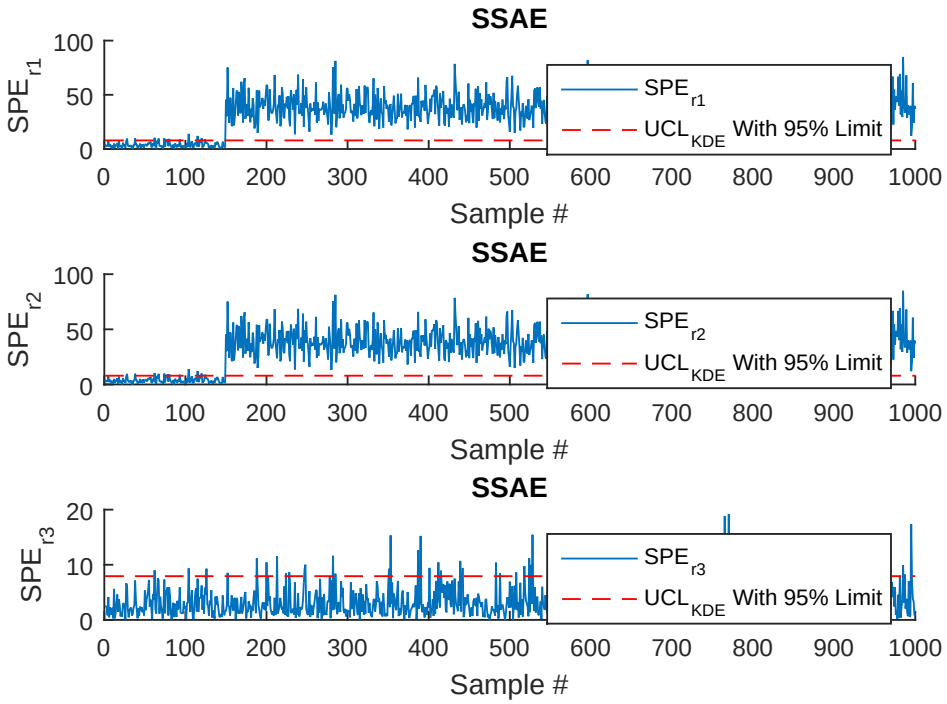

Figure 4.29: Fault isolation using reconsctruction principle for SSAE using KDE(fault in the 3rd sensor)

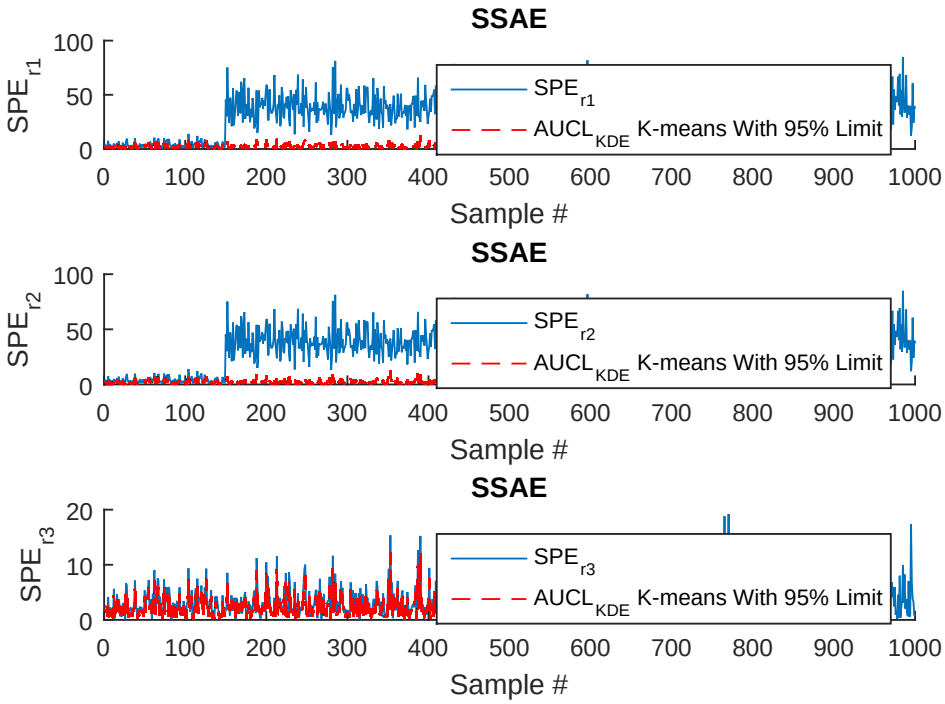

Figure 4.30: Fault isolation using reconsctruction principle for SSAE using AUCL Kmeans(fault in the 3rd sensor)

It is important to note that the results obtained from this method were specific to the SSAE network only.

In conclusion, in this synthetic application, we have explored the use of three models: PCA, Sparse PCA, and SSAE, for fault detection and localization. We have utilized static and adaptive thresholding methods, as well as the SVI method and contribution plots for sensor localization. Additionally, we have observed that the non-linear reconstruction performed by the SSAE model has shown promising results.

The PCA and Sparse PCA models were effective in fault detection but also generated some false alarms due to outlier measurements. On the other hand, the SSAE model demonstrated consistent performance in detecting faults and accurately identifying faulty sensors.

The SVI method proved to be a useful approach for localizing the faulty sensor and restoring the process to normal operating conditions. Furthermore, the use of contribution plots provided additional insights into the importance of sensors in fault detection.

Lastly, the non-linear reconstruction performed by the SSAE model yielded encouraging results, suggesting that the model is capable of effectively capturing underlying fault patterns.

Overall, this synthetic application has demonstrated the effectiveness of PCA, Sparse PCA, and SSAE models in fault detection and localization. The various approaches used, such as static and adaptive thresholds, SVI, and contribution plots, contributed to a better understanding of faults and more accurate decision-making in sensor localization.

4.3 Case study: Application to drinking Water Treatment Plant

A water treatment plant is a facility designed to purify and treat water from various sources to make it safe for human consumption and other purposes. The main objective of a water treatment plant is to remove impurities, contaminants, and harmful substances from raw water, ensuring that the treated water meets the required quality standards.

The process of water treatment in a water treatment plant typically involves several stages, each designed to target specific contaminants and achieve the desired level of water quality. These stages may include:

1. Coagulation and Flocculation: Chemicals are added to the water to facilitate the clumping together of particles and impurities, forming larger particles called flocs.
2. Sedimentation: The water is allowed to settle, and the flocs, along with heavier impurities, settle to the bottom of a sedimentation basin or clarifier, forming a sludge layer.
3. Filtration: The water passes through various filtration systems, such as sand filters or activated carbon filters, to remove smaller suspended particles and remaining impurities.
4. Disinfection: Chemical disinfectants, such as chlorine or ozone, are added to kill harmful bacteria, viruses, and other microorganisms present in the water.
5. pH Adjustment: The pH level of the water is adjusted to a suitable range to prevent corrosion and ensure its compatibility with distribution systems.
6. Advanced Treatment: Depending on the specific water source and quality requirements, additional treatment processes may be employed, such as membrane filtration, reverse osmosis, or ultraviolet (UV) disinfection.

The operation of a water treatment plant involves the careful monitoring and control of various parameters such as flow rate, pressure, turbidity, pH, disinfectant levels, and water quality indicators. This is typically done using automated control systems and continuous monitoring equipment to ensure consistent and reliable treatment processes.

The application of diagnostic techniques in a water treatment plant is crucial for identifying and addressing any issues or malfunctions that may occur. Diagnostic methods involve the analysis of operational data, sensor readings, and performance indicators to detect abnormalities, identify potential equipment failures, optimize process efficiency, and ensure compliance with regulatory standards.

Some common diagnostic techniques used in water treatment plants include statistical analysis, trend monitoring, data-driven modeling, and fault detection algorithms. These methods help plant operators and maintenance personnel to identify and diagnose problems, initiate corrective actions, and minimize downtime and operational disruptions.

We will focus on the practical application of diagnostic and control techniques to a specific water treatment plant, namely the Oued El Othmania water purification plant. Located in Constantine, a city in northeast Algeria, this plant is responsible for supplying drinking water to a large number of citizens in the surrounding area. Constantine itself is the third most populous city in the country.

The Oued El Othmania water treatment plant is known for its critical role in ensuring the distribution of safe and clean drinking water to the local population. It deals with several important parameters such as turbidity, temperature, flow rate, and pH value, among others. These parameters are crucial for maintaining the quality and safety of the treated water.

For our study, we have collected and analyzed data that includes both raw water parameters and treated water parameters. Specifically, we focus on the parameters of turbidity, temperature, pH, and oxygen levels, resulting in a total of eight monitored parameters. The data acquisition unit, using a SCADA system, has sampled these observations over a period of 356 days, considering various time intervals.

Our objective is to utilize this dataset to construct and develop a Stacked Sparse Autoencoders (SSAE) model for the water treatment plant. By applying the SSAE method to the collected data, we aim to uncover meaningful patterns, detect anomalies, and contribute to the overall diagnostic capabilities of the plant.

Through the analysis and application of the SSAE model, we seek to enhance the plant's ability to monitor and control the water treatment processes effectively. By leveraging the insights gained from the developed model, the plant can detect deviations from normal operation, localize potential faults, and take proactive measures to ensure the delivery of high-quality drinking water.

By focusing on the specific case of the Oued El Othmania water purification plant, this chapter provides a practical demonstration of the diagnostic and control techniques in the context of a real-world water treatment facility. The results obtained from this application will contribute to improving the reliability, efficiency, and maintenance practices of the plant, ultimately benefiting the local population's access to safe drinking water.

Now, let us delve into the detailed analysis and application of the Stacked Sparse Autoencoders (SSAE) model using the collected dataset from the Oued El Othmania water treatment plant.

The dataset utilized in this study comprises measurements of raw water and treated water parameters, including Turbidity, Temperature, pH, and O_2. Therefore, a total

of eight parameters are monitored, implying the presence of eight sensors. The data is sampled by a data acquisition unit (SCADA system) over a span of 356 days, encompassing different periods. This database serves as the foundation for constructing and developing the PCA, SPCA, and SSAE models.

For the assessment of normal conditions using three types of thresholds, we examine the Squared Prediction Error (SPE) on the actual data. The results are presented for SPCA and SSAE in figures 4.33, 4.34, 4.35, 4.41, 4.37, 4.38, 4.39 and 4.40. As depicted in the previous figures, the SSAE model demonstrates a favorable outcome, indicating successful fault detection. Conversely, no significant results were obtained for PCA.

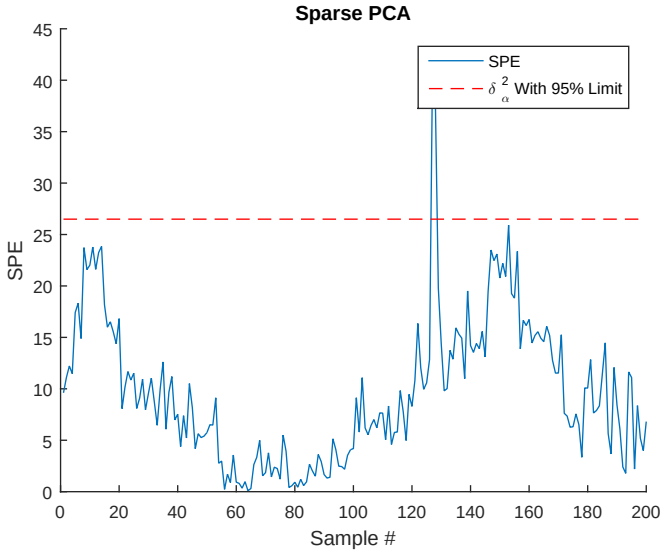

Figure 4.33: SPE: data in normal state for SPCA with δ_α^2 threshold

The SSAE model exhibits the capability to detect and isolate faults, as evidenced by the detection accuracy depicted in figures 4.41, 4.42, 4.43, 4.44, 4.45, 4.46, 4.47 and 4.48. In our specific case, the fault was detected starting from sample 120, and the faulty sensor was identified as the eighth sensor, which corresponds to the O_2 Treated Water (TW) parameter, as shown in figure 4.50 for the SSAE network. Conversely, for the SPCA model, a false result was obtained due to the fault being injected at the eighth sensor instead of the seventh sensor, as illustrated in figure 4.49.

Furthermore, we employed the SVI (Sensor Validation Index) method for fault isolation. The results of this method are presented in figure **??** specifically for the SSAE network, as it was the only method that yielded conclusive results in this regard.

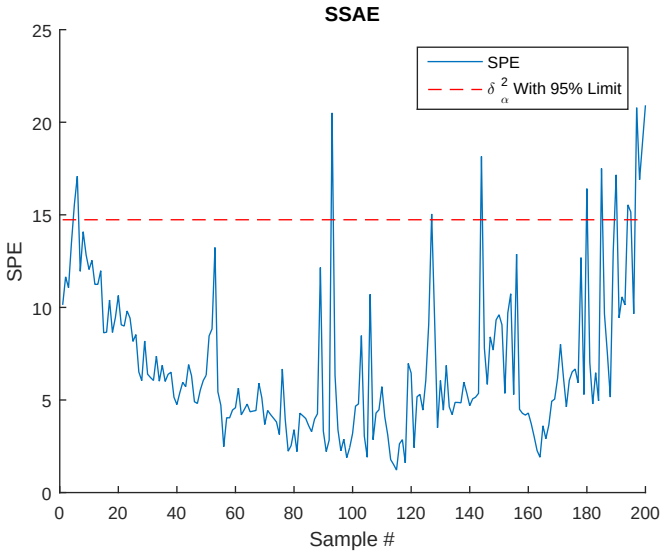

Figure 4.34: SPE: data in normal state for SSAE with δ_α^2 threshold

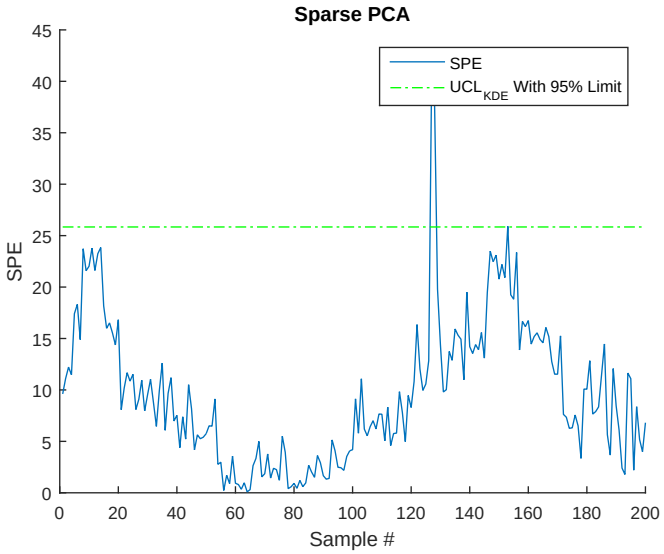

Figure 4.35: SPE: data in normal state for SPCA with KDE threshold

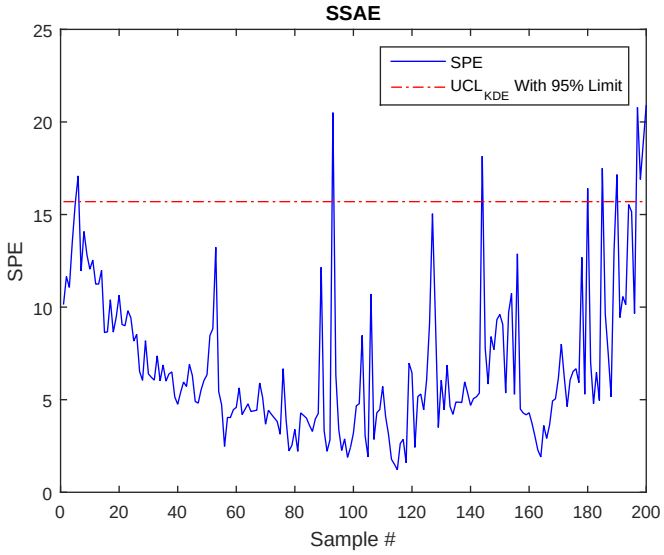

Figure 4.36: SPE: data in normal state for SSAE with KDE threshold

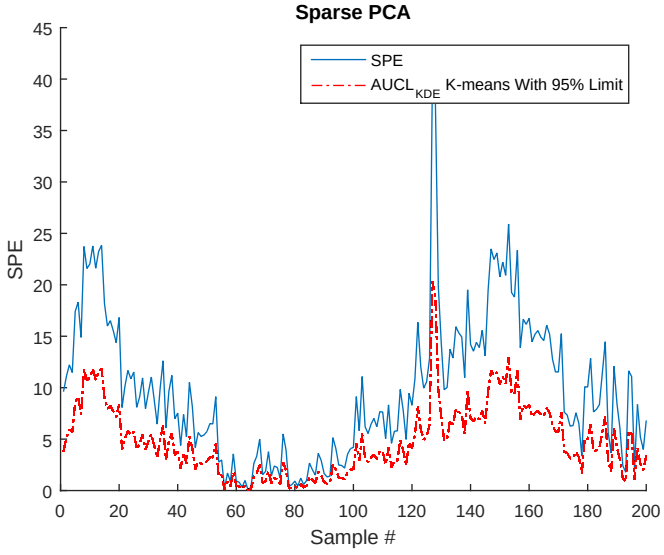

Figure 4.37: SPE: data in normal state for SPCA with AUCL threshold using K-means clustering

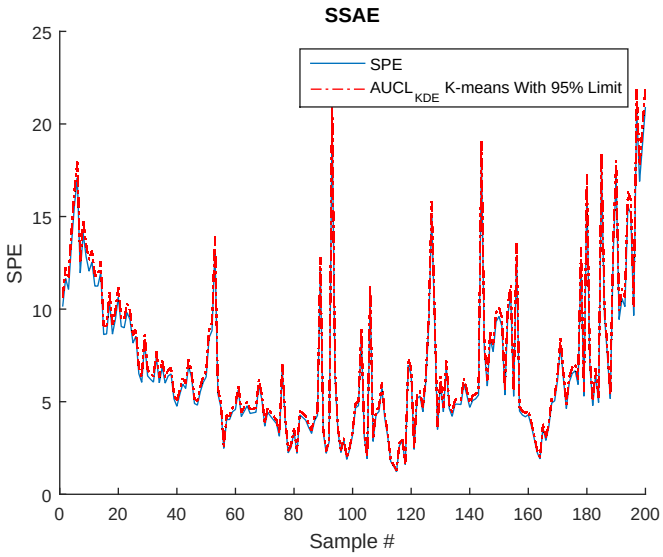

Figure 4.38: SPE: data in normal state for SSAE with AUCL threshold using K-means clustering

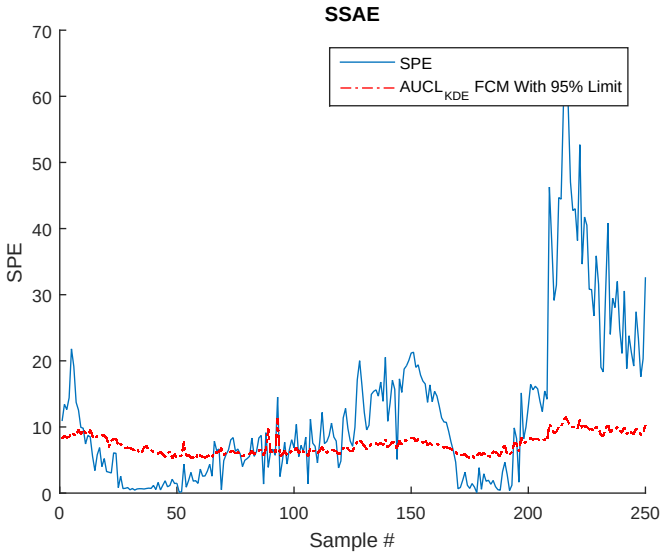

Figure 4.39: SPE: data in normal state for SPCA with AUCL threshold using FCM

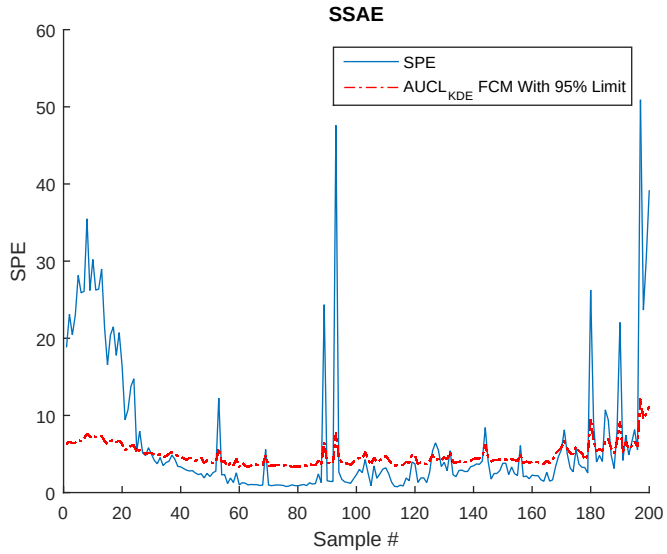

Figure 4.40: SPE: data in normal state for SSAE with AUCL threshold using FCM

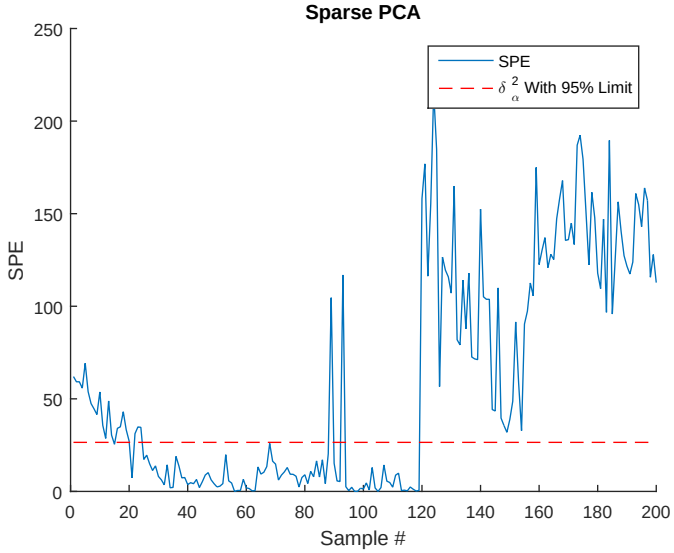

Figure 4.41: SPE: data in faulty state for SPCA with δ_α^2 threshold

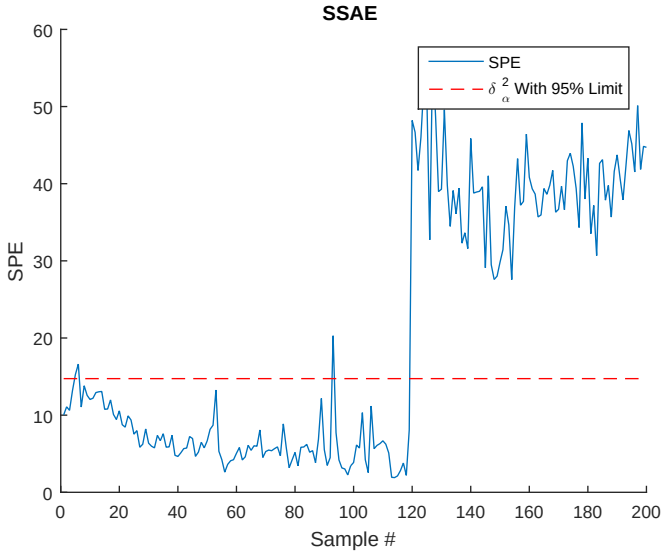

Figure 4.42: SPE: data in faulty state for SSAE with δ_α^2 threshold

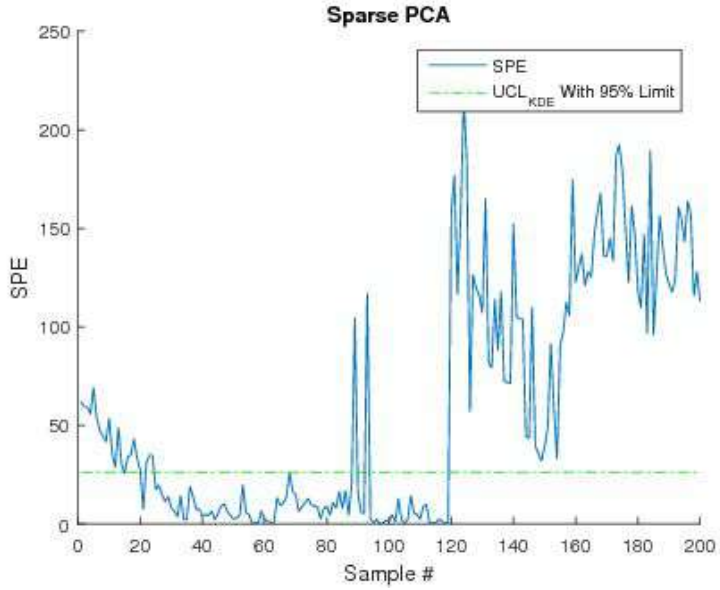

Figure 4.43: SPE: data in faulty state for SPCA with KDE threshold

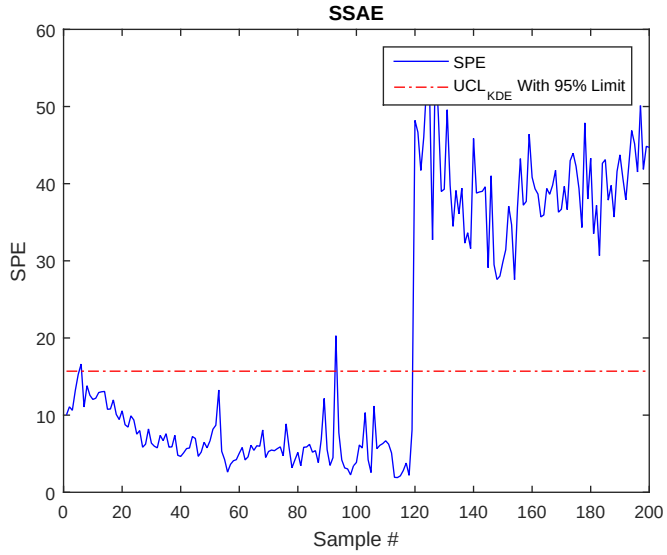

Figure 4.44: SPE: data in faulty state for SSAE with KDE threshold

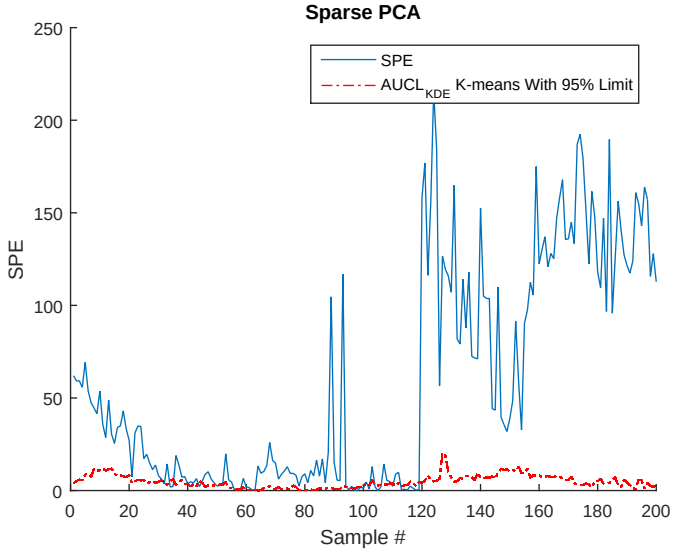

Figure 4.45: SPE: data in normal state for SPCA with AUCL threshold using K-means clustering

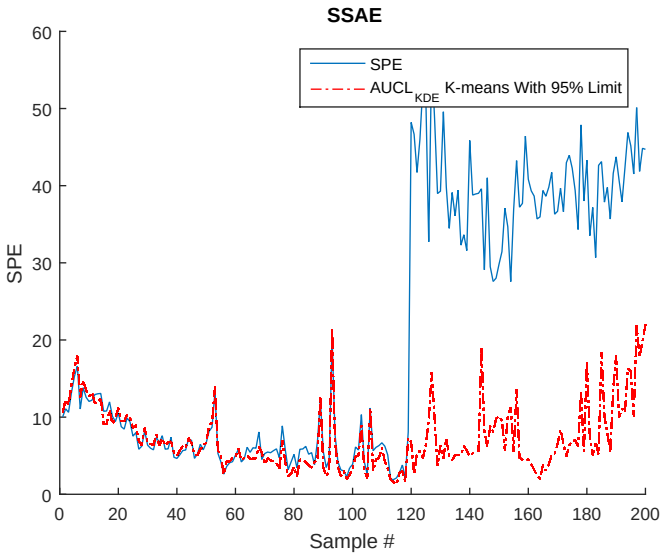

Figure 4.46: SPE: data in normal state for SSAE with AUCL threshold using K-means clustering

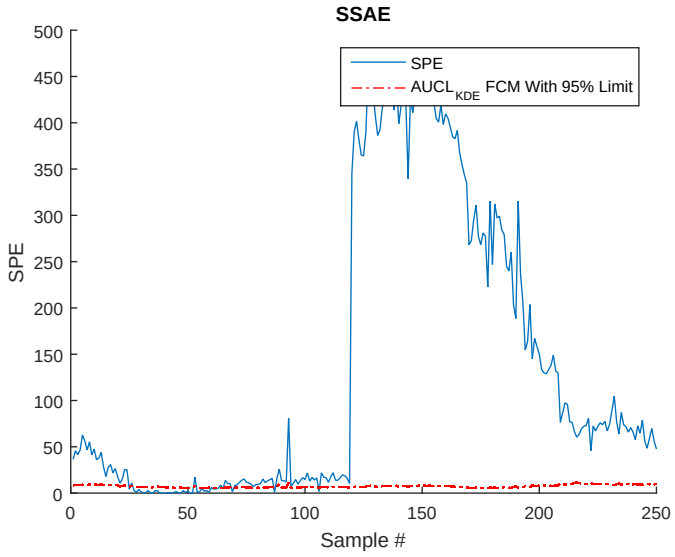

Figure 4.47: SPE: data in normal state for SPCA with AUCL threshold using FCM

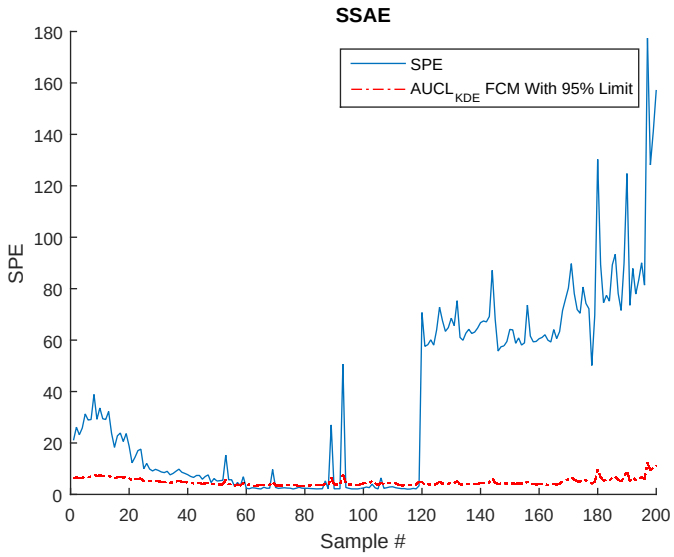

Figure 4.48: SPE: data in normal state for SSAE with AUCL threshold using FCM

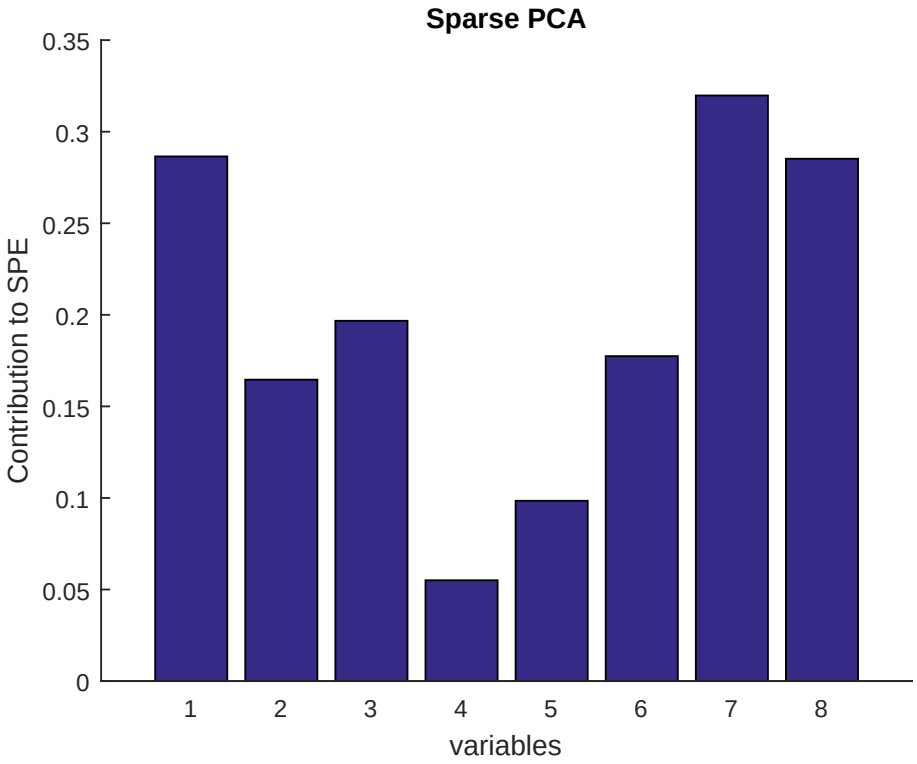

Figure 4.49: Fault identification using normalized contribution plots for SPAC

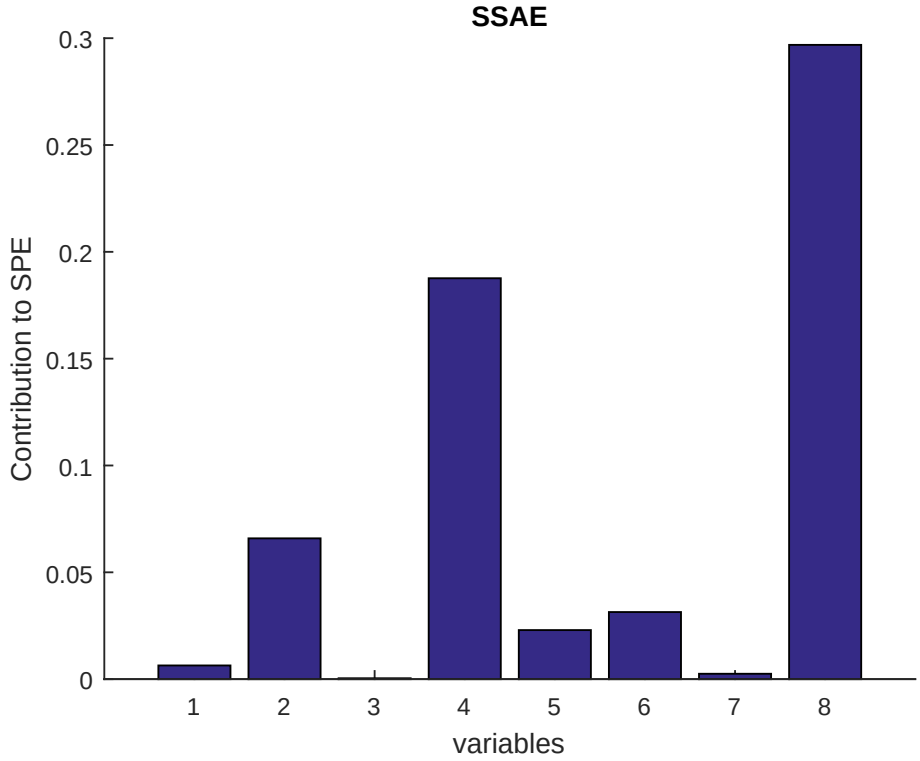

Figure 4.50: Fault identification using normalized contribution plots for SSAE (Fault in the 8 sensor)

4.4 Discussion and Conclusion

Chapter 4 has presented a detailed application of the Stacked Sparse Autoencoders (SSAE) method in a real-world drinking water treatment process. We have explored the practical implementation of the SSAE-based approach and discussed the methods used for fault detection and localization. Through this application, we have demonstrated the effectiveness of the SSAE method in diagnosing and managing dynamic systems, specifically in the context of water treatment.

The application of the SSAE method in the water treatment system has proven to be highly relevant and promising. Water treatment systems play a crucial role in providing safe and quality drinking water. However, these systems are prone to various faults such as leaks, equipment malfunctions, water quality variations, and more. By applying the SSAE-based methodology, we can extract relevant features from the data collected through appropriate sensors. These learned features by the SSAE are then utilized to efficiently detect and localize faults within the water treatment system.

The fault detection and localization methods employed in this application leverage the hierarchical and nonlinear representations learned by the SSAE. These methods include analyzing neuron activations in the hidden layers of the SSAE, comparing extracted features with reference models, utilizing clustering algorithms to group similar data, and more. These approaches offer precise fault detection and efficient localization within the water treatment system.

In conclusion, the utilization of Stacked Sparse Autoencoders (SSAE) presents an innovative and powerful approach for diagnosing and operating dynamic systems. Through the real-world application in a drinking water treatment system, we have showcased the capability of the SSAE method to accurately detect faults and efficiently locate them, contributing to the maintenance and optimization of dynamic system performance.

As we conclude Chapter 4, we have achieved significant progress in understanding the diagnostic and operational aspects of dynamic systems using the SSAE method. This research opens up new possibilities for monitoring and managing complex dynamic systems.

General Conclusion

By focusing on the diagnosis and operation of dynamic systems, an innovative approach based on Stacked Sparse Autoencoders (SSAE) has been used, starting with Principal Component Analysis (PCA) and sparse PCA. The main objective of this research is to develop an effective methodology for the detection and localization of faults in dynamic systems, with a focus on its application to a water treatment system.

Initially, we studied the limitations of PCA and sparse PCA in capturing the nonlinear relationships present in dynamic systems. While these methods are commonly used to reduce the dimensionality of data and extract the most significant information, they are not suitable for capturing complex and nonlinear relationships. This is where SSAEs come into play.

SSAEs are deep artificial neural networks capable of learning hierarchical and nonlinear representations of data. By using SSAEs, we can extract increasingly abstract features from the data, facilitating the detection of meaningful patterns. These features learned by SSAEs are then used for the detection and localization of faults in dynamic systems.

To apply this methodology to the water treatment system, we collected data using appropriate sensors and preprocessed it to eliminate noise and outliers. Then, the SSAEs were trained on this preprocessed data to learn compact and informative representations. Using these representations, we developed system-specific fault detection and localization techniques for the water treatment system.

The results obtained demonstrate that the SSAE-based approach allows for accurate detection and efficient localization of faults in the water treatment system. This methodology offers a promising solution for the monitoring and maintenance of dynamic

systems, enabling early detection of issues and informed decision-making for performance optimization.

In conclusion, we have demonstrated the effectiveness of Stacked Sparse Autoencoders (SSAE) for the diagnosis and operation of dynamic systems, starting with PCA and sparse PCA. The application of this approach to the water treatment system enabled precise fault detection and localization, paving the way for new possibilities in the monitoring and management of complex dynamic systems.

Numerous perspectives for future research are open, such as the development of precise fault localization methods, exploration of more advanced model architectures, integration of data fusion techniques, extension of the application to complex and diverse systems, and the development of real-time diagnostic methods. These objectives will contribute to a better understanding, more efficient maintenance, and performance optimization of dynamic systems.

Bibliography

[1] Sang Bin Lee, Greg C Stone, Jose Antonino-Daviu, Konstantinos N Gyftakis, Elias G Strangas, Pascal Maussion, and Carlos A Platero. Condition monitoring of industrial electric machines: State of the art and future challenges. *IEEE Industrial Electronics Magazine*, 14(4):158–167, 2020.

[2] Ahmed Bouraiou, Ammar Neçaibia, Saad Motahhir, Mohammed Salah Bouakkaz, Issam Attoui, Nadir Boutasseta, Rachid Dabou, and Abderrezzaq Ziane. Supervision and monitoring of photovoltaic systems using siemens plc and hmi. In *Digital Technologies and Applications: Proceedings of ICDTA 21, Fez, Morocco*, pages 1147–1157. Springer, 2021.

[3] Monica Tiboni, Francesco Aggogeri, Nicola Pellegrini, and Cesare Augusto Perani. Smart modular architecture for supervision and monitoring of a 4.0 production plant. *International Journal of Automation Technology*, 13(2):310–318, 2019.

[4] Sandipan Chakraborty, Shouvik Mukherjee, Suvrojit Kumar Saha, and Himadri Nath Saha. Autonomous vehicle for industrial supervision based on google assistant services & iot analytics. In *2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, pages 1061–1070. IEEE, 2019.

[5] Miloš D Djordjević, Jana M Vračar, and Aleksandra S Stojković. Supervision and monitoring system of the power line poles using iiot technology. In *2020 55th International Scientific Conference on Information, Communication and Energy Systems and Technologies (ICEST)*, pages 58–61. IEEE, 2020.

[6] Weihua Li, Ruyi Huang, Jipu Li, Yixiao Liao, Zhuyun Chen, Guolin He, Ruqiang Yan, and Konstantinos Gryllias. A perspective survey on deep transfer learning for fault diagnosis in industrial scenarios: Theories, applications and challenges. *Mechanical Systems and Signal Processing*, 167:108487, 2022.

[7] Shreyas Gawde, Shruti Patil, Satish Kumar, Pooja Kamat, Ketan Kotecha, and Ajith Abraham. Multi-fault diagnosis of industrial rotating machines using data-driven approach: A review of two decades of research. *Engineering Applications of Artificial Intelligence*, 123:106139, 2023.

[8] Yalin Wang, Dongzhe Wu, and Xiaofeng Yuan. Lda-based deep transfer learning for fault diagnosis in industrial chemical processes. *Computers & Chemical Engineering*, 140:106964, 2020.

[9] Rolf Isermann and Peter Balle. Trends in the application of model-based fault detection and diagnosis of technical processes. *Control engineering practice*, 5(5):709–719, 1997.

[10] Rolf Isermann. Supervision, fault-detection and fault-diagnosis methods—an introduction. *Control engineering practice*, 5(5):639–652, 1997.

[11] Michael Quinn Patton. Enhancing the quality and credibility of qualitative analysis. *Health services research*, 34(5 Pt 2):1189, 1999.

[12] Geneviève Dreyfus-Armand, Robert Frank, Marie-Françoise Lévy, and Michelle Zancarini-Fournel. *Les années 68: le temps de la contestation*. Complexe Paris, 2000.

[13] Haithem Derbel. *Diagnostic à base de modèles des systèmes temporisés et d'une sous-classe de systèmes dynamiques hybrides*. PhD thesis, Université Joseph-Fourier-Grenoble I, 2009.

[14] Nassim Laouti. *Diagnostic de défauts par les Machines à Vecteurs Supports: application à différents systèmes mutivariables nonlinéaires*. PhD thesis, Université Claude Bernard-Lyon I, 2012.

[15] You-Jin Park, Shu-Kai S Fan, and Chia-Yu Hsu. A review on fault detection and process diagnostics in industrial processes. *Processes*, 8(9):1123, 2020.

[16] NV Rozhentcova, AR Galyautdinova, RA Khayaliev, Alex V Udaratin, and Svetlana Ilyashenko. Automated diagnostic system for power transformers using a qr code. *International Journal of Technology*, 11(8):1519–1527, 2020.

[17] Shen Yuong Wong, Xiaofeng Ye, Fengkai Guo, and Hui Hwang Goh. Computational intelligence for preventive maintenance of power transformers. *Applied Soft Computing*, 114:108129, 2022.

[18] Ronan Le Lagadec, Laura Rubio, Larissa Alexandrova, Rubén A Toscano, Ekaterina V Ivanova, Rolandas Meškys, Valdas Laurinavičius, Michel Pfeffer, and Alexander D Ryabov. Cyclometalated n, n-dimethylbenzylamine ruthenium (ii) complexes [ru (c6hr1r2r3-o-ch2nme2)(bpy)(rcn) 2] pf6 for bioapplications: synthesis, characterization, crystal structures, redox properties, and reactivity toward pqq-dependent glucose dehydrogenase. *Journal of organometallic chemistry*, 689(25):4820–4832, 2004.

[19] Payman Hajihosseini, Mohammad Mousavi Anzehaee, and Behzad Behnam. Fault detection and isolation in the challenging tennessee eastman process by using image processing techniques. *ISA transactions*, 79:137–146, 2018.

[20] Salowa Methnani. *Diagnostic, reconstruction et identification des défauts capteurs et actionneurs: application aux station d'épurations des eaux usées*. PhD thesis, Université de Toulon; École nationale d'ingénieurs de Sfax (Tunisie), 2012.

[21] Heiko Webert, Tamara Döß, Lukas Kaupp, and Stephan Simons. Fault handling in industry 4.0: definition, process and applications. *Sensors*, 22(6):2205, 2022.

[22] PS Helliwell and WJ15708931 Taylor. Classification and diagnostic criteria for psoriatic arthritis. *Annals of the rheumatic diseases*, 64(suppl 2):ii3–ii8, 2005.

[23] C Niek van Dijk, Umile Giuseppe Longo, Mattia Loppini, Pino Florio, Ludovica Maltese, Mauro Ciuffreda, and Vincenzo Denaro. Classification and diagnosis of acute isolated syndesmotic injuries: Esska-afas consensus and guidelines. *Knee Surgery, Sports Traumatology, Arthroscopy*, 24:1200–1216, 2016.

[24] Jiangquan Zhang, Sun Yi, GUO Liang, GAO Hongli, HONG Xin, and SONG Hongliang. A new bearing fault diagnosis method based on modified convolutional neural networks. *Chinese Journal of Aeronautics*, 33(2):439–447, 2020.

[25] Janan Zaytoon and Stéphane Lafortune. Overview of fault diagnosis methods for discrete event systems. *Annual Reviews in Control*, 37(2):308–320, 2013.

[26] Raffaele Petrone, Zhixue Zheng, Daniel Hissel, Marie-Cécile Péra, Cesare Pianese, Marco Sorrentino, Mohamed Béchérif, and Nadia Yousfi-Steiner. A review on model-based diagnosis methodologies for pemfcs. *International Journal of Hydrogen Energy*, 38(17):7077–7091, 2013.

[27] EYEY Chow and Alan Willsky. Analytical redundancy and the design of robust failure detection systems. *IEEE Transactions on automatic control*, 29(7):603–614, 1984.

[28] Louise Travé-Massuyes, Teresa Escobet, and Xavier Olive. Diagnosability analysis based on component-supported analytical redundancy relations. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 36(6):1146–1160, 2006.

[29] Ben Salem et al. Principal component analysis (pca). *La Tunisie Medicale*, 99(4):383–389, 2021.

[30] Mahbubunnabi Tamal. Intensity threshold based solid tumour segmentation method for positron emission tomography (pet) images: a review. *Heliyon*, 6(10):e05267, 2020.

[31] Chengze Liu, Andrzej Cichon, Grzegorz Królczyk, and Zhixiong Li. Technology development and commercial applications of industrial fault diagnosis system: a review. *The International Journal of Advanced Manufacturing Technology*, pages 1–33, 2021.

[32] Takio Kurita. Principal component analysis (pca). *Computer Vision: A Reference Guide*, pages 1–4, 2019.

[33] Ferath Kherif and Adeliya Latypova. Principal component analysis. In *Machine Learning*, pages 209–225. Elsevier, 2020.

[34] MA Bin Shams, HM Budman, and TA Duever. Fault detection, identification and diagnosis using cusum based pca. *Chemical Engineering Science*, 66(20):4488–4498, 2011.

[35] Peña Malavera, L Gutierrez, and M Balzarini. Componentes principales en mapeo asociativo principal components in associative mapping.

[36] Sergio Valle, Weihua Li, and S Joe Qin. Selection of the number of principal components: the variance of the reconstruction error criterion with a comparison to other methods. *Industrial & Engineering Chemistry Research*, 38(11):4389–4401, 1999.

[37] A Haydar, MA Mandal, MB Ahmed, MM Hannan, R Karim, MA Razvy, UK Roy, and M Salahin. Studies on genetic variability and interrelationship among the different traits in tomato (lycopersicon esculentum mill.). *Middle-East J. Sci. Res*, 2(3-4):139–142, 2007.

[38] Adda Saadoune, Abderrahmane Amrouche, and Sid-Ahmed Selouani. Perceptual subspace speech enhancement using variance of the reconstruction error. *Digital signal processing*, 24:187–196, 2014.

[39] Alok Mukherjee, Palash K Kundu, and Arabinda Das. A supervised principal component analysis-based approach of fault localization in transmission lines for single line to ground faults. *Electrical Engineering*, pages 1–14, 2021.

[40] Sagi Rathna Prasad and AS Sekhar. Detection and localization of fatigue-induced transverse crack in a rotor shaft using principal component analysis. *Structural Health Monitoring*, 20(2):513–531, 2021.

[41] Marco AF Pimentel, David A Clifton, Lei Clifton, and Lionel Tarassenko. A review of novelty detection. *Signal processing*, 99:215–249, 2014.

[42] Radu-Emil Precup, Plamen Angelov, Bruno Sielly Jales Costa, and Moamar Sayed-Mouchaweh. An overview on fault diagnosis and nature-inspired optimal control of industrial process applications. *Computers in Industry*, 74:75–94, 2015.

[43] Xuemei Ding, Yuhua Li, Ammar Belatreche, and Liam P Maguire. An experimental evaluation of novelty detection methods. *Neurocomputing*, 135:313–327, 2014.

[44] Thomas J Harris and William H Ross. Statistical process control procedures for correlated observations. *The canadian journal of chemical engineering*, 69(1):48–57, 1991.

[45] John F MacGregor and Theodora Kourti. Statistical process control of multivariate processes. *Control engineering practice*, 3(3):403–414, 1995.

[46] Q Peter He and Jin Wang. Statistical process monitoring as a big data analytics tool for smart manufacturing. *Journal of Process Control*, 67:35–43, 2018.

[47] Youn-Min Chou, Robert L Mason, and John C Young. Power comparisons for a hotelling's t2 statistic. *Communications in Statistics-Simulation and Computation*, 28(4):1031–1050, 1999.

[48] Aneetha Avalappampatty Sivasamy and Bose Sundan. A dynamic intrusion detection system based on multivariate hotelling's t2 statistics approach for network environments. *The Scientific World Journal*, 2015, 2015.

[49] H Henry Yue and S Joe Qin. Reconstruction-based fault identification using a combined index. *Industrial & engineering chemistry research*, 40(20):4403–4414, 2001.

[50] Tran Duc Chung, Rosdiazli B Ibrahim, Vijanth Sagayan Asirvadam, Nordin B Saad, and Sabo Miya Hassan. Adopting ewma filter on a fast sampling wired link contention in wirelesshart control system. *IEEE Transactions on Instrumentation and Measurement*, 65(4):836–845, 2016.

[51] Jicong Fan, S Joe Qin, and Youqing Wang. Online monitoring of nonlinear multivariate industrial processes using filtering kica–pca. *Control Engineering Practice*, 22:205–216, 2014.

[52] Yen-Chi Chen. A tutorial on kernel density estimation and recent advances. *Biostatistics & Epidemiology*, 1(1):161–187, 2017.

[53] Takuya Kishimoto, Hanwool Woo, Ren Komatsu, Yusuke Tamura, Hideki Tomita, Kenji Shimazoe, Atsushi Yamashita, and Hajime Asama. Path planning for localization of radiation sources based on principal component analysis. *Applied Sciences*, 11(10):4707, 2021.

[54] Johan A Westerhuis, Stephen P Gurden, and Age K Smilde. Generalized contribution plots in multivariate statistical process monitoring. *Chemometrics and intelligent laboratory systems*, 51(1):95–114, 2000.

[55] AK Conlin, EB Martin, and AJ Morris. Confidence limits for contribution plots. *Journal of Chemometrics*, 14(5-6):725–736, 2000.

[56] Changkyu Lee, Sang Wook Choi, and In-Beum Lee. Sensor fault identification based on time-lagged pca in dynamic processes. *Chemometrics and Intelligent Laboratory Systems*, 70(2):165–178, 2004.

[57] Jeanne Fine and Yves Romain. Reduced principal component analysis. *Communications in Statistics-Theory and Methods*, 15(4):493–512, 1984.

[58] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Kernel principal component analysis. In *Artificial Neural Networks—ICANN'97: 7th International Conference Lausanne, Switzerland, October 8–10, 1997 Proceeedings*, pages 583–588. Springer, 2005.

[59] Xiangyu Kong, Changhua Hu, Zhansheng Duan, Xiangyu Kong, Changhua Hu, and Zhansheng Duan. Generalized principal component analysis. *Principal Component Analysis Networks and Algorithms*, pages 185–233, 2017.

[60] Hui Zou, Trevor Hastie, and Robert Tibshirani. Sparse principal component analysis. *Journal of computational and graphical statistics*, 15(2):265–286, 2006.

[61] Deyu Meng, Qian Zhao, and Zongben Xu. Improve robustness of sparse pca by l1-norm maximization. *Pattern Recognition*, 45(1):487–497, 2012.

[62] Ron Zass and Amnon Shashua. Nonnegative sparse pca. *Advances in neural information processing systems*, 19, 2006.

[63] Wenwen Min, Juan Liu, and Shihua Zhang. Edge-group sparse pca for network-guided high dimensional data analysis. *Bioinformatics*, 34(20):3479–3487, 2018.

[64] Chris M Bishop. Neural networks and their applications. *Review of scientific instruments*, 65(6):1803–1832, 1994.

[65] Sagar Sharma, Simone Sharma, and Anidhya Athaiya. Activation functions in neural networks. *Towards Data Sci*, 6(12):310–316, 2017.

[66] Dor Bank, Noam Koenigstein, and Raja Giryes. Autoencoders. *arXiv preprint arXiv:2003.05991*, 2020.

[67] Domenico Buongiorno, Cristian Camardella, Giacomo Donato Cascarano, Luis Pelaez Murciego, Michele Barsotti, Irio De Feudis, Antonio Frisoli, and Vitoantonio Bevilacqua. An undercomplete autoencoder to extract muscle synergies for motor intention detection. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2019.

[68] Andrew Ng et al. Sparse autoencoder. *CS294A Lecture notes*, 72(2011):1–19, 2011.

[69] Andri Ashfahani, Mahardhika Pratama, Edwin Lughofer, and Yew-Soon Ong. Devdan: Deep evolving denoising autoencoder. *Neurocomputing*, 390:297–314, 2020.

[70] Jun Xu, Lei Xiang, Qingshan Liu, Hannah Gilmore, Jianzhong Wu, Jinghai Tang, and Anant Madabhushi. Stacked sparse autoencoder (ssae) for nuclei detection on breast cancer histopathology images. *IEEE transactions on medical imaging*, 35(1):119–130, 2015.

[71] Wafa Bougheloum, Mounir Bekaik, and Sofiane Gherbi. Multimode system condition monitoring using sparsity reconstruction for quality control. *International Journal of Electrical & Computer Engineering (2088-8708)*, 13(3), 2023.

[72] Shi Na, Liu Xumin, and Guan Yong. Research on k-means clustering algorithm: An improved k-means clustering algorithm. In *2010 Third International Symposium on intelligent information technology and security informatics*, pages 63–67. Ieee, 2010.

[73] Janmenjoy Nayak, Bighnaraj Naik, and HSr Behera. Fuzzy c-means (fcm) clustering algorithm: a decade review from 2000 to 2014. In *Computational Intelligence in Data Mining-Volume 2: Proceedings of the International Conference on CIDM, 20-21 December 2014*, pages 133–149. Springer, 2015.

[74] R Babuka, PJ Van der Veen, and Uzay Kaymak. Improved covariance estimation for gustafson-kessel clustering. In *2002 IEEE World Congress on Computational Intelligence. 2002 IEEE International Conference on Fuzzy Systems. FUZZ-IEEE'02. Proceedings (Cat. No. 02CH37291)*, volume 2, pages 1081–1085. IEEE, 2002.