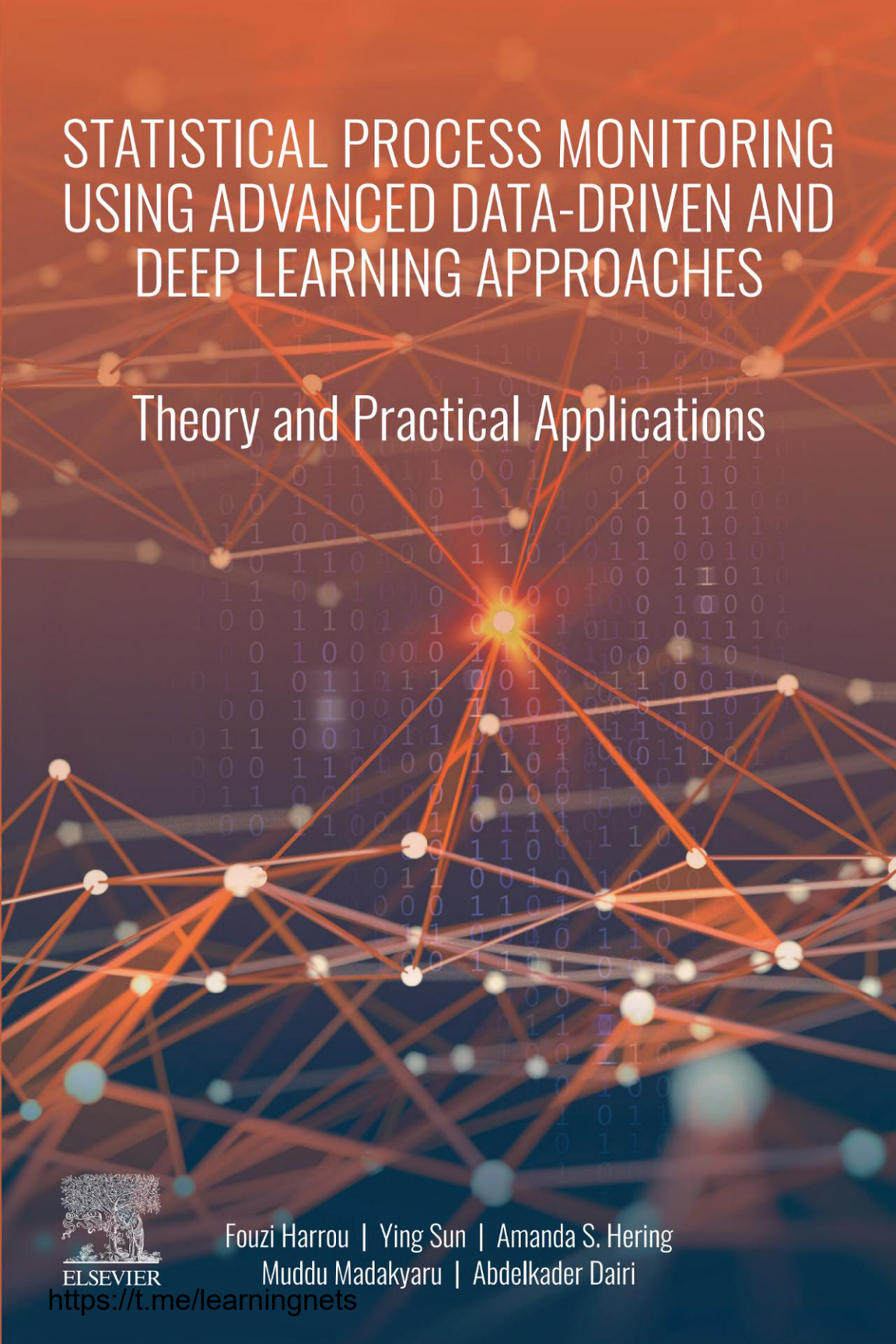
STATISTICAL PROCESS MONITORING USING ADVANCED DATA-DRIVEN AND DEEP LEARNING APPROACHES

Theory and Practical Applications



Fouzi Harrou | Ying Sun | Amanda S. Hering
Muddu Madakyaru | Abdelkader Dairi

ELSEVIER

https://t.me/learningnets

Statistical Process Monitoring using Advanced Data-Driven and Deep Learning Approaches

https://t.me/learningnets

https://t.me/learningnets

Statistical Process Monitoring using Advanced Data-Driven and Deep Learning Approaches

Theory and Practical Applications

Fouzi Harrou
King Abdullah University of Science and Technology
Computer, Electrical and Mathematical Sciences and Engineering (CEMSE) Division
Thuwal, Saudi Arabia

Ying Sun
King Abdullah University of Science and Technology
Computer, Electrical and Mathematical Sciences and Engineering (CEMSE) Division
Thuwal, Saudi Arabia

Amanda S. Hering
Baylor University, Dept of Statistical Science
Waco, TX, United States

Muddu Madakyaru
Department of Chemical Engineering, Manipal Institute of Technology
Manipal Academy of Higher Education
Manipal, India

Abdelkader Dairi
University of Science and Technology of Oran-Mohamed Boudiaf
Computer Science Department, Signal, Image and Speech Laboratory
Oran, Algeria



ELSEVIER

https://t.me/learningnets

Elsevier
Radarweg 29, PO Box 211, 1000 AE Amsterdam, Netherlands
The Boulevard, Langford Lane, Kidlington, Oxford OX5 1GB, United Kingdom
50 Hampshire Street, 5th Floor, Cambridge, MA 02139, United States

Copyright © 2021 Elsevier Inc. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or any information storage and retrieval system, without permission in writing from the publisher. Details on how to seek permission, further information about the Publisher's permissions policies and our arrangements with organizations such as the Copyright Clearance Center and the Copyright Licensing Agency, can be found at our website: www.elsevier.com/permissions.

This book and the individual contributions contained in it are protected under copyright by the Publisher (other than as may be noted herein).

Notices

Knowledge and best practice in this field are constantly changing. As new research and experience broaden our understanding, changes in research methods, professional practices, or medical treatment may become necessary.

Practitioners and researchers must always rely on their own experience and knowledge in evaluating and using any information, methods, compounds, or experiments described herein. In using such information or methods they should be mindful of their own safety and the safety of others, including parties for whom they have a professional responsibility.

To the fullest extent of the law, neither the Publisher nor the authors, contributors, or editors, assume any liability for any injury and/or damage to persons or property as a matter of products liability, negligence or otherwise, or from any use or operation of any methods, products, instructions, or ideas contained in the material herein.

Library of Congress Cataloging-in-Publication Data
A catalog record for this book is available from the Library of Congress

British Library Cataloguing-in-Publication Data
A catalogue record for this book is available from the British Library

ISBN: 978-0-12-819365-5

For information on all Elsevier publications
visit our website at https://www.elsevier.com/books-and-journals

Publisher: Susan Dennis
Acquisitions Editor: Anita A Koch
Editorial Project Manager: Lena Sparks
Production Project Manager: Kumar Anbazhagan
Designer: Miles Hitchen

Typeset by VTeX

ELSEVIER Book Aid International Working together to grow libraries in developing countries
www.elsevier.com • www.bookaid.org

https://t.me/learningnets

Contents

Preface ix
Acknowledgments xi

1. Introduction

1.1 Introduction 1
1.1.1 Motivation: why process monitoring 1
1.1.2 Types of faults 2
1.1.3 Process monitoring 4
1.1.4 Physical redundancy vs analytical redundancy 5
1.2 Process monitoring methods 6
1.2.1 Model-based methods 7
1.2.2 Knowledge-based methods 9
1.2.3 Data-based monitoring methods 9
1.3 Fault detection metrics 13
1.4 Conclusion 14
References 15

2. Linear latent variable regression (LVR)-based process monitoring

2.1 Introduction 19
2.2 Development of linear LVR models 20
2.2.1 Full rank methods 21
2.2.2 Latent variable regression (LVR) models 22
2.3 Dynamic LVR models 30
2.4 Process monitoring methods 32
2.4.1 Univariate chart for process monitoring 32
2.4.2 Distribution-based process monitoring schemes 39
2.4.3 Multivariate process monitoring schemes with parametric and nonparametric thresholds 44
2.5 Linear LVR-based process monitoring strategies 47
2.5.1 Conventional LVR monitoring statistics 47
2.5.2 Fault isolation 50
2.6 Cases studies 53
2.6.1 Simulated example 53

https://t.me/learningnets

2.6.2 Monitoring influent measurements at water resource recovery facilities 55
2.7 Discussion 63
References 63

3. Fault isolation

3.1 Introduction 71
3.1.1 Pitfalls of standardizing data 72
3.1.2 Shortcomings of contribution plots/scores 77
3.2 Fault isolation 79
3.2.1 Variable thinning 79
3.2.2 Iterative traditional isolation 80
3.2.3 Variable selection methods 83
3.3 Fault classification 99
3.4 Fault isolation metrics 100
3.4.1 Fault isolation errors 101
3.4.2 Precision and recall 102
3.4.3 Phase I FI metrics 102
3.4.4 Discussion 103
3.5 Case studies 103
3.5.1 Retrospective fault isolation 104
3.5.2 Real-time fault isolation 108
3.6 Further reading 111
References 112

4. Nonlinear latent variable regression methods

4.1 Introduction 119
4.2 Limitations of linear LVR methods for process monitoring 121
4.3 Developing nonlinear LVR methods for process monitoring 123
4.3.1 Nonlinear partial least squares 123
4.3.2 ANFIS-PLS modeling framework 127
4.3.3 Kernel PCA 131
4.3.4 Kernel principal components analysis (KPCA) model 131
4.3.5 KPCA-based fault detection procedures 135
4.4 Cases study: monitoring WWTP 138
4.4.1 Anomaly detection using KPCA-OCSVM method 139
4.5 Simulated synthetic data 142
4.5.1 Application of plug flow reactor 143
4.6 Discussion 149
References 151

5. Multiscale latent variable regression-based process monitoring methods

5.1 Introduction 155

https://t.me/learningnets

5.2 Theoretical background of wavelet-based data representation 158
5.2.1 Wavelet transform 159
5.2.2 Multiscale representation of data using wavelets 159
5.2.3 Advantages of multiscale representation 164
5.3 Multiscale filtering using wavelets 167
5.3.1 Single scale filter method 167
5.3.2 Multiscale filtering methods 168
5.3.3 Advantages of multiscale denoising 169
5.4 Wavelet-based multiscale univariate monitoring techniques 170
5.4.1 An illustrative example 172
5.5 Multiscale LVR modeling 176
5.5.1 Benefits of multiscale denoising in LVR modeling 176
5.6 Multiscale LVR modeling 177
5.7 Results and discussions 180
5.7.1 Application with synthetic data 180
5.7.2 Application of monitoring distillation column 183
5.8 Discussion 186
References 188

6. Unsupervised deep learning-based process monitoring methods

6.1 Introduction 193
6.2 Clustering 195
6.2.1 Partition-based clustering techniques 196
6.2.2 Hierarchy-based clustering techniques 197
6.2.3 Density-based approach 198
6.2.4 Expectation maximization 201
6.3 One-class classification 202
6.3.1 One-class SVM 202
6.3.2 Support vector data description (SVDD) 203
6.4 Deep learning models 206
6.4.1 Autoencoders 206
6.4.2 Probabilistic models 210
6.4.3 Deep neural networks 213
6.4.4 Deep Boltzmann machine 215
6.5 Deep learning-based clustering schemes for process monitoring 217
6.6 Discussion 218
References 219

7. Unsupervised recurrent deep learning scheme for process monitoring

7.1 Introduction 225
7.2 Recurrent neural networks approach 227
7.2.1 Basics of recurrent neural networks 227
7.2.2 Long short-term memory 229

https://t.me/learningnets

7.2.3 Gated recurrent neural networks 234
7.3 Hybrid deep models 235
7.3.1 RNN-RBM 236
7.3.2 RNN-RBM method 237
7.3.3 LSTM-RBM model 238
7.3.4 LSTM-DBN 239
7.4 Recurrent deep learning-based process monitoring 241
7.4.1 Residuals-based process monitoring approaches 242
7.4.2 Recurrent deep learning-based clustering schemes for process monitoring 243
7.5 Applications: monitoring influent conditions at WWTP 244
7.6 Discussion 250
References 251

8. Case studies

8.1 Introduction 255
8.2 Stereovision 258
8.2.1 Deep stacked autoencoder-based KNN approach 261
8.2.2 Data description 266
8.2.3 Results and discussion 266
8.2.4 Model trained using data with no obstacles 267
8.2.5 Evaluation of performance for busy scenes 269
8.2.6 Obstacle detection using the Bahnhof dataset 271
8.3 Detecting abnormal ozone measurements using deep learning 274
8.3.1 Introduction 274
8.3.2 Data description 276
8.3.3 Ozone monitoring based on deep learning approaches 278
8.3.4 Detection results 284
8.4 Monitoring of a wastewater treatment plant using deep learning 288
8.4.1 Introduction 288
8.4.2 Proposed DBN-based kNN, OCSVM, and k-means algorithms 290
8.4.3 Real data application: monitoring a decentralized wastewater treatment plant in Golden, CO, USA 291
8.4.4 Conclusion 297
References 297

9. Conclusion and further research directions

References 308

Index 311

https://t.me/learningnets

Preface

Anomaly detection and isolation have a vital role in modern industrial processes to enhance productivity, efficiency, and safety, as well as to avoid expensive maintenance. Therefore, it is important to be able to detect and identify any possible anomalies or failures in the system as early as possible. Generally, anomalies in modern automatic processes are difficult to avoid and may result in serious process degradations. The role of detection is to identify any anomaly event and indicate a distance from the system behavior compared to its nominal behavior. Furthermore, anomaly isolation determines the probable source of the detected anomaly. To illustrate, an accidental or even deliberate contamination of a drinking water distribution network can lead to financial losses, as well as to serious health risks. Therefore, early detection of anomalies is crucial not only to maintain proper process operation but also for the sake of people's health. Today engineered and environmental processes have become far more complex due to advances in technology. Multiple key variables need to be monitored simultaneously, and data may have both temporal and spatial aspects. New features of these processes require new and better statistical tools for process monitoring.

Early detection and isolation of potential faults in complex engineering and environmental processes have proven to be particularly challenging. In the absence of a physics-based process model, data-driven statistical techniques for process monitoring have proved themselves in practice over the past four decades. These approaches use information derived directly from input data and require no explicit models for which development is usually costly or time-consuming. This book is intended to report recent developments in statistical process monitoring using advanced data-driven and deep learning techniques. The book is divided into nine chapters, and they are grouped into two parts. The objective of the first part is to tackle multivariate challenges in process monitoring by merging the advantages of univariate and traditional multivariate techniques to enhance their performance and widen their practical applicability. The second part aims to merge the desirable properties of shallow learning approaches, such as a one-class support vector machine, k-nearest neighbors, and unsupervised deep learning approaches to develop more sophisticated and efficient monitoring techniques. Throughout the book, the presented approaches are demonstrated using experimental data from many processes including wastewater treatment plants at KAUST and Golden, CO, USA, ozone air quality data,

https://t.me/learningnets

and stereovision data for obstacle detection in driving environments. Thus, the reader will find illustrative examples from a range of environmental and engineering processes.

The book should be of interest to engineering and academic readers from process chemometrics and data analytics, process monitoring and control, data scientists, applied statistics, and industrial statisticians. In fact, this book can be assimilated by advanced undergraduates and graduate students having knowledge of basic multivariate statistical analysis and machine learning.

https://t.me/learningnets

Acknowledgments

Addressing anomaly detection and isolation is essential to promptly detect abnormalities and helpful in the decision making of the operators to better optimize, take corrective actions, and maintain downstream processes. This book is primarily based on data-driven based approaches for anomaly detection and isolation. The reader of this book will gain an in-depth understanding of fault detection and isolation in complex and multivariate systems, familiarizing with the most suitable data-driven based techniques including multivariate statistical techniques and deep learning-based methods. It gives the reader several real engineering and environmental applications to clearly show the implementation of anomaly detection and isolation approaches.

Ying Sun and Fouzi Harrou would like to gratefully acknowledge the financial support by funding from King Abdullah University of Science and Technology (KAUST), Office of Sponsored Research (OSR) under Award No: OSR-2019-CRG7-3800 and OSR-2015-CRG4-2582. They would like also to express their sincere gratitude to the team of Publication Services and Researcher Support at KAUST for their support. In addition, we would also like to thank Professor Tzahi Cath of Colorado School of Mines who provided the decentralized wastewater treatment data.

Amanda S. Hering would like to thank Professor Tzahi Cath of Colorado School of Mines who has been instrumental in introducing her to fault isolation problems and who has shared data from his facilities with her. She would also like to thank her graduate students, Molly Klanderman and Kathryn Newhart; their expertise and insight accumulated over the course of working together for the past few years has been invaluable. Her work on this project has been supported by King Abdullah University of Science and Technology (KAUST) Office of Sponsored Research (OSR), Grant/Award Number: OSR-2015-CRG4-2582; Partnerships for Innovation: Building Innovation Capacity, National Science Foundation, Grant/Award Number: 1632227; the National Science Foundation Engineering Research Center program under cooperative agreement EEC-1028968 (ReNUWIt); and Baylor University through a research leave sabbatical.

https://t.me/learningnets

Muddu Madakyaru would like to thank the Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India, for continuous support during the preparation of this book.

Finally, we would like to thank Lena Sparks, Author Service Manager, for her continuous assistance during the preparation of this book.

https://t.me/learningnets

Chapter 1

Introduction

1.1 Introduction

1.1.1 Motivation: why process monitoring

Recent decades have witnessed a huge growth in new technologies and advancements in instrumentation, industrial systems, and environmental processes, which are becoming increasingly complex. Diagnostic operation has become an essential element of these processes and systems to ensure their operational reliability and availability. In an environment where productivity and safety are paramount, failing to detect anomalies in a process can lead to harmful effects to a plant's productivity, profitability, and safety. Several serious accidents have happened in the past few decades in various industrial plants across the world, including the Bhopal gas tragedy [1,2], the Piper Alpha explosion [3,4], the accidents at the Mina al-Ahmadi Kuwait refinery [5] and two photovoltaic plants in the US burned in 2009 and 2011 (a 383 KWp PV array in Bakersfield, CA and a 1.208 MWp power plant in Mount Holly, NC, respectively) [6]. The Bhopal accident, also referred to as the Bhopal gas disaster, was a gas leak accident at the Union Carbide pesticide plant in India in 1984 that resulted in over 3000 deaths and over 400,000 others gravely injured in the local area around the plant [1,2]. The explosion of the Piper Alpha oil production platform, which is located in the North Sea and managed by Occidental Petroleum, caused 167 deaths and a financial loss of around $3.4 billion [3,4]. In 2000, an explosion occurred in the Mina Al-Ahmadi oil refinery in Kuwait, killing five people and causing serious damage to the plant. The explosion was caused by a defect in a condensate line in a refinery. Nimmo [7] has estimated that the petrochemical industry in the USA can avoid losing up to $20 billion per year if anomalies in inspected processes could be discovered in time. In safety-critical systems such as nuclear reactors and aircrafts, undetected faults may lead to catastrophic accidents. For example, the pilot of the American Airlines DC10 that crashed at Chicago O'Hare International Airport was notified of a fault only 15 seconds before the accident happened, giving the pilot too little time to react; this crash could easily have been avoided according to [8]. Recently, the Fukushima accident of 2011 in Japan highlighted the importance of developing accurate and efficient monitoring systems for nuclear plants. Essentially, monitoring of industrial processes represents the backbone for ensuring the safe operation of these processes and to ensure that the process is always functioning properly.

https://doi.org/10.1016/B978-0-12-819365-5.00007-3
Copyright © 2021 Elsevier Inc. All rights reserved.
https://t.me/learningnets

1.1.2 Types of faults

Generally speaking, three main subsystems are merged to form a plant or system: sensors, actuators, and the main process itself. These systems' components are permanently exposed to faults caused by many factors, such as aging, manufacturing, and severe operating conditions. A *fault* or *anomaly* is a tolerable deviation of a characteristic property of a variable from its acceptable behavior that could lead to a failure in the system if it is not detected early enough so that the necessary correction can be performed [9]. Conventionally, a fault, if it is not detected in time, could progress to produce a failure or malfunction. Note that there is a distinction between failure and malfunction; this distinction is important. A *malfunction* can be defined as an intermittent deviation of the accomplishment of a process's intended function [10], whereas *failure* is a persistent suspension of a process's capability to perform a demanded function within indicated operating conditions [10].

In industrial processes, a fault or an abnormal event is defined as the departure of a calculated process variable from its acceptable region of operation. The underlying causes of a fault can be malfunctions or changes in sensor, actuator, or process components:

- *Process faults or structural changes.* Structural change usually takes place within the process itself due to a hard failure of the equipment. The information flow between the different variables is affected because of these changes. Failure of a central controller, a broken or leaking pipe, and a stuck valve are a few examples of process faults. These faults are distinguished by slow changes across various variables in the process.
- *Faults in sensors and actuators.* Sensors and actuators play a very important role in the functioning of any industrial process since they provide feedback signals that are crucial for the control of the plant. Actuators are essential for transforming control inputs into appropriate actuation signals (e.g., forces and torques needed for system operation). Generally, actuator faults may lead to higher power consumption or even a total loss of control [11]. Faults in pumps and motors are examples of actuator faults. On the other hand, sensor-based errors include positive or negative bias errors, out of range errors, precision degradation error, and drift sensor error. Sensor faults are generally characterized by quick deviations in a few numbers of process variables. Fig. 1.1 shows examples of the most commonly occurring sensor faults: bias, drift, degradation, and sensor freezing.

We can also find in the literature another type of anomaly called gross parameter changes in a model. Indeed, parameter failure occurs when there is a disturbance entering the monitored process from the environment through one or more variables. Some common examples of such malfunctions include a change in the heat transfer coefficient, a change in the temperature coefficient in a heat exchanger, a change in the liquid flow rate, or a change in the concentration of a reactant.

https://t.me/learningnets

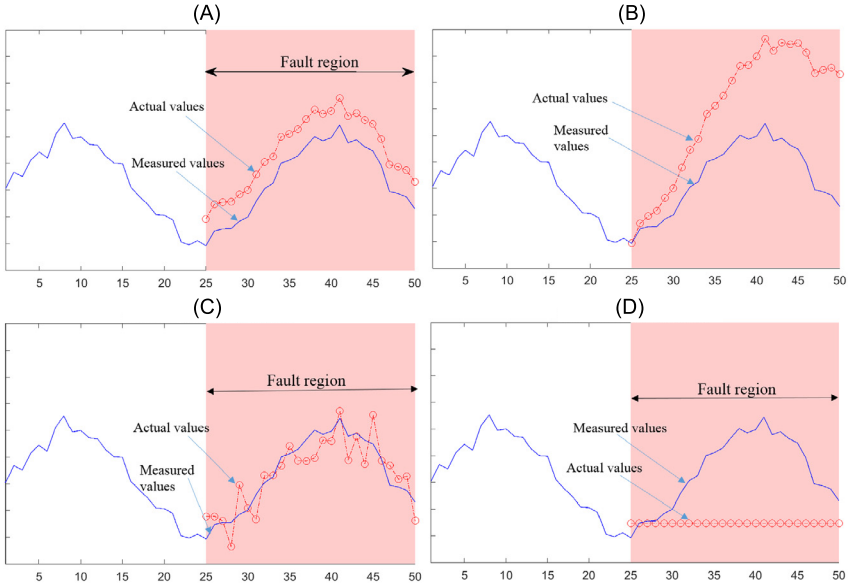


FIGURE 1.1 Commonly occurring sensor faults. (A) Bias sensor fault. (B) Drift sensor fault. (C) Degradation sensor fault. (D) Freezing sensor fault.

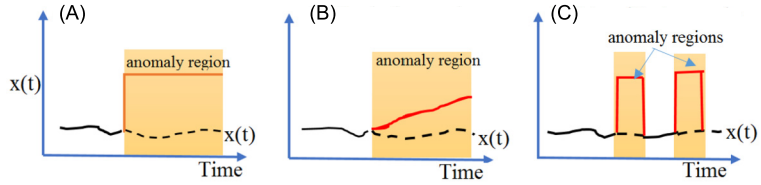


FIGURE 1.2 Fault types. (A) Abrupt anomaly. (B) Gradual anomaly. (C) Intermittent anomaly.

Thus, sensor or process faults can affect the normal functioning of a process plant. In today's highly competitive industrial environment, improved monitoring of processes is an important step towards increasing the efficiency of production facilities.

In practice, there is a tendency to classify anomalies according to their time-variant behavior. Fig. 1.2 illustrates three commonly occurring types of anomalies that can be distinguished by their time-variant form: abrupt, incipient, and intermittent faults. Abrupt anomalies happen regularly in real systems and are generally typified by a sudden change in a variable from its normal operating range (Fig. 1.2A). The faulty measurement can be formally expressed as

$$M(t) = \begin{cases} r(t), & t < t_a, \\ r(t) + F, & t \geq t_a, \end{cases} \tag{1.1}$$

where F is a bias that happens at the time instant t_s.

https://t.me/learningnets

The drift anomaly type can be caused by the aging or degradation of a sensor and can be viewed as a linear change of the magnitude of fault in time. Here, the measurement corrupted with a drift fault is modeled as

$$m(t) = \begin{cases} r(t), & t < t_a, \\ r(t) + \theta(t - t_a), & t \geq t_s, \end{cases} \tag{1.2}$$

where θ is a slope of the slow drift and t_a is the start time of the occurred fault. Finally, intermittent faults are faults characterized by discontinuous occurrence in time; they occur and disappear repeatedly (Fig. 1.2C).

Generally, industrial and environmental processes are exposed to various types of faults that negatively affect their productivity and efficiency. According to the form in which the fault is introduced, faults can be further classified as *additive and multiplicative faults*. Additive faults often appear as offsets of sensors or as additive bias, while multiplicative faults influence process parameters. Specifically, in an additive fault, the measurable variable $Y(t)$ is corrupted by an additive fault, θ_t, as $Y = Y_t + \theta_t$. On the other hand, a multiplicative fault influences a measurable variable Y by the product of another variable U with θ_t (i.e., $Y = (a + f)U_t$), where U_t is the input variable.

1.1.3 Process monitoring

Before automation became commonplace in the field of process monitoring, human operators carried out important control tasks in managing process plants. However, the complete reliance on human operators to cope with abnormal events and emergencies has become increasingly difficult because of the complexity and a large number of variables in modern process plants. Considering such difficult conditions, it is understandable that human operators tend to make mistakes that can lead to significant economic, safety, and environmental problems. Thanks to advancements in technology over recent years, automation of process fault detection and isolation has been a major milestone in automatic process monitoring. Automatic process monitoring has been shown to respond very well to abnormal events in a process plant with much fewer mistakes compared to fault management by human operators.

The demand for a monitoring system that is capable of appropriately detecting abnormal changes (sensor or process faults) has attracted the attention of researchers from different fields. The *detection and isolation* of anomalies that may occur in a monitored system are the two main elements of process monitoring (Fig. 1.3). The purpose of the detection step is to detect abnormal changes that affect the behavior of the monitored system. Once the anomaly is detected, effective system operation also requires evaluation of the risk of a system shutdown, followed by fault isolation or correction before the anomaly contaminates the process performance [12,13]. The purpose of fault isolation is to determine the source responsible for the occurring anomalies, i.e., to determine which sensor or process component is faulty. In practice, sometimes it is also essential to

https://t.me/learningnets

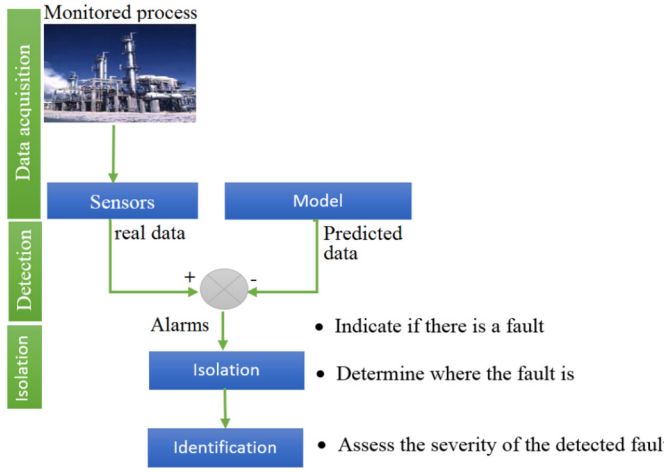


FIGURE 1.3 Steps of process monitoring.

assess the severity of the occurred fault, which is done by the fault identification step. Here, we will focus only on fault detection and isolation.

There are two types of anomaly detection:

- *Online fault detection.* The objective of online anomaly detection is to set up a decision rule capable of detecting, as quickly as possible, the transition from a normal operating state to an abnormal operating state. Online detection is based on the idea that system evolution is considered a succession of stationary modes separated by fast transitions.
- *Offline fault detection.* The purpose of offline fault detection is to detect the presence of a possible anomaly outside the use of the monitored system. The system is observed for a finite period (the system is in stationary mode), and then, based on these observations, a decision is made on the state of the monitored system. Offline detection methods rely on an observation number fixed a priori, where the observations also come from the same law.

1.1.4 Physical redundancy vs analytical redundancy

Process monitoring is essentially based on the exploitation of redundant sources of information. There are two types of redundancy in the process: *physical redundancy and analytical redundancy* (Fig. 1.4A–B). The essence of hardware or physical redundancy, which is a traditional method in process monitoring, consists of measuring a particular process variable using several sensors (e.g., two or more sensors). To detect and isolate simple faults, the number of sensors to use should be doubled. Specifically, under normal conditions, one sensor is sufficient to monitor a particular variable, but adding at least two extra sensors is generally needed to guarantee reliable measurements and monitoring under

https://t.me/learningnets

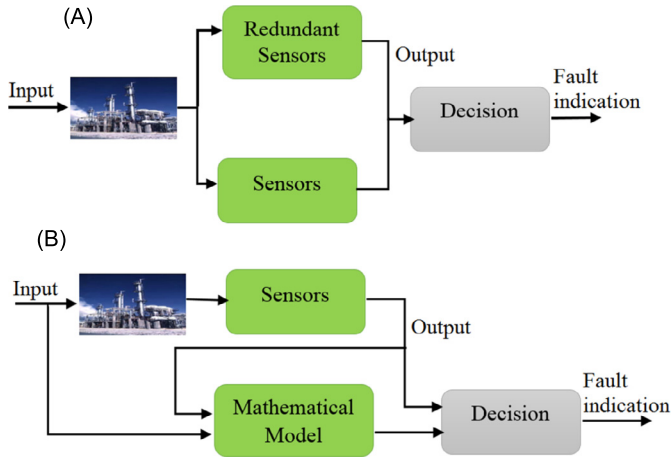


FIGURE 1.4 Conceptual representation of (A) physical and (B) analytical redundancy.

faulty conditions. Typically, fault detection and isolation are achieved by a majority vote between all the redundant sensors. This strategy has been widely used in the industry because of its reliability and simplicity of implementation. In practice, the main disadvantage of hardware redundancy is the additional cost of equipment and maintenance, as well as the space needed to install the equipment that increases complexity considerably in the already very complex systems. In addition, this method is limited in practice to sensor faults and cannot detect faults in variables that are not measured directly. This approach is mainly only justified for critical systems, such as nuclear reactors and aeronautic systems. Unlike a physical redundancy, which is performed by adding more sensors (hardware) to measure a specific process variable, the analytical redundancy does not require additional hardware because it is based on using the existing relations between the dependent measured variables that are or are not of the same nature. Analytical redundancy is a more accessible strategy that compares the measured variable with the predicted values from a mathematical model of the monitored system. It thereby exploits redundant analytical relationships among various measured variables of the monitored process and avoids replicating every hardware separately.

1.2 Process monitoring methods

Today, engineering and environmental processes have become far more complex due to advances in technology. Anomaly detection and isolation have become necessary to monitor the continuity and proper functioning of modern industrial systems and environmental processes. Depending on the field of application, the repercussions of anomalies become binding and harmful if it concerns human safety, such as in aeronautical systems and nuclear reactors. Advancements in

https://t.me/learningnets

the field of process control and automation over the last few years have yielded various methods for successful diagnosis and detection of abnormal events. To meet safety and productivity requirements, extensive theoretical and practical monitoring methods have been developed. These methods are generally divided into three families of approaches, depending on the nature of the knowledge available on the system: model-, knowledge-, and data-based methods. A thorough overview of process fault detection and diagnosis can be found in [5]. Fig. 1.5 shows a summary of various monitoring methods; this section presents a brief overview of these monitoring techniques.

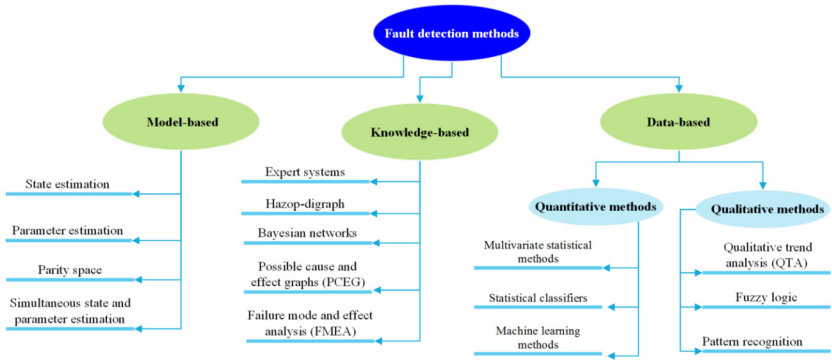


FIGURE 1.5 A summary of various fault detection approaches.

1.2.1 Model-based methods

Over the past three decades, numerous monitoring methods to improve the safety and productivity of several environmental and engineering processes have emerged. Model-based methods have proven especially useful in industrial applications where keeping the desired performance is highly required. A model-based method involves comparing the process's measured variables with the prediction from the mathematical model of the process. The conceptual schematic of the model-based fault detection is illustrated in Fig. 1.6. The

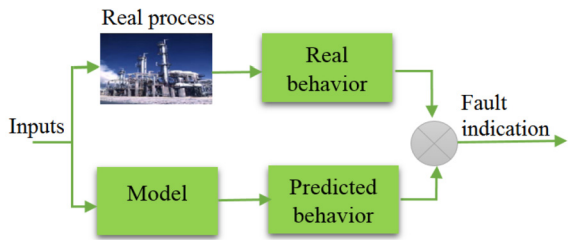


FIGURE 1.6 Conceptual schematics of model-based process monitoring.

https://t.me/learningnets

backbone of the model-based method is the generation of residuals by comparing the measurement data with their predictions from the analytical model of the monitored process. Indeed, the residuals play the role of a fault indicator. Ideally, in the absence of modeling uncertainties and errors, the residual will be zero and the model will perfectly fit the measurements. Thus, any departure of the residual from zero indicates the presence of faults. However, in practice, we cannot avoid the presence of modeling uncertainties and noise measurements. In other words, a perfectly precise analytical model of an inspected process is never available. Notice that there is a distinction between a deviation of the real measurement and its prediction from a reference model, even under no-fault conditions. Hence, instead of using the departure of residuals from zero as a fault indicator, detection can be done by constructing a detection threshold that distinguishes between fault-free residuals and anomalies. The detection performance is mainly related to the selected detection threshold. This means that if the value of the thresholds is too small, then we get repeat false alarms due to errors and uncertainties when the residuals overpass the threshold and are consequently flagged as faults; this scenario obviously must be avoided. The detection threshold should thus be computed so that the frequency of correct detection is maximized for a given small number of false alarms (e.g., 5% or 1%). To address this concern, several statistical schemes have been proposed to monitor the residuals vector, including the generalized likelihood ratio approach, cumulative sum (CUSUM) type schemes, and EWMA schemes. In the case of multivariate data, when the residuals matrix is generated, multivariate extensions of CUSUM and EWMA and T^2 are usually used to detect faults in the mean/variance of process.

In summary, fault detection and isolation using model-based methods usually take place in two distinct steps:

- The first step consists of residual generation. Ideally, these residuals must be zero in normal operation and nonzero in the presence of an anomaly. However, the presence of noise and modeling errors make the residuals fluctuate around zero. A significant divergence of the residual from zero is an indication of faults.
- The second step concerns the evaluations of the residuals based on a decision procedure for detecting and isolating faults. This is done using statistical detection techniques such as EWMA, CUSUM, and generalized likelihood ratio (GLR) test [12].

A substantial amount of research work has been carried out on model-based monitoring methods. Methods that fall into the model-based monitoring category include parity space approaches [14–17], observer-based approaches [18,19], and interval approaches [20]. A related discussion and a comprehensive survey on model-based fault detection methods can be found in [21–23].

Essentially, the detection performance of model-based approaches is closely related to the accuracy of the reference model. The availability of an accurate

https://t.me/learningnets

model that mimics the nominal behavior of the monitored process is very helpful for facilitating the detection of faulty measurements. However, for complex processes, such as those of many industrial and environmental processes with a large number of variables, deriving and developing accurate models is not always easy and can be time-consuming, which makes them nonapplicable for many applications. For instance, modeling the inflow measurements of wastewater treatment plants is very challenging because of the presence of a large number of variables that are nonlinearly dependent and autocorrelated. Additionally, modeling modern industrial and environmental processes is challenging because of the complexity and the absence of a precise understanding of these processes. The successful detection of faults using model-based approaches can, therefore, be considered a challenging and unsuitable approach. Alternatively, data-based methods are more commonly used.

1.2.2 Knowledge-based methods

The success of modern industrial systems relies on their proper and safe operation. Early detection of anomalies as they emerge in the inspected process is essential for avoiding extensive damage and reducing the downtime needed for reparation [24]. As discussed above, when the information available to understand the process under fault-free operation is insufficient to construct an accurate analytical model, analytical monitoring methods are no longer effective. Knowledge-based methods present an alternative solution to bypass this difficulty. In the following, we use artificial intelligence methods and available historical measurements, which inherently represent the correlation of the process variables, to extract the underlying knowledge and system characteristics. In other words, we utilize process characteristic values, such as variance, magnitude, and state variables, for extracting features under fault-free and faulty conditions based on heuristic and analytical knowledge. Fault detection is then performed in a heuristic manner. Specifically, the actual features from the online data are compared with the obtained under-lying knowledge. Methods that fall in this category include expert systems [25], fuzzy logic, Hazop-digraph (HDG) [5], possible cause and effect graphs (PCEG) [26], neuro-fuzzy based causal analysis, failure mode and effect analysis (FMEA) [27], and Bayesian networks [28]. The major drawback of these techniques is that they are more appropriate for small-scale systems and thus may not be suited to inspect modern systems.

1.2.3 Data-based monitoring methods

Engineering and environmental processes have undoubtedly become far more complex due to advances in technology. Consequently, designing an accurate model for complex, high dimensional and nonlinear systems has also become very challenging, expensive, and time-consuming to develop. Setting simplifications and assumptions on models leads to limits in their capacity to capture

https://t.me/learningnets

certain relevant features and operation modes, and induces a modeling bias that significantly degrades the efficiency of the monitoring system. In the absence of a physics-based process model, data-driven statistical techniques for process monitoring have proved themselves in practice over the past four decades. Indeed, data-based implicit models only require an available process-data resource for process monitoring [5]. Data-based monitoring techniques are mainly based on statistical control charts and machine-learning methods.

Essentially, these monitoring techniques rely on historical data collected from the monitoring system. The system is modeled as a black box with input and output data (Fig. 1.7). At first, a reference empirical model that mimics the nominal behavior of the inspected process is constructed using the fault-free data, and then this model is used for detecting faults in new data. In contrast to model-based methods, only historic process data is required to be available in the data-based fault detection methods, and they are classified into two classes: qualitative and quantitative methods.

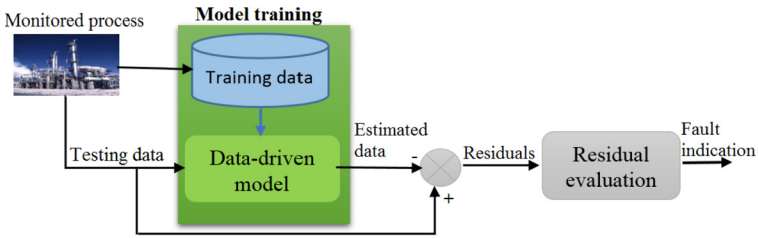


FIGURE 1.7 Data-based methods.

Unsupervised data-based techniques for fault detection and isolation do not use any prior information on faults affecting the process. Unsupervised data-based techniques cover a set of methods for monitoring industrial processes through tools such as statistical control charts (see Fig. 1.8). Univariate techniques, such as a Shewhart chart, exponentially weighted moving average (EWMA) [29], and cumulative sum (CUSUM), are used for monitoring only a single process variable at a given time instant. Monitoring charts have been extensively exploited in most industrial processes. CUSUM and EWMA schemes show good capacity in sensing small changes compared to the Shewhart chart. In [30], a spectral monitoring scheme is designed based on the information embedded in the coefficients of the signal Fourier. However, these conventional schemes are designed based on the hypotheses that the data are Gaussian and uncorrelated. To escape these basic assumptions, multiscale monitoring schemes using wavelets have been developed [31]. Furthermore, the above-discussed schemes use static thresholds computed using the fault-free data. Recently, several adaptive monitoring methods have been developed. These schemes are, in practice, more flexible and efficient than conventional schemes with fixed parameters. For more details, see [32–35]. These univariate monitoring schemes

https://t.me/learningnets

examine a particular variable at a time instant by assuming independence between variables. When monitoring multivariate data using several univariates, even when the number of false alarms of each scheme is small, the collective rate could be very large [36–38]. In addition, measurements from modern industrial processes are rarely independent and present a large number of process variables. Since univariate schemes ignore the cross-correlation between variables, they consequently suffer from an inflated number of undetected issues and false alarms, which makes this monitoring scheme unsuitable [36–38].

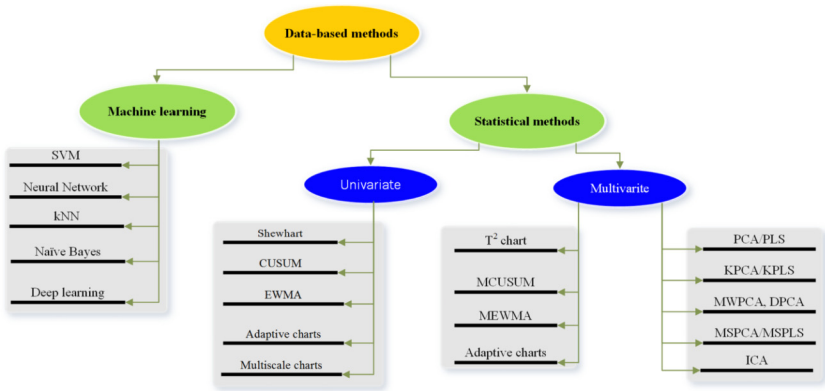


FIGURE 1.8 Data-based monitoring techniques.

To alleviate this difficulty and to handle high dimensional data effectively, multivariate monitoring schemes have been developed to take into account the correlations between the variables, and thus monitor processes with several variables. These schemes include Hotelling T^2 [39], multivariate EWMA [40], and multivariate CUSUM [41]. However, the performance of these multivariate schemes degrades as the number of variables monitored increases, which makes them unsuitable for high dimensional data.

Multivariate monitoring methods for monitoring multivariate data have been designed to directly tackle the above limitations. Multivariate statistical methods are useful for compressing data and retaining relevant information, which is more appropriate to analyze than the original data. Moreover, these methods are efficient at handling noise and interactions between variables to effectively extract pertinent information. The most common multivariate methods for fault detection are principal component analysis (PCA) [22,42], partial least squares (PLS), principal component regression (PCR), canonical variate analysis (CVA), and independent component analysis (ICA) [43]. The essential element of multivariate statistical methods, such as PCA, is their ability to transform multivariate correlated variables to a reduced set of uncorrelated variables. In the past two decades, these techniques have been extensively used to monitor industrial processes. For fault detection purposes, the original data is first projected into a

https://t.me/learningnets

latent subspace, where latent variables and residuals are monitored. PCA and PLS are the two most popular multivariate statistical methods that use latent variable methods for monitoring because they have a strong mathematical foundation that is available in the literature. Indeed, the PCA or PLS model is constructed based on historical normal process operations. This empirical model could be used to monitor the future behavior of the process. Any departure from the model should be flagged as a potential anomaly, such as sensor fault or process drift. PCA is used to reduce dimensionality in the process data and to retain important features of the data. PCA projects the observations from a higher dimension on to a lower-dimensional subspace and is optimal in terms of capturing the data variability. The PCA procedure is applied to a single data matrix only, whereas PLS models the relationship between two data matrices while compressing them simultaneously. The PCA technique is used to monitor and detect the faults in a multivariate process, along with the two fault detection indices, T^2 and the squared prediction error (SPE) statistics. The major advantage of latent variable approaches (i.e., PCA and PLS) is that a limited number of monitoring schemes are needed for monitoring multivariate data using monitoring indices of T^2 and SPE.

However, data from modern industrial processes are time-dependent, nonstationary, nonlinear, non-Gaussian, and multiscale [44–47]. Most process monitoring methods assume that the process measurements at a given time are independent of the observations at a past sampling instant. Industrial processes are operated under dynamic conditions and variables have strong autocorrelation properties. Augmenting observations at a previous sampling time with observations at the present sampling time is referred to as Dynamic PCA (DPCA) [48,49]. For high-dimensional and time-dependent industrial data, using a fixed model monitoring approach could lead to poor diagnostic results [50]. However, process monitoring for such processes could be improved by updating the model using a recursive PCA and a moving window PCA technique [50]. Recursive PCA updates the model continuously online; similarly, online adaptive PCA updates the model using EWMA [50,51]. For nonlinear processes, a nonlinear version of data-based methods has been used, such as kernel PCA, kernel PLS, polynomial PLS and quadratic or fuzzy PLS, to reveal nonlinear relationships between variables [46]. In practice, most of the data need not be Gaussian in nature; to handle the non-Gaussian nature of the data, independent component analysis (ICA), the Gaussian mixture model (GMM), and its nonlinear variant have been used [47]. Other extensions have been developed, such as multiway PCA [45] that permits analyzing data from batch processes, and multiscale PCA that monitors processes at different frequency bands and denoises the data and reduces autocorrelation. Overall, these extensions are introduced based on an understanding of the nature of the data gathered from the inspected process. Accordingly, understanding the process characteristics is a central factor to meet practical expectations and construct an effective statistical monitoring system.

https://t.me/learningnets

Other approaches that fall into this category are based on machine- and deep-learning methods, which have recently gained considerable attention from researchers due to their ability to learn from large and complex datasets. Under a machine-learning framework, support vector machines (SVM) [52–54] and artificial neural networks (ANNs) have become an important tool in fault detection literature. Recently, increasing process complexity has resulted in the development of several monitoring methods based on deep learning that can account for features such as time dependency, nonlinearity, and nonnormality. A major strategy has been to extract features from the data using deep-learning models, such as Restricted Boltzmann Machine (RBM), Deep Belief Network (DBM), Deep Boltzmann Machine (DBM), Long Short-Term Memory (LSTM), and recurrent neural network (RNN), and to monitor the extracted features using binary clustering schemes or traditional monitoring charts. For instance, [55] introduced an approach that integrated an RNN-RBM model with clustering algorithms including k-means, spectral clustering, and OCSVM, for anomaly detection in WWTPs. In [56], several deep learning-based monitoring methods, such as DBN, deep-stacked auto-encoders, and restricted Boltzmann machines-based clustering procedures, were applied to detect abnormal ozone pollution. Deep-learning methods are appealing because of their flexibility to not make restricting assumptions on the underlying data. Also, applications using deep learning cover detection in complex data as multivariate time-series data [57], images and videos [58,59].

1.3 Fault detection metrics

To verify the performance of fault detection methods, several well-know metrics are commonly employed in the context of binary detection problems. Basically, many detection performance metrics are computed based on the 2×2 confusion matrix that reports the number of true positives (TP), false positives (FP), false negatives (FN), and true negatives (TN) [60]. The detection quality of the fault detection methods can be assessed using a false positive rate (FPR) (i.e., false alarm rate), a true positive rate (TPR) (i.e., detection rate), precision, accuracy, F-measure, recall, and the area under the curve (AUC). Fig. 1.9 displays a confusion matrix and recapitulate equations of the well-known related metrics that are frequently used to assess the performance of a binary decision method [60,61].

Also, another metric called average run length (ARL), which is able to characterize both types of error, I and II, is usually used to evaluate detection quality. Specifically, there are two kinds of ARL: ARL0 and ARL1. ARL0 is the average number of data points a fault detection method takes to flag out an alarm when the process is under control. ARL1 refers to the number of data points it takes a monitoring method to uncover a fault under faulty conditions (i.e., speed of detection).

https://t.me/learningnets

The confusion matrix

		True class: Positive	True class: Negative	Row Totals
Predicted class	Positive	*true positive* (tp)	*false positive* (fp)	$PP = tp + fp$
Predicted class	Negative	*false negative* (fn)	*true negative* (tn)	$PN = fn + tn$
Column Totals		$RP = tp + fn$	$RN = fp + tn$	

Performance Metrics

$$TPR = \frac{tp}{RP} \qquad FPR = \frac{fp}{RN}$$

$$Accuracy = \frac{tp+tn}{RP+RN} \qquad Precision = \frac{tp}{PP}$$

$$F_1Score = 2 * \frac{Precision*TPR}{Precision+TPR} \qquad AUC = \frac{TPR-FPR+1}{2}$$

FIGURE 1.9 Fault detection metrics.

1.4 Conclusion

In summary, accurately detecting and isolating faults that can occur in industrial and environmental processes is essential to minimize downtime, increase safety, reduce maintenance costs, and extend equipment lifetime. Process monitoring is required to successfully detect, isolate, and remove the faults before they affect the process performance. Several aspects should be considered when designing or using a particular fault detection approach, including the type of fault, process dynamics, measured variables, available data, and complexity. The simplest and most common practice is to directly check the limit of a measurable variable. However, these techniques are limited when monitoring large-scale processes. This has led to the development of reliable techniques that incorporate information from not just one process variable, but that include more knowledge about the process such as process state and parameters. Some approaches rely on accurate process models whereas others use available historical process data. Process model-based monitoring that incorporates dynamics information is easy to implement for well-defined systems; however, process model-based monitoring needs accurate models that are not always easy to obtain, in particular for complex processes. On the other hand, when information on the reliance of faults and symptoms is available, knowledge-based approaches are preferable; however, these approaches are limited to small and simple processes. An alternative approach is to use data-based monitoring techniques, which are flexible and assumption-free. Of course, when a large amount of process data is available, and the process is too complex to be explicitly modeled, data-based techniques are more appropriate because of their flexibility to handle large, noisy, and nonlinear data.

https://t.me/learningnets

References

[1] V.R. Dhara, R. Dhara, The union carbide disaster in Bhopal: a review of health effects, Archives of Environmental Health: An International Journal 57 (5) (2002) 391–404.

[2] B. Bowonder, The Bhopal accident, Technological Forecasting & Social Change 32 (2) (1987) 169–182.

[3] M.E. Paté-Cornell, Learning from the Piper Alpha accident: a postmortem analysis of technical and organizational factors, Risk Analysis 13 (2) (1993) 215–232.

[4] L.W.D. Cullen, The public inquiry into the Piper Alpha disaster, Drilling Contractor; (United States) 49 (4) (1993).

[5] V. Venkatasubramanian, R. Rengaswamy, S.N. Kavuri, K. Yin, A review of process fault detection and diagnosis: part III: process history based methods, Computers & Chemical Engineering 27 (3) (2003) 327–346.

[6] B. Brooks, The Bakersfield fire: a lesson in ground-fault protection, SolarPro Magazine 62 (2011).

[7] I. Nimmo, Adequately address abnormal operations, Chemical Engineering Progress 91 (9) (1995).

[8] R.J. Patton, Fault-tolerant control: the 1997 situation, IFAC Proceedings Volumes 30 (18) (1997) 1029–1051.

[9] O. Büyüköztürk, M.A. Taşdemir, Nondestructive Testing of Materials and Structures, vol. 6, Springer Science & Business Media, 2012.

[10] R. Isermann, Fault-Diagnosis Systems: an Introduction from Fault Detection to Fault Tolerance, Springer Science & Business Media, 2006.

[11] G.J. Ducard, Fault-Tolerant Flight Control and Guidance Systems: Practical Methods for Small Unmanned Aerial Vehicles, Springer Science & Business Media, 2009.

[12] M. Basseville, I.V. Nikiforov, et al., Detection of Abrupt Changes: Theory and Application, vol. 104, Prentice Hall, Englewood Cliffs, 1993.

[13] F. Harrou, L. Fillatre, I. Nikiforov, Anomaly detection/detectability for a linear model with a bounded nuisance parameter, Annual Reviews in Control 38 (1) (2014) 32–44.

[14] E. Chow, A. Willsky, Analytical redundancy and the design of robust failure detection systems, IEEE Transactions on Automatic Control 29 (7) (1984) 603–614.

[15] P.M. Frank, Fault diagnosis in dynamic systems using analytical and knowledge-based redundancy: a survey and some new results, Automatica 26 (3) (1990) 459–474.

[16] R.J. Patton, J. Chen, A review of parity space approaches to fault diagnosis, IFAC Proceedings Volumes 24 (6) (1991) 65–81.

[17] J. Ragot, D. Maquin, F. Kratz, Analytical redundancy for systems with unknown inputs. Application to faults detection, Control Theory and Advanced Technology 9 (3) (1993) 775–788.

[18] R.N. Clark, D.C. Fosth, V.M. Walton, Detecting instrument malfunctions in control systems, IEEE Transactions on Aerospace and Electronic Systems 4 (1975) 465–473.

[19] R.J. Patton, P.M. Frank, R.N. Clarke, Fault Diagnosis in Dynamic Systems: Theory and Application, Prentice-Hall, Inc., 1989.

[20] K. Benothman, D. Maquin, J. Ragot, M. Benrejeb, Diagnosis of uncertain linear systems: an interval approach, International Journal of Sciences and Techniques of Automatic control & computer engineering 1 (2) (2007) 136–154.

[21] P.M. Frank, Analytical and qualitative model-based fault diagnosis–a survey and some new results, European Journal of Control 2 (1) (1996) 6–28.

[22] L.H. Chiang, E.L. Russell, R.D. Braatz, Fault Detection and Diagnosis in Industrial Systems, Springer Science & Business Media, 2000.

[23] N. Martin, Advanced signal processing and condition monitoring, Insight-Non-Destructive Testing and Condition Monitoring 49 (8) (2007) 459–464.

[24] Z. Gao, C. Cecati, S. Ding, A survey of fault diagnosis and fault-tolerant techniques—part II: fault diagnosis with knowledge-based and hybrid/active-based approaches, IEEE Transactions on Industrial Electronics 62 (6) (2015) 3768–3774.

https://t.me/learningnets

[25] S. Kim, S. jin Ahn, J. Chung, I. Hwang, S. Kim, M. No, S. Sin, A rule based approach to network fault and security diagnosis with agent collaboration, in: International Conference on AI, Simulation, and Planning in High Autonomy Systems, Springer, 2004, pp. 597–606.
[26] N. Wilcox, D. Himmelblau, The possible cause and effect graphs (PCEG) model for fault diagnosis—I. Methodology, Computers & Chemical Engineering 18 (2) (1994) 103–116.
[27] R. Wirth, B. Berthold, A. Krämer, G. Peter, Knowledge-based support of system analysis for the analysis of failure modes and effects, Engineering Applications of Artificial Intelligence 9 (3) (1996) 219–229.
[28] V. Sylvain, T. Teodor, K. Abdessamad, Fault detection with Bayesian network, in: Frontiers in Robotics, Automation and Control, IntechOpen, 2008.
[29] J.M. Lucas, M.S. Saccucci, Exponentially weighted moving average control schemes: properties and enhancements, Technometrics 32 (1) (1990) 1–12.
[30] T. Tiplica, A. Kobi, A. Barreau, Spectral control chart, Quality Engineering 17 (4) (2005) 695–702.
[31] R. Ganesan, T.K. Das, V. Venkataraman, Wavelet-based multiscale statistical process monitoring: a literature review, IIE Transactions 36 (9) (2004) 787–806.
[32] M.S. De Magalhães, E.K. Epprecht, A.F. Costa, Economic design of a Vp $\overline{X}$ chart, International Journal of Production Economics 74 (1–3) (2001) 191–200.
[33] R.B. Kazemzadeh, M. Karbasian, M.A. Babakhani, An EWMA t chart with variable sampling intervals for monitoring the process mean, The International Journal of Advanced Manufacturing Technology 66 (1–4) (2013) 125–139.
[34] D.S. Bai, K. Lee, An economic design of variable sampling interval $\overline{X}$ control charts, International Journal of Production Economics 54 (1) (1998) 57–64.
[35] Y. Su, L. Shu, K.-L. Tsui, Adaptive EWMA procedures for monitoring processes subject to linear drifts, Computational Statistics & Data Analysis 55 (10) (2011) 2819–2829.
[36] J.F. MacGregor, T. Kourti, Statistical process control of multivariate processes, Control Engineering Practice 3 (3) (1995) 403–414.
[37] P. Nomikos, J.F. MacGregor, Multivariate SPC charts for monitoring batch processes, Technometrics 37 (1) (1995) 41–59.
[38] U. Kruger, L. Xie, Advances in Statistical Monitoring of Complex Multivariate Processes: With Applications in Industrial Process Control, Wiley, 2012.
[39] H. Hotteling, Multivariate quality control, illustrated by the air testing of sample bombsights, in: M.W.H.C. Eisenhart, W.A. Wallis (Eds.), Selected Techniques of Statistical Analysis, McGraw-Hill, New York, NY, USA, 1947.
[40] C.A. Lowry, W.H. Woodall, C.W. Champ, S.E. Rigdon, A multivariate exponentially weighted moving average control chart, Technometrics 34 (1) (1992) 46–53.
[41] R.B. Crosier, Multivariate generalizations of cumulative sum quality-control schemes, Technometrics 30 (3) (1988) 291–303.
[42] S. Wold, K. Esbensen, P. Geladi, Principal component analysis, Chemometrics and Intelligent Laboratory Systems 2 (1–3) (1987) 37–52.
[43] A. Hyvärinen, E. Oja, Independent component analysis: algorithms and applications, Neural Networks 13 (4–5) (2000) 411–430.
[44] S.W. Choi, E.B. Martin, A.J. Morris, I.-B. Lee, Adaptive multivariate statistical process control for monitoring time-varying processes, Industrial & Engineering Chemistry Research 45 (9) (2006) 3108–3118.
[45] P. Nomikos, J.F. MacGregor, Monitoring batch processes using multiway principal component analysis, AIChE Journal 40 (8) (1994) 1361–1375.
[46] J.-H. Cho, J.-M. Lee, S.W. Choi, D. Lee, I.-B. Lee, Fault identification for process monitoring using kernel principal component analysis, Chemical Engineering Science 60 (1) (2005) 279–288.
[47] J.-M. Lee, C. Yoo, I.-B. Lee, Statistical process monitoring with independent component analysis, Journal of Process Control 14 (5) (2004) 467–485.

https://t.me/learningnets

[48] W. Ku, R.H. Storer, C. Georgakis, Disturbance detection and isolation by dynamic principal component analysis, Chemometrics and Intelligent Laboratory Systems 30 (1) (1995) 179–196.
[49] K. Chow, K. Tan, H. Tabe, J. Zhang, N. Thornhill, Dynamic principal component analysis using integral transforms, in: AIChE Annual Meeting, Miami Beach, vol. 13, 1999.
[50] M. Kano, K. Nagao, S. Hasebe, I. Hashimoto, H. Ohno, R. Strauss, B.R. Bakshi, Comparison of multivariate statistical process monitoring methods with applications to the Eastman challenge problem, Computers & Chemical Engineering 26 (2) (2002) 161–174.
[51] W. Li, H.H. Yue, S. Valle-Cervantes, S.J. Qin, Recursive PCA for adaptive process monitoring, Journal of Process Control 10 (5) (2000) 471–486.
[52] S. Yin, X. Gao, H.R. Karimi, X. Zhu, Study on support vector machine-based fault detection in Tennessee Eastman process, in: Abstract and Applied Analysis, vol. 2014, Hindawi, 2014.
[53] M. Namdari, H. Jazayeri-Rad, S.-J. Hashemi, Process fault diagnosis using support vector machines with a genetic algorithm based parameter tuning, Journal of Automation and Control 2 (1) (2014) 1–7.
[54] Z.B. Sahri, R.B. Yusof, Support vector machine-based fault diagnosis of power transformer using k nearest-neighbor imputed DGA dataset, Journal of Computer and Communications 2 (09) (2014) 22.
[55] A. Dairi, T. Cheng, F. Harrou, Y. Sun, T. Leiknes, Deep learning approach for sustainable WWTP operation: a case study on data-driven influent conditions monitoring, Sustainable Cities and Society 50 (2019) 101670.
[56] F. Harrou, A. Dairi, Y. Sun, F. Kadri, Detecting abnormal ozone measurements with a deep learning-based strategy, IEEE Sensors Journal 18 (17) (2018) 7222–7232.
[57] P. Malhotra, L. Vig, G. Shroff, P. Agarwal, Long short term memory networks for anomaly detection in time series, in: Proceedings, Presses universitaires de Louvain, 2015, p. 89.
[58] A. Dairi, F. Harrou, M. Senouci, Y. Sun, Unsupervised obstacle detection in driving environments using deep-learning-based stereovision, Robotics and Autonomous Systems 100 (2018) 287–301.
[59] A. Dairi, F. Harrou, Y. Sun, M. Senouci, Obstacle detection for intelligent transportation systems using deep stacked autoencoder and k-nearest neighbor scheme, IEEE Sensors Journal 18 (12) (2018) 5122–5132.
[60] D.M. Powers, Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation, Journal of Machine Learning Technology 2 (2011) 37–63.
[61] D.L. Olson, D. Delen, Advanced Data Mining Techniques, Springer Science & Business Media, 2008.

https://t.me/learningnets

Chapter 2

Linear latent variable regression (LVR)-based process monitoring

2.1 Introduction

With the advancement in instrumentation, data acquisition, and rapid development in the "Internet-of-Things" technology, which connects a large number of digital devices, enormous amounts of information have become available from anywhere at any time, from a multitude of smart devices. Indeed, large datasets are produced by the collection of large number of measurements from modern engineering and environmental processes. Exploiting these measurements with a certain level of redundancy, it becomes feasible to detect abnormal change and locate its sources in the inspected process. However, in the absence of effective tools, the information in these datasets cannot be suitably extracted and exploited for inference and process monitoring.

Over the past decade, the necessity for prediction and fault-detection tools has resulted in the design of several fault-detection mechanisms, which belong to either model-based (or analytical) or data-driven methods [1,2]. Analytical models, based on ideal hypotheses that utilize first principles, could theoretically explain a system's behavior; however, they need prior calibration of model parameters, which is challenging and costly in high-dimensional cases and may result in ill-conditioning problems [3]. Data-driven approaches can perform systematic and objective exploration, visualization, and interpretation of data, can identify essential factors, features, or patterns, and can endorse and optimize data-supported decision-making [4]. Data-based techniques carry information on faults by extracting relevant features from data. Data-driven approaches are more currently commonly applied in engineering and petrochemical processes [5]. For instance, in the petrochemical industry where soft-sensors are widely used, billions of dollars were once lost annually because of the occurrence of faults [6]. Environmental data have been exploited by data-driven approaches for anomaly detection in, for example, meteorological signals [7] or monitoring of sludge bulking in wastewater treatment plants (WWTPs) [8]. For instance, fault detection in chemical process industries is challenging due to the large number of variables involved, the dynamic characteristics and noisy measurements that occur in these processes. Indeed, a large number of variables leads to collinearity, which increases the uncertainty about the model parameter estimates. The latent variable regression (LVR) model is a commonly used

https://doi.org/10.1016/B978-0-12-819365-5.00008-5
Copyright © 2021 Elsevier Inc. All rights reserved.
https://t.me/learningnets

modeling framework to remedy such problems. The LVR model can deal with collinearity among variables, by constructing a model from a reduced number of variables (which are a linear combination of the original variables) called latent variables or principal components. This approach results in well-conditioned models [9,10]. LVR model estimation techniques include principal component regression (PCR) [11,9] and partial least squares (PLS) [12,13].

The organization of this chapter is as follows. In Sect. 2.2, we present a brief introduction to inferential modeling methods, including full rank models and latent variable regression (LVR) techniques. The presented full rank modeling techniques include ordinary least squares (OLS) regression and ridge regression (RR), while the latent variable regression techniques include PCR and PLS. Since the conventional LVR models are static and more appropriate for handling steady-state processes, the dynamic version of the LVR models is also briefly presented. Section 2.3 is devoted to an overview of some common statistical techniques that are applied in statistical process monitoring. Specifically, this section presents the basic univariate monitoring schemes, namely Shewhart, exponentially-weighted moving average (EWMA), cumulative sum (CUSUM), generalized likelihood ratio (GLR), and distribution-based algorithms, and we discuss their limitations. Section 2.4 presents the general framework of fault detection based on LVR approaches. In Sect. 2.5, we discuss one of the commonly used fault isolation approaches, namely contribution plots. We also present an innovative method that uses the radial visualization RadViz to perform root cause diagnosis in Sect. 2.5. The main objective of this chapter is to investigate these multivariate monitoring schemes (PCA and PLS) and their practical applications. In Sect. 2.6, we assess the performances of the developed inferential modeling technique using simulated and practical examples. In addition, we evaluate the method of using PCA-based anomaly detection by importing seven years of influent characteristics (ICs) data from a coastal municipal WWTP where multiple abnormal events occurred. The chapter concludes with a discussion and remarks in Sect. 2.7.

2.2 Development of linear LVR models

Measurements from engineering and industrial processes are usually massive and include a large number of (high-dimensional) variables because of the complexity of the processes involved. Using traditional regression models like least squares are unsuited to provide reliable predictions due to high colinearity and ill-conditioning issues. There are a large variety of estimation techniques to address this modeling problem, including full-rank methods and latent variable regression methods. In this section, we present the basic theoretical perspective of some commonly used linear regression models that are used to design process monitoring algorithms, namely, OLS, RR, PCR, and PLS. In this section, we review the traditional linear correlation models for multivariate data that are the basis for designing fault detection methods. The basic concepts of each approach and discussion on their advantages and weaknesses are presented.

https://t.me/learningnets

2.2.1 Full rank methods

2.2.1.1 Ordinary least squares regression

We regress the data matrix $\mathbf{y} \in \mathbb{R}^n$ (the measured output) to $\mathbf{X} \in \mathbb{R}^{n \times m}$ (selected group of process variables whose values are known precisely) as

$$\mathbf{Y} = \mathbf{X}\beta + \epsilon, \tag{2.1}$$

where $\beta \in \mathbb{R}^n$ is a vector of unknown constants to be estimated, and $\epsilon \in \mathcal{N}(0, \sigma I_n)$ is a zero-mean Gaussian noise with the known variance. The essence of the ordinary least squares (OLS) regression is to estimate the model parameters by minimizing the following objective function [14,11]:

$$\min_{\beta} \left(\|\mathbf{X}\beta - \mathbf{y}\|_2^2 \right). \tag{2.2}$$

The unbiased maximum likelihood estimate of β, if the matrix $\mathbf{X}^T\mathbf{X}$ is nonsingular and the elements of noise ϵ are uncorrelated [15,16], is

$$\widehat{\beta}_{OLS} = \left(\mathbf{X}^T\mathbf{X} \right)^{-1} \mathbf{X}^T\mathbf{y}. \tag{2.3}$$

When the input process variables are highly correlated, the variances of the OLS regression coefficients become very high, and the estimates may be inaccurate. In other words, the determinant of the matrix $\mathbf{X}^T\mathbf{X}$ is then very close to zero, hence giving unstable values for the variance of the estimated regression parameters ($V(b) = \sigma^2(\mathbf{X}^T\mathbf{X})^{-1}$). Moreover, the parameter estimates change considerably if elements of $\mathbf{y}$ are changed slightly and thus $\mathbf{y}$ is poorly predicted when utilized with new $\mathbf{X}$ measurements.

In summary, when $(\mathbf{X}^T\mathbf{X})$ is close to being singular, the variance of $\hat{\beta}$ is inflated, which also results in increasing the uncertainty about its estimation. Even if numerical issues can be surpassed via methods such as pseudo-inverse, the statistical features of the model are not suited to inflated variance. One way to cope with this collinearity problem and the ill-conditioning of $\mathbf{X}$ is through regularization methods, such as ridge regression (RR), which is presented in the following.

2.2.1.2 Ridge regression (RR)

As discussed above, in cases when the input process variables are highly cross-correlated, the OLS method can result in a poor estimate of the regression coefficients. One way to mitigate this problem is to relax the condition that $\widehat{\beta}_{OLS}$ should be an unbiased estimator. There are several methods in the literature to obtain biased estimators of regression coefficients. The RR approach, which was originally introduced by Hoerl and Kennard [17], is commonly used to alleviate the collinearity problem and tuned to obtain good prediction models

https://t.me/learningnets

by trading-off bias and variance. The RR estimator is computed by minimizing the following objective function [17].

$$\min_{\beta} \left(\|\mathbf{X}\beta - \mathbf{y}\|_2^2 + \lambda \|\beta\|_2^2 \right), \tag{2.4}$$

$$\hat{\beta}_{RR} = \left(\mathbf{X}^T\mathbf{X} + \lambda \mathbf{I} \right)^{-1} \mathbf{X}^T\mathbf{y}, \tag{2.5}$$

where λ is a positive constant, and $I \in \mathbb{R}^{m \times m}$ is the identity matrix. Note that from Eq. (2.5), the term $\lambda\mathbf{I}$ added to $(\mathbf{X}^T\mathbf{X})$ enhances the conditioning of the estimation problem. Of course, the RR estimator, $\hat{\beta}_{RR}$, is basically a linear transformation of the OLS estimator $\hat{\beta}_{OLS}$. Eq. (2.5) can be rewritten as

$$\hat{\beta}_{RR} = \left(\mathbf{X}^T\mathbf{X} + \lambda \mathbf{I} \right)^{-1} (\mathbf{X}^T\mathbf{X})\hat{\beta}_{OLS} = \mathbf{Z}_\lambda \hat{\beta}_{OLS}. \tag{2.6}$$

Thus, the RR estimator is a biased estimator since

$$\mathbb{E}(\hat{\beta}_{RR}) = \mathbb{E}(\mathbf{Z}_\lambda \hat{\beta}_{OLS}) = \mathbf{Z}_\lambda \beta. \tag{2.7}$$

The covariance matrix of $\hat{\beta}_{RR}$ is expressed as

$$V(\hat{\beta}_{RR}) = \sigma^2 \left(\mathbf{X}^T\mathbf{X} + \lambda \mathbf{I} \right)^{-1} \mathbf{X}^T \left(\mathbf{X}^T\mathbf{X} + \lambda \mathbf{I} \right)^{-1}. \tag{2.8}$$

The basic concept when using RR is to select a value of λ that guarantees a greater decrease in the variance term than an increase in the squared bias. If this is accomplished, the MSE of $\hat{\beta}_{RR}$ will be less than the variance of $\hat{\beta}_{OLS}$. In [18], it has been demonstrated that there is a positive constant λ for which the MSE $\hat{\beta}_{RR}$ is less than the variance of $\hat{\beta}_{OLS}$.

In practice, various procedures have been developed to choose the value of λ. For instance, in [18] the authors proposed to determine a suitable value of λ by inspecting the ridge trace, which is a plot of elements of $\hat{\beta}_{RR}$ versus λ, where $\lambda \in [0\ 1]$. The aim is to determine a reasonably small value of λ for which the ridge estimates are stable. In [19], an appropriate selection of λ is given as, $\kappa = \frac{m\widehat{\sigma}^2}{\beta_{OLS}^T \beta_{OLS}}$, where β_{OLS} and $\widehat{\sigma}^2$ are determined using a least squares solution.

Of course, these models can be used as an alternative to mitigate the ill-conditioning problem. However, they are not easily interpretable, whereas an important purpose of data modeling is interpretability; see [15,16,18].

2.2.2 Latent variable regression (LVR) models

Multivariate statistical projection methods such as PCA, PCR, and PLS are commonly utilized to handle a high number of highly correlated process variables by conducting regression on a smaller number of transformed variables (i.e., latent

https://t.me/learningnets

or principal components), which are linear combinations of the raw measurements. After computing the latent variables in the process being investigated, these fewer number of variables are then used instead of using the raw data. This latent variables regression (LVR) approach generally results in well-conditioned parameter estimates and reliable model predictions [20]. In this section, these LVR methods are briefly presented. For more details, refer to [21–23]. Before presenting PCR and PLS regression methods, we present PCA, which is a popular multivariate dimensionality-reduction approach.

2.2.2.1 Principal component analysis

Feature extraction with PCA

PCA, a dimensionality-reduction approach, is an increasingly popular modeling framework for discovering relevant and crucial features from multivariate data. The foundation of PCA can be tracked back to Pearson (1901) [24] and Hotelling (1933) [25]. By projecting process variables into a lower-dimensional subspace, PCA reveals the inherent cross-correlation among process variables [26]. In this regard, PCA latent variables or principal components (PCs) (also called scores), which consist of linear combinations of physical variables, can efficiently describe a process in a reduced subspace. PCA-based methods are currently more commonly applied in data compression [27], pattern recognition, data smoothing, classification [28], and fault detection [29].

PCA does not differentiate between input data $\mathbf{X}$ and output data $\mathbf{Y}$. It is applied to one data set that contains all the process variables involved in the problem. Here, $\mathbf{X}$ is used to represent the whole data set. Let $\mathbf{X} = \left[\mathbf{x}_1^T, \ldots, \mathbf{x}_n^T\right]^T \in \mathbb{R}^{n \times m}$ be a dataset gathered from a process having n observations and m variables.

Let us first discuss an important point before going into any further in detail. When performing PCA on multivariate data, it is assumed that all the data are on a comparable scale. If scaling of the data is omitted, then certain variables in the data have to be adjusted to avoid the occurrence of misleading dominance. Scaling of data changes the covariance matrix and consequently affects the principal components. Scaling is important for both the variance and mean adjustments [30]. When the process variables are measured with different units, the purpose of the usual scaling is to make the variance the same (i.e., to give standard units), which gives a correlation matrix. Other variance-stabilizing transformations, such as log transformation, are used in the literature. The most commonly used scaling converts the variables to zero mean and unit variance. Each variable $\mathbf{x}_j \in \mathbb{R}^n$, $j = 1, \ldots, m$, should be scaled to have zero mean and unit variance prior to using PCA:

$$\mathbf{x}_{j,s} = \frac{\mathbf{x}_j - \mu_{x_j}}{\sigma_{x_j}}. \tag{2.9}$$

https://t.me/learningnets

From now on, we consider that autoscaled data is zero-mean centered with unit variance,

$$\mathbf{X} = \begin{pmatrix} x_{1,1} & x_{1,2} & \dots & x_{1,m} \\ x_{2,1} & x_{2,2} & \dots & x_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n,1} & x_{n,2} & \dots & x_{n,m} \end{pmatrix}_{n \times m}.$$

The scaled data $\mathbf{X}$ can be expressed using singular value decomposition (SVD) as a product of two factors:

$$\mathbf{X} = \mathbf{t}_1 \mathbf{w}_1^T + \mathbf{t}_2 \mathbf{w}_2^T + \cdots + \mathbf{t}_m \mathbf{w}_m^T = \mathbf{T}\mathbf{W}^T, \tag{2.10}$$

where $\mathbf{T} \in \mathbb{R}^{n \times m}$ represents a matrix of the principal components (PCs) and $\mathbf{W} \in \mathbb{R}^{m \times m}$ is the loading matrix. The PCs are linear combinations of the original data, and each PC is not correlated with the others. The loading matrix is frequently calculated through SVD of the covariance matrix $\mathbf{S}$ of the data $\mathbf{X}$:

$$\mathbf{S} = \frac{1}{n-1}\mathbf{X}^T\mathbf{X} = W \Lambda W^T \text{ with } WW^T = W^T W = I_n, \tag{2.11}$$

where, $\Lambda = \text{diag}(\sigma_1^2, \dots, \sigma_m^2)$ is a matrix comprising eigenvalues of $\mathbf{S}$ arranged diagonally in decreasing magnitude. The eigenvalues λ_i are equal to the variance of the PC t_i, σ_i^2 (i.e., $\text{var}(\mathbf{w}_i^T \mathbf{X}) = \lambda_i$).

In the presence of cross-correlated multivariate data, $\mathbf{X}$, the first l PCs (where $k < m$) are sufficient for preserving relevant information in the original data. One important step in PCA model development is to select the number of PCs.

Criteria for selecting the number of principal components to use

A core step in designing LVR approaches is selecting by the number of LVs, l, to appropriately extract relevant information from the received data. In other words, the prediction performance of the designed LVR model is influenced by the choice of the number of LVs, l. Accordingly, an appropriate estimation of the number of LVs is necessary to avoid the problem of the model underfitting or overfitting the data. Some of these techniques are briefed below:

- *The scree test.* The scree plot displays the variance caught by every PC against the number of the PCs [31]. Then, the number of PCs to retain are obtained by finding the value of the eigenvalue λ corresponding to the profile with an elbow shape (i.e., the profile is no longer linear). This identification procedure is easy to visualize but it could be not easy for automatic implementation.
- *Parallel analysis.* Parallel analysis compares the variance profile to that obtained by assuming independent variables, to determine the number of PCs. Specifically, l is determined at the point where the two profiles cross [31,32].
- *The cumulative percentage variance (CPV) procedure.* The CPV procedure has been commonly employed to find the number of PCs explaining a certain

https://t.me/learningnets

percentage of the total variance (e.g., 90%) [31]:

$$CPV(l) = \frac{\sum_{i=1}^{l} \lambda_i}{\sum_{i=1}^{m} \lambda_i} \times 100. \tag{2.12}$$

This procedure is attractive since it is intuitive and easy to implement [31].

- *Cross-validation.* Basically, the key concept of the cross-validation mechanism is splitting the data in training datasets for model construction and testing datasets for model validation [33]. The model is verified using the test data, and residuals are generated by comparing the estimated values to the measured values. In the CV approach, the optimum number of PCs is determined by using Predictive Sum of Squares (PRESS) statistics [33],

$$PRESS_l = \sum_{i=1}^{n} (X_i - \hat{X}_i^l)^2, \tag{2.13}$$

where l is the number of PCs vectors retained to calculate $\hat{X}$, i.e., the dimension of the PCs. The dimensionality is determined by finding the number of PCs corresponding to the minimum of the PRESS [33].

Based on the PCA model, after selecting the appropriate number of PCs to include in the model, the data matrix $\mathbf{X}$ can be expressed as a sum of the approximated matrix, $\widehat{\mathbf{X}}$, and residual data, $\mathbf{E}$ (Fig. 2.1),

$$\mathbf{X} = \mathbf{T}\mathbf{W}^T = \sum_{i=1}^{k} t_i w_i^T + \sum_{i=k+1}^{m} t_i w_i^T = \widehat{\mathbf{X}} + \mathbf{E}, \tag{2.14}$$

where $\mathbf{T} \in \mathbb{R}^{n \times m}$ represents a matrix of the principal components (PCs) and $\mathbf{W} \in \mathbb{R}^{m \times m}$ is the loading matrix.

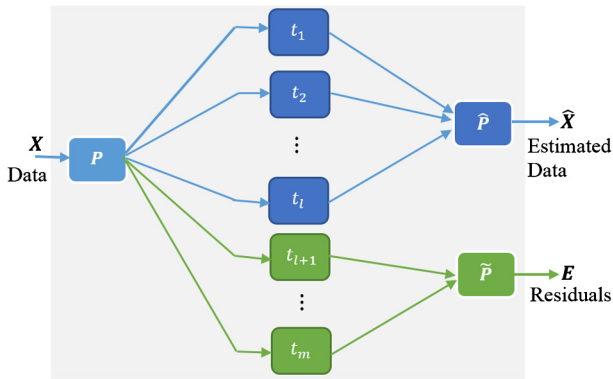


FIGURE 2.1 Schematic representation of PCA model.

https://t.me/learningnets

As described above, the orthogonal eigenvectors of the covariance matrix are equal to the loading matrix $\mathbf{W} = (\mathbf{v}_1, \mathbf{w}_2, \ldots, \mathbf{w}_m)$, and eigenvalue λ_i is the variance of score t_i. The loading matrix can be partitioned into two parts, $\widehat{\mathbf{W}}$ and $\widetilde{\mathbf{W}}$, i.e., $\mathbf{W} = [\widehat{\mathbf{W}} \quad \widetilde{\mathbf{W}}]$. Here $\widetilde{\mathbf{W}} = (\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{W}_l)$, represents the first l principal loading vectors (PCs) and $\widetilde{\mathbf{W}} = (\mathbf{w}_{l+1}, \mathbf{w}_{l+2}, \ldots, \mathbf{W}_m)$ represents the remaining $m - l$ PCs. The partition is shown below:

$$\mathbf{S} = \frac{1}{n-1}\mathbf{X}^T\mathbf{X} = \begin{pmatrix} \widehat{\mathbf{W}} & \widetilde{\mathbf{W}} \end{pmatrix} \begin{pmatrix} \widehat{\Lambda} & 0 \\ 0 & \widetilde{\Lambda} \end{pmatrix} \begin{pmatrix} \widehat{\mathbf{W}}^T \\ \widetilde{\mathbf{W}}^T \end{pmatrix}. \tag{2.15}$$

The data matrix $\mathbf{X}$ can be factorized as

$$\overline{\mathbf{X}} = \overbrace{\left[\widehat{\mathbf{T}} \mid \widetilde{\mathbf{T}}\right]}^{\mathbf{T}} \overbrace{\left[\widehat{\mathbf{W}} \mid \widetilde{\mathbf{W}}\right]^T}^{\mathbf{W}^T} = \widehat{\mathbf{T}}\widehat{\mathbf{W}}^T + \widetilde{\mathbf{T}}\widetilde{\mathbf{W}}^T = \overbrace{\overline{\mathbf{X}}\widehat{\mathbf{W}}\widehat{\mathbf{W}}^T}^{\widehat{\mathbf{X}}} + \overbrace{\overline{\mathbf{X}}\left(\mathbf{I}_m - \widehat{\mathbf{W}}\widehat{\mathbf{W}}^T\right)}^{\mathbf{E}}. \tag{2.16}$$

Here $\widehat{\mathbf{T}} \in \mathbb{R}^{n \times l}$ is the PC matrix $(n \times l)$, which describes the values of variables in the transformed $(n \times l)$ basic space spanned by $\widehat{\mathbf{W}}$, while l is chosen to capture most of the variability in the data, and no relevant information is lost in $\mathbf{E}$. The matrices $\widehat{\mathbf{W}}\widehat{\mathbf{W}}^T$ and $(\mathbf{I}_m - \widehat{\mathbf{W}}\widehat{\mathbf{W}}^T)$ span the principal component and residual subspaces, respectively. The row vectors in $\overline{\mathbf{X}}$ and $\mathbf{E}$ are orthogonal, i.e., $\overline{\mathbf{X}}^T\mathbf{E} = 0$.

The residual matrix plays a core role in uncovering abnormal features in process monitoring. For the purpose of anomaly detection, we will evaluate the generated residuals based on the developed PCA reference model by univariate or multivariate statistical monitoring schemes. More details on process monitoring are given in the subsequent sections.

2.2.2.2 Principal component regression

PCR is an alternative to OLS regression for addressing the issue of ill-conditioning or collinearity in multivariate linear regression, which results in a poor estimation of the model parameters. PCR is a linear regression approach that can handle highly correlated process variables by latent variables as regressors in the regression. It can be implemented in two steps. The first step in PCR consists of projecting the input variables via PCA to account for collinearity and reduce their dimensions. To this end, SVD is frequently employed to compute the PCs. In the second step, OLS regression is conducted between the retained PCs and the response [14,11] (Fig. 2.2).

To sum up, the key idea of PCR is to use uncorrelated l score vectors from the PCA instead of the l columns in $\mathbf{X}$. Specifically, the multicollinearity among the predictor variables can be eliminated by using a subset of orthogonal PCs from the input data $\mathbf{X}$ via PCA. Then, OLS is performed between the response variable $\mathbf{Y}$ and the retained l PCs of $\mathbf{X}$. From the PCA model, the matrix $\mathbf{X}$ can

https://t.me/learningnets

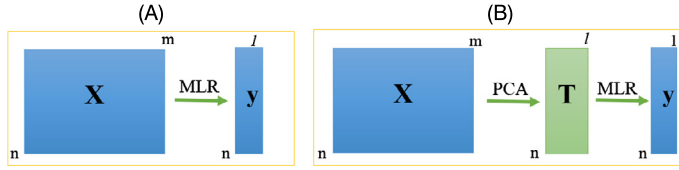


FIGURE 2.2 Schematic representation of (A) MLR and (B) PCR models.

be decomposed as follows:

$$\mathbf{X} = \mathbf{T}\mathbf{W}^T = \sum_{i=1}^{l} t_i w_i^T + \sum_{i=l+1}^{m} t_i w_i^T = \widehat{\mathbf{X}} + \mathbf{E}, \tag{2.17}$$

where $\mathbf{T} \in \mathbb{R}^{n \times m}$ represent a matrix of the PCs and $\mathbf{W} \in \mathbb{R}^{m \times m}$ is the loading matrix. Then, a subset of these PCs (with the largest variance) are utilized to build a linear model relating these PCs to the response variable, $\mathbf{y}$, using OLS regression,

$$\mathbf{y} = \widehat{\mathbf{T}}\hat{\beta}, \tag{2.18}$$

where $\widehat{\mathbf{T}} = [\mathbf{t}_1 \dots \mathbf{t}_l]$ is the retained PCs (with the largest eigenvalues) used to construct the model, with $l \leq m$; l is selected such that there is no important loss in process information retained in residuals. The regression matrix $\hat{\beta}$ is obtained by solving the following minimization problem:

$$\min_{\beta} \left(\|\widehat{\mathbf{T}}\beta - \mathbf{y}\|_2^2 \right), \tag{2.19}$$

$$\hat{\beta} = \left(\widehat{\mathbf{T}}^T \widehat{\mathbf{T}} \right)^{-1} \widehat{\mathbf{T}}^T \mathbf{y}. \tag{2.20}$$

Note that PCR is equivalent to OLS if all PCs are used in designing the PCR model (i.e., $l = m$). In the case of uncorrelated input variables, OLS would be the first option in regression. Note that all PCs in PCR are determined without taking the model response into consideration. Next we present another approach to cope with the multicollinearity problem, which takes into account the input–output relationship when determining the PCs, called partial least squares (PLS).

2.2.2.3 *Partial least squares*

This section introduces the PLS regression modeling (also known as the projection on latent structures), which was first proposed in [34] in the field of econometrics. Later in [35] a detailed PLS algorithm was provided. In [36], the geometry of two procedures to perform PLS has been illustrated. This technique is used in chemometrics and chemical engineering for soft sensor development [37], process monitoring, and fault diagnosis.

The capacity of PLS to deal with multivariate input–output data with collinearity is one of its desirable characteristics [38]. When the matrix $\mathbf{X}^T\mathbf{X}$

https://t.me/learningnets

is singular or ill-conditioned, PLS determines an optimum pair of LVs in the input and output data (**X** and **Y**) so that these transformed variables have the largest covariance. Unlike PCR, PLSR exploits the information in input and output variables by using the covariance between them and reducing the impact of irrelevant variations of input variables. In other words, PLSR is designed using both PCs of **X** and **Y**. Basically, the PLS model is performed by searching a set of PCs that explains the maximum cross-correlation between **X** and **Y** (Fig. 2.3).

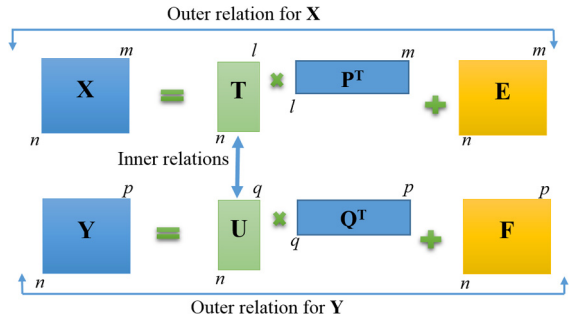


FIGURE 2.3 Schematic representation of PLS model.

Consider an input with n samples and m variables, $\mathbf{X} \in \mathbb{R}^{n \times m}$, and output with n samples and p variables, $\mathbf{Y} \in \mathbb{R}^{n \times p}$. PLS extracts the principal components iteratively by maximizing the covariance of the extracted principal components. PLS model development has two components, one is to develop inner models and the other is to develop outer models [39,40]. Outer models have a relationship with the inner model such that

$$\begin{cases} \mathbf{X} = \sum_{i=1}^{l} \mathbf{t}\mathbf{p}_i^T = \mathbf{T}\mathbf{P}^T + \mathbf{G}, \\ \mathbf{Y} = \sum_{i=1}^{l} \mathbf{u}\mathbf{q}_i^T = \mathbf{U}\mathbf{Q}^T + \mathbf{F}, \end{cases} \tag{2.21}$$

where $\mathbf{T} \in \mathbb{R}^{n \times l}$ and $\mathbf{U} \in \mathbb{R}^{n \times q}$ are matrices of the transformed uncorrelated variables. The loading matrices of input and output space are $\mathbf{P} \in \mathbb{R}^{m \times l}$ and $\mathbf{Q} \in \mathbb{R}^{p \times q}$, respectively. The model residuals are **G** and **F**. The number of PCs, l, is determined by cross-validation.

The retained latent variables of the input and output space are related by the linear inner model as

$$\mathbf{U} = \mathbf{T}\mathbf{B} + \mathbf{H}, \tag{2.22}$$

where **B** is a regression matrix linking the input and response PCs, and **H** is a residual matrix. The regression coefficients of **B** can be obtained by minimization of residuals **H**. The response **Y** is given as

$$\mathbf{Y} = \mathbf{T}\mathbf{B}\mathbf{Q}^T + \mathbf{F}^*. \tag{2.23}$$

https://t.me/learningnets

Notice that each pair of latent variables in the PLS model (i.e., t_j and u_j ($j = 1, \ldots, l$)) is estimated iteratively [35,41]. Various procedures are developed in the literature to obtain PLS estimators, including nonlinear iterative partial least squares (NIPALS) and SIMPLS methods. For more details, refer to [35,36, 34,12].

The first pair of latent variable vectors is calculated so that the following covariance:

$$\arg\max_{\mathbf{p}_i, \mathbf{q}_i} \ \mathrm{cov}(\mathbf{t}_1, \mathbf{u}_1) = \mathbf{t}_1^T \mathbf{u}_1 = \mathbf{p}_1^T \mathbf{X}^T \mathbf{Y} \mathbf{q}_1 \tag{2.24}$$

can be maximized with constraints $\mathbf{p}_i^T \ \mathbf{p}_i = 1$ and $\mathbf{q}_i^T \ \mathbf{q}_i = 1$.

The first pair (p_1, q_1) of loading vectors, which represents the dominant direction, is computed so that the covariance between $\mathbf{X}$ and $\mathbf{Y}$ is maximized. Then, the first set of latent variable vectors ($t_1 = \mathbf{X}p_1; u_1 = \mathbf{Y}q_1$) is obtained by projecting $\mathbf{X}$ data on p_1 and Y data on q_1 (the outer model). After that, the inner model can be established between t_1 and u_1 ($\widehat{u}_1 = t_1 b_1$).

After the first set of scores and loadings are computed, the residuals of the input and output variables are calculated as

$$\begin{cases} \mathbf{E_1} = \mathbf{X} - t_1 p_1, \\ \mathbf{F_1} = \mathbf{Y} - u_1 q_1 = \mathbf{Y} - t_1 b_1 q_1. \end{cases} \tag{2.25}$$

Overall, PLS iteratively estimates both LVs for $\mathbf{X}$ and $\mathbf{Y}$, so that they have maximal covariance. These pairs of LVs are estimated and added to the model in an iterative way. The input and output residuals are generated and the procedure is iterated based on the residual until cross-validation error is minimized [11,35, 34,14]. Fig. 2.4 illustrates the recursive process of determining the LVs in PLS.

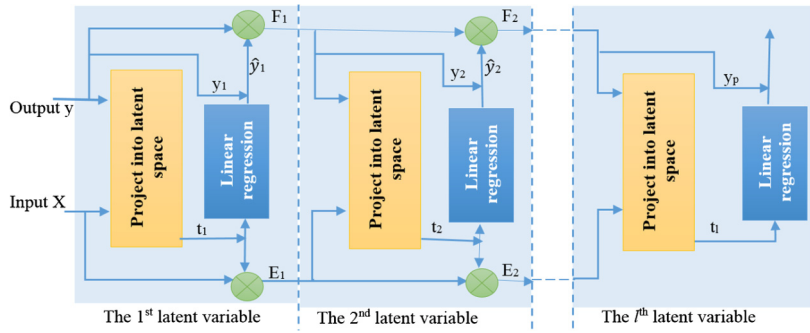


FIGURE 2.4 Schematic representation of the recursive procedure to determine the PCs in PLS.

The NIPALS algorithm, which is commonly used to derive PLS models, is summarized below [42]:

Step 1. Set data $\mathbf{X}$ and $\mathbf{Y}$ to have mean zero and unit variance
Step 2. Set $\mathbf{u}$ equal to a column of $\mathbf{Y}$

https://t.me/learningnets

Step 3. Let $\mathbf{w} = \frac{\mathbf{u}^T\mathbf{X}}{\mathbf{u}^T\mathbf{u}}$
Step 4. Normalize $\mathbf{u}$ to have unit length
Step 5. Evaluate the scores, $\mathbf{t} = \frac{\mathbf{Xw}}{\mathbf{w}^T\mathbf{w}}$
Step 6. Evaluate the new $\mathbf{u}$ vector, $\mathbf{u} = \frac{\mathbf{Yq}}{\mathbf{q}^T\mathbf{q}}$
Step 7. Check convergence on $\mathbf{u}$: if YES go to Step 8, if NO go to Step 2
Step 8. Evaluate $\mathbf{X}$ loading, $\mathbf{p} = \frac{\mathbf{X}^T\mathbf{t}}{\mathbf{t}^T\mathbf{t}}$
Step 9. Evaluate the residual matrices, $\mathbf{E} = \mathbf{X} - \mathbf{tp}^T$ and $\mathbf{F} = \mathbf{Y} - \mathbf{tq}^T$
Step 10. If additional PLS dimensions are necessary then replace $\mathbf{X}$ and $\mathbf{Y}$ by $\mathbf{E}$ and $\mathbf{F}$, respectively, and repeat Steps 1 through 9.

Since PLS is using a covariance objective function, it frequently needs multiple LVs even in the case of a single output in $\mathbf{Y}$. However, sometimes an important part of the LV subspace is orthogonal or irrelevant to the output, despite the fact that the subspace includes large variability of the input data [43]. Thus, to further improve PLS, numerous extensions have been developed such as orthogonalized PLS [44] and concurrent PLS approaches [45].

Note that the above described LVR methods all exploit the latent structured relationships between the process variables that are linear and static. They establish the basic framework for further enhancements to nonlinear or dynamic LVR modeling.

2.3 Dynamic LVR models

From the above discussion, we have shown that LVR models such as PLS and PCR can be used to handle multivariate data with collinearity among the variables by designing a model from a reduced number of variables (which are a linear combination of the original variables) termed latent variables. These methods result in well-conditioned models. However, LVR models are static and ignore process dynamics, which make them unsuitable to catch the temporal evolution of data. In other words, the use of such methods to select the key variables is performed by assuming that the variables are uncorrelated in time. Since many practical data produced from engineering and environmental processes are correlated in time, it is necessary to have a model incorporating such information to deal with process dynamics.

For dynamic processes such as engineering and chemical processes, frequently the actual observations of the process variable depend on past observations. The application of static LVR approaches (e.g., PLS and PCA) to dynamic data will not give accurate modeling of the relations among the variables, but just a linear static approximation. To remedy this limitation and consider the dynamic information, an augmented process dataset, including previously autocorrelated measurements, should be created. A commonly used approach to bypass such limitations is dynamic PCA (DPCA), which has been introduced in [46]. Basically, DPCA is the conventional PCA applied to augmented data including time-lagged measurements of process variables. Specifically, to de-

https://t.me/learningnets

scribe the temporal dynamics, the Hankel matrix of the original data, which is usually employed in time series modeling, is used in [46]. The augmented data that includes time-lagged variables at time instance k is

$$\begin{aligned}\mathbf{X}_z &= [\mathbf{X}(k)\ \mathbf{X}(k-1)\dots\ \mathbf{X}(k-z)] \\ &= \begin{bmatrix} x^T(0) & x^T(1) & \dots & x^T(z) \\ x^T(1) & x^T(2) & \dots & x^T(z+1) \\ \vdots & \vdots & \ddots & \vdots \\ x^T(n-z) & x^T(n-z+1) & \dots & x^T(n) \end{bmatrix},\end{aligned} \tag{2.26}$$

where z is the time lags and its length is related to the past memory entered in the variables.

The DPCA is applied to the augmented process data matrix in a similar way to conventional PCA [46]. Indeed, this is basically the same as the static PCA except that the input data is augmented to include past measurements. The selection of the appropriate number of lagged data plays a key role in DPCA to appropriately model the process dynamics. For highly nonlinear data, the number of lags, z, to incorporate in the data may take a higher value to achieve better linear approximation. DPCA modeling can be outlined in the following steps:

(1) Start with $z = 0$
(2) Compute the augmented data matrix $\mathbf{X}_z$
(3) Design PCA model using the augmented data
(4) Select the optimal PCs to be kept in the model using some known criteria such as Cumulative Percent Variance (CPV) approach
(5) Check the autocorrelation function (ACF) of the residuals of the PCA model
(6) If ACF is within the threshold, i.e., the residuals are not correlated, go to Step (8), otherwise, proceed
(7) Increment the number of lags $z = z + 1$ and go to Step (2)
(8) End

The essence of DPCA is to apply PCA using time-lagged data, thus both the linear static and dynamic relationships among process variables are captured. To sum up, DPCA exploits both the desirable characteristics of PCA to high-dimensional data and the flexibility of the time series model, Autoregressive Integrated Moving Average (ARIMA), to capture the time dependency in data [47,48].

On the other hand, several approaches are designed in the literature to handle dynamics in multivariate input–output processes based on dynamic PLS. One approach consists of incorporating a large number of time-lagged input measurements in the input data matrix $\mathbf{X}$, which conducts to a PLS-Finite Impulse Response (FIR) model [49]. Analogous to DPCA, both the time-lagged data of the input and output process variables are included in the input data matrix $\mathbf{X}$, which results in the PLS-Autoregressive Moving Average (ARMA) model. Both

https://t.me/learningnets

PLS-FIR and PLS-ARMA models need a large augmentation in the dimension of the input data matrix $\mathbf{X}$, which may be cumbersome to handle. To remedy this difficulty, in [50] a simple and flexible method is presented permitting the inclusion of the process dynamics as part of inner PLS model and avoiding the consideration of significant time-lagged input and output variables in the input data. The key benefit of this approach is that no lagged variables are used in the PLS outer model. In [51], a dynamic Autoregressive with Exogenous Terms (ARX) or Hammerstein model is used to account for process dynamics in PLS, for inner relation between t_i and u_i instead of a static model.

The aforementioned LVR methods are all extensively used for multivariate process monitoring. To do so, these LVR methods are combined with fault detection indices such as the Hotelling T^2 and the squared prediction error schemes. The general framework of LVR-based process monitoring strategies is presented in Sect. 2.5.

2.4 Process monitoring methods

Detecting anomalies in industrial processes plays a core role in developing efficient production systems that have acceptable performance and meet the desired requirements and specifications. Without an efficient detection procedure, chemical processes such as distillation columns would be damaged by unexpected faults and could result in financial losses and serious damages. Univariate statistical monitoring schemes are widely applied in numerous production processes as tools for checking product quality when the inspected variable is univariate. The goal of statistical process monitoring schemes is to uncover any deviation of the supervised process from the desired performance. For many decades, these univariate schemes were frequently applied in quality control applications, and now they have been extended to many other fields, such as air quality [29], cybersecurity [52], healthcare systems [53,54], and economics [53]. In this section, we describe the essence of some basic univariate monitoring schemes, such as Shewhart, CUSUM, EWMA, and GLR charts.

2.4.1 Univariate chart for process monitoring

In this subsection, we summarize univariate process monitoring charts including Shewhart, CUSUM, EWMA, and GLR.

2.4.1.1 Shewhart-based monitoring scheme

Shewhart introduced the Shewhart monitoring scheme in 1931 to supervise the product quality at different phases of a manufacturing process [55]. In practice, this monitoring chart is one of the most frequently applied statistical quality control schemes [55]. Instead of waiting to examine the quality of the final product, early inspection and monitoring would enable companies save costs with regards to inspection and rejection of the finished product [55]. This would help

https://t.me/learningnets

ensure that uniform quality of products is maintained, thus leading to increased economic benefits and improved time efficiency. Statistical decisions in Shewhart schemes are based on current observations and no memory about the past is considered. Thus, they are suitable for detecting relatively large faults. The Shewhart chart is used online to evaluate the process performance based on the current measured data.

Consider that $(x_1, x_2, \ldots, x_n)$ are individual observations received from the supervised process. Shewhart schemes are designed under the assumption that the measurements are uncorrelated and the data under normal operating conditions are normally distributed [55]. If these two assumptions are verified, the control limits of the Shewhart chart are defined as [55,56]

$$UCL, LCL = \mu_0 \pm z_{1-\frac{\alpha}{2}}\sigma_0, \tag{2.27}$$

UCL and LCL denote the upper control limit (UCL) and the lower control limit (LCL) while $z_{1-\frac{\alpha}{2}}$ is the $(1-\frac{\alpha}{2})$th quartile of the Gaussian distribution $\mathcal{N}(0, 1)$. Also μ_0 and σ_0 represent the mean and standard deviation of the measurements without anomalies. The term $z_{1-\frac{\alpha}{2}}$ is usually called the width of the control limits and it is generally fixed to be 3, which is equivalent to a false alarm rate of 0.27%. The Shewhart scheme flags a fault if

$$x_t < LCL \text{ or } x_t > UCL. \tag{2.28}$$

In summary, the performance of the Shewhart charts is limited when utilized to sense small changes in the process mean. They consider only the current measurement of the process, thus they are classified as detection charts without memory. To tackle this deficiency, improved mechanisms with increased process memory would be very helpful. Memory-type monitoring approaches, such as CUSUM, moving average, and EWMA charts, are designed to detect small changes.

2.4.1.2 Cumulative sum (CUSUM)-based monitoring schemes

Cumulative sum (CUSUM) monitoring schemes are well reputed in fault detection and were first introduced by Page [57]. Compared to Shewhart-type approaches, the CUSUM schemes are a suitable alternative for detecting small changes, which are often a major concern in process monitoring [57]. Instead of using only the current measurement, the CUSUM scheme exploits all the available information from previous and current measurements to uncover faults. The CUSUM statistic (S_i) is determined as [58]

$$S_i = \sum_{j=1}^{i}(x_j - \mu_0), \tag{2.29}$$

where S_i is the cumulative sum of all available measurements including the current and previously received measurements, and μ_0 is the fault-free process

https://t.me/learningnets

mean. The CUSUM decision function is obtained in a recursive manner as [58]

$$S_i = (x_i - \mu_0) + S_{i-1}. \tag{2.30}$$

One-sided CUSUM statistic is calculated as follows [58]:

$$S_i = \sum_{j=1}^{i} \left[x_j - (\mu_0 + k)\right], \tag{2.31}$$

where k is a parameter that is employed as a reference for detecting a change in the process mean. If S_t changes into a negative value, the CUSUM decision statistic is automatically set to zero. A fault is flagged out when the CUSUM statistic S_t overpasses the decision threshold, H. In practice, the threshold H of 4σ or 5σ, which results in good detection of a deviation of about 1σ in the process mean, is recommended [59]. Here σ is the standard deviation of the monitored variable.

Numerous variations of the CUSUM exist; one of the most common forms is the two-sided CUSUM (tabular) [56]. The recursive formula for high and low side shifts are:

$$S_t^+ = \max\left[0, x_t - (\mu_0 + k) + S_{t-1}^+\right], \tag{2.32}$$

$$S_t^- += \max\left[0, (\mu_0 - k) - x_t + S_{t-1}^-\right], \tag{2.33}$$

where the statistics S^+ and S^- are respectively the upper and lower one-sided CUSUMs, and $S_0^+ = S_0^- = 0, \mu_0$. A fault is declared if either S_t^- or S_t^+ exceeds the decision threshold $H = h\sigma$, where h relies on the shift to be detected.

2.4.1.3 *Exponentially weighted moving average (EWMA) schemes*

While CUSUM schemes consider all available measurements with equal weight in process monitoring, EWMA schemes exponentially weight the measurements based on their importance in characterizing the process [60]. The EWMA shows suitable performance in detecting small changes in the process mean. The EWMA scheme was first designed by Roberts [61], and was frequently applied in quality control and process monitoring [56]. The EWMA monitoring statistic is defined as follows:

$$\begin{cases} z_0 = \mu_0, \\ z_t = \gamma x_t + \left(1 - \gamma\right) z_{t-1}, \end{cases} \tag{2.34}$$

where $z_0 = \mu_0$ is the mean of fault-free data, γ is a weighting factor with the range $0 < \gamma \leq 1$, which defines the temporal memory of the EWMA scheme. Eq. (2.34) indicates that the EWMA statistic utilizes all the available information to sense small anomalies. To highlight this point, the EWMA statistic, z_t,

https://t.me/learningnets

can be expressed recursively as:

$$\begin{aligned} z_t &= \gamma z_t + (1-\gamma)\overbrace{\left[\gamma z_{t-1} + (1-\gamma) z_{t-2}\right]}^{z_{t-1}} \\ &= \gamma z_t + \gamma(1-\gamma) z_{t-1} + (1-\gamma)^2 z_{t-2}. \\ &= \gamma z_n + \gamma(1-\gamma) z_{n-1} + \gamma(1-\gamma)^2 z_{n-2} + \cdots \\ &\quad + \gamma(1-\gamma)^{n-1} z_1 + (1-\gamma)^n z_0. \end{aligned} \tag{2.35}$$

The EWMA decision function in (2.35) can be expressed in compact form as

$$z_t = \gamma \sum_{t=1}^{n} (1-\gamma)^{n-t} z_t + (1-\gamma)^n z_0, \tag{2.36}$$

where $\gamma(1-\gamma)^{n-t}$ denotes the weight for z_t, which exponentially decreases for previous observations. The parameters L and γ play an important role in designing the EWMA scheme [56,54]. The value of L is frequently fixed in practice to be 3, which implies a false alarm rate of 0.27%. Generally, a choice of small values of γ (i.e., where less importance is placed on the newer observations) is used in order to extend the sensitivity to small deviations, while the use of large values of γ (i.e., EWMA with short memory) is suited for detecting larger changes in the process mean [56,62,56]. For the purpose of detecting small changes, in practice the value of γ is usually selected in the interval $[0.1, 0.3]$ [62,56].

In the absence of anomalies, the distribution of the EWMA statistic is $z_t \sim \mathcal{N}(\mu_0, \sigma_{z_t}^2)$, where $\sigma_{z_t} = \sigma_0 \sqrt{\frac{\gamma}{(2-\gamma)}[1-(1-\gamma)^{2t}]}$ and σ_0 represents the standard deviation of the fault-free measurements. However, when a mean shift occurs at the time $1 \leq \tau \leq n$, the distribution of the EWMA statistic is computed as $z_t \sim \mathcal{N}(\mu_0 + \left[1-(1-\gamma)^{n-\tau+1}\right](\mu_1 - \mu_0), \sigma_{z_t}^2)$. Accordingly, when faults happen, the mean of the EWMA decision function, z_t, is a weighted average of μ_0 and μ_1, and the weight related to μ_1 becomes large when n is large. Then, this clearly highlights that the statistic z_t provides pertinent information about the mean shift. The EWMA scheme flags faults when the monitoring statistic z_t, as given in (2.34), exceeds the upper and lower control limits defined as

$$\begin{cases} UCL = \mu_0 + L\sigma_{z_t}, \\ LCL = \mu_0 - L\sigma_{z_t}, \end{cases} \tag{2.37}$$

where μ_0 is the targeted mean, L is the width of the control limit, and σ is the standard deviation of the fault-free or preliminary data set.

From σ_{z_t}, it can be seen that when t becomes larger, the term $[1-(1-\gamma)^{2t}]$ is asymptotically equivalent to unity. In other words, the control limits attain

https://t.me/learningnets

their steady-state values [56]:

$$\begin{cases} UCL = \mu_0 + L\overbrace{\sigma_0\sqrt{\frac{\gamma}{(2-\gamma)}}}^{\sigma}, \\ LCL = \mu_0 - L\overbrace{\sigma_0\sqrt{\frac{\gamma}{(2-\gamma)}}}^{\sigma}. \end{cases} \tag{2.38}$$

As described previously, in the Shewhart schemes, anomaly detection is based only on the current measurement and all past measurements are ignored (Fig. 2.5). Accordingly, these schemes provide unsatisfactory monitoring results when used for sensing small changes in the process mean. This limitation can be mitigated by incorporating the information from the actual and the past measurements in the decision process such as in EWMA and CUSUM schemes (Fig. 2.5). In the CUSUM scheme, information from all available measurements are exploited and the same weight is assigned to all observations (Fig. 2.5). On the other hand, the EWMA scheme, which is designed by using an exponentially weighted average of all available measurements, is also sensitive in detecting small changes in the process mean.

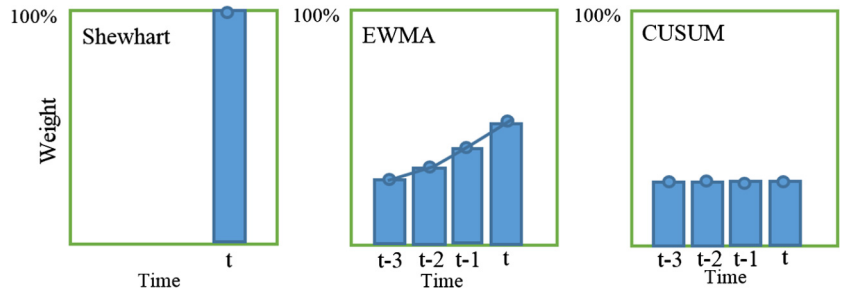


FIGURE 2.5 Univariate process monitoring charts.

In EWMA schemes, a larger value of the smoothing parameter is suited to rapidly detect faults with a large amplitude, while a smaller value can efficiently detect small faults in the mean of the process [60]. Therefore, by using a unique value for the smoothing parameter, monitoring-based EWMA schemes cannot reach a good detection capacity for both small and large faults simultaneously [60]. Since the univariate EWMA control schemes assume fixed thresholds, which may not be suitable for dealing with nonstationary (or time-varying) data. Therefore, several adaptive EWMA and CUSUM methods have been designed in the literature by allowing the thresholds of these methods to vary online to account for the changing nature of the data [63,64]. The idea behind the adaptive EWMA is to adapt the weight of the past observations, according to the magnitude of the error ($e_t = x_t - z_{t-1}$, see (2.39)), and to detect

https://t.me/learningnets

in a more balanced way faults with different sizes:

$$\begin{aligned} z_t &= \gamma x_t + (1-\gamma) z_{t-1} \\ &= \gamma \overbrace{(x_t - z_{t-1})}^{e_t} + z_{t-1}. \end{aligned} \tag{2.39}$$

Also, several adaptive CUSUM (ACUSUM) schemes have been developed in the literature to achieve suitable detection performance covering a range of mean change magnitudes [64,65]. For instance, the basic idea behind the ACUSUM proposed in [64] is to update the reference value (K) in CUSUM based on the EWMA estimate.

2.4.1.4 Generalized likelihood ratio (GLR) hypothesis testing approach

The above-described monitoring schemes (i.e., Shewhart, CUSUM, and EWMA) are more or less suited to some specific range of fault amplitudes. For instance, Shewhart-type approaches provide satisfactory detection of large faults, but they are insensitive to small changes in the process mean [54,59]. While CUSUM and EWMA schemes are effective in detecting small changes, they fail to detect large faults. However, in practice, the magnitude of occurring faults is unknown. Accordingly, it is desirable to automatically detect a large range of faults and thus reduce the rate of missed detection. To this end, one approach to achieve a reliable detection of different sizes of process anomalies is to base the monitoring scheme on a generalized likelihood ratio test (usually called GLR charts) [66]. The benefits of the GLR approach are its efficiency in separating composite hypotheses, simplicity, and absence of complex computations. Extensive literature has been dedicated to studying GLR properties. Signficant efforts have been devoted to establishing different asymptotic optimality properties of this hypothesis testing approach and can found in [67–71]. The GLR detector is widely used in several applications including air quality monitoring [29] and train safety [66].

Here, we consider problems related to binary composite hypothesis testing. When testing two composite hypotheses in which their corresponding data probability density functions (PDFs) comprise unknown parameters, the GLR approach is commonly utilized for separating the two possibilities. The null hypothesis generally defines the nominal operating situation, while the alternatives characterize departures whose presence should be either confirmed or discarded. The essence of the GLR approach is to maximize the likelihood ratio statistic over all possible faults to decide between two composite hypotheses [68–71]. In other words, the aim of the GLR approach is to separate two composite hypotheses, $\mathcal{H}_0$ and $\mathcal{H}_1$, based on the observed data.

For the purpose of anomaly detection, let's consider an observation vector $Y = [y_1, y_2, \ldots, y_n] \in \mathbb{R}^n$ being generated by one of these Gaussian distribu-

https://t.me/learningnets

tions:

$$\begin{cases} \mathcal{H}_0: \ Y \sim \mathcal{N}(0, \sigma^2 I_n), \\ \mathcal{H}_1: \ Y \sim \mathcal{N}(\theta \neq 0, \sigma^2 I_n), \end{cases} \tag{2.40}$$

where θ is the value of the anomaly and $\sigma^2 > 0$ is the variance. In this chapter, the null hypothesis, $\mathcal{H}_0$, represents the fault-free situation, and the alternative hypothesis, $\mathcal{H}_1$, represents the situation with potential faults. Generally speaking, to decide between the two hypotheses, the GLR approach compares the decision statistic, $\mathcal{L}(Y)$, to the control limit, $h(\alpha)$:

$$\delta(Y) = \begin{cases} \mathcal{H}_0 \ \text{if} \ \mathcal{L}(Y) = 2\log \frac{\sup_{\theta \in \mathbb{R}^n} f_\theta(Y)}{f_{\theta=0}(Y)} < h(\alpha), \\ \mathcal{H}_1 \ \text{else.} \end{cases} \tag{2.41}$$

The GLR charting statistic, $\mathcal{L}(Y)$, is given as

$$\mathcal{L}(Y) = 2\log \sup_\theta \left\{ \exp\left\{ -\frac{\|Y-\theta\|_2^2}{2\sigma^2} \right\} / \exp\left\{ -\frac{\|Y\|_2^2}{2\sigma^2} \right\} \right\}, \tag{2.42}$$

where $\|\cdot\|_2$ is the Euclidean norm and $f_\theta(Y) = \frac{1}{(2\pi)^{\frac{n}{2}}\sigma^n} \exp\left\{ -\frac{1}{2\sigma^2}\|Y-\theta\|_2^2 \right\}$ is the pdf of Y. Then, (2.42) can be expressed as

$$\mathcal{L}(Y) = \frac{1}{\sigma^2}\left\{ \min_\theta \|Y-\theta\|_2^2 + \|Y\|_2^2 \right\} = \frac{1}{\sigma^2}\left\{ \|Y-\widehat{\theta}\|_2^2 + \|Y\|_2^2 \right\}. \tag{2.43}$$

After the estimation of θ as $\widehat{\theta} = \arg\min_\theta \|Y-\theta\|_2^2 = Y$, $\mathcal{L}(Y)$ can be expressed as

$$\mathcal{L}(Y) = \frac{1}{\sigma^2}\|Y\|_2^2. \tag{2.44}$$

The control limit, $h(\alpha)$, is defined to achieve the desired probability of false alarms, selected a priori:

$$\mathbb{P}_0\left(\mathcal{L}(Y) \geq h(\alpha)\right) = \int_h^\infty f_0(y)dy = 1 - F_{\chi_1^2}(h) = \alpha. \tag{2.45}$$

The power function of the GLR approach is determined as

$$\beta_{\delta^*}(c^2) = \mathbb{P}_\theta(\delta^*(Y) = \mathcal{H}_1) = 1 - F_{1,\gamma(\theta)}(h), \tag{2.46}$$

where $F_{1,\gamma}(Y)$ is the non-central $\chi^2(1,\gamma)$ distribution with one degree of freedom, and the noncentrality parameter $\gamma(\theta) = \frac{1}{\sigma^2}\|P_H^\perp \theta\|_2^2$.

In summary, a fault is flagged by the GLR approach when the decision statistic, $\mathcal{L}(Y)$, exceeds the control limit, $h(\alpha)$. Otherwise, the supervised process is performing normally.

https://t.me/learningnets

To conclude this section, note that the aforementioned univariate monitoring schemes are designed based on two essential assumptions: the process data are uncorrelated and normally distributed. However, measurements from environmental and modern industrial processes are usually autocorrelated and/or nonnormally distributed. The absence of verification of these two major assumptions degrades the monitoring performance of these conventional schemes. To monitor processes with serially correlated measurements, several monitoring schemes have been designed in the literature [72–77]. Two major monitoring approaches to consider correlation in the data can be distinguished [76,58]. The essence of the first approach, which is known as a residuals-based scheme, is to describe the autocorrelation in the data by using a mathematical model. Then, traditional monitoring schemes can be applied to the uncorrelated residuals obtained from the model. The efficiency of these approaches, however, is very sensitive to the prediction quality of the model, which is not always easy to build [78,76]. The essence of the second approach to monitoring the autocorrelated data is to apply the monitoring schemes on the original data, and adjust their decision thresholds suitably to consider the effect of the correlated measurements [76,58]. Many authors have investigated the influence of the violation of the normality assumption on the process monitoring schemes [56,79, 80]. In [79], the impact of a violation of the normality assumption on the Shewhart scheme has been studied by using various known distributions including the uniform, right triangular, gamma, and bimodal distributions. Also, in [81] the impact of skewed distribution on the Shewhart chart has been studied. Several works have designed process monitoring charts for non-Gaussian distribution when the form of the underlying distribution is known. The authors in [82, 83] have developed the EWMA monitoring scheme for multivariate Poisson-distributed data. Also, univariate monitoring charts were designed to monitor nonnormally chi-square distributed processes [84]. However, often in real data, the form of the underlying distribution is unknown, and then our choice may be to use the normal theory results [56]. In such a case, ignoring the nonnormality in the process data can impact the statistical performance of the designed monitoring scheme.

2.4.2 Distribution-based process monitoring schemes

The most commonly used monitoring techniques detect the changes in the mean or variance of the process. However, in many real applications, changes frequently affect the distribution of the inspected process while its mean or variance rests unchanged. This section presents the Kullback–Leibler and Hellinger distances, which are commonly utilized to quantify the deviation separating two distributions. The results of this section are essential in designing LVR-based approaches for monitoring the entire distribution of the process and to broaden the practical application.

https://t.me/learningnets

2.4.2.1 Kullback–Leibler-based monitoring scheme

The essence of Kullback–Leibler divergence is to compute the dissimilarity between two probability distributions. The KLD was originated in the information theory domain and has been applied to several disciplines, such as classification, speech and image recognition [85,86], transportation [87], telecommunication [88], industry [89,90], and medicine [91]. As reported in the literature, KLD has a core role in solving anomaly detection and change detection problems [89,92,93]. This section presents the basic idea of KL divergence and how it can be used as an anomaly detector.

Definition 2.1. *The Kullback–Leibler information between two probability density functions (PDFs)* $p_1(x)$ *and* $p_2(x)$ *is defined as*

$$I(p_1 : p_2) = \int_{\mathbb{R}^{d_x}} p_1(x) \log \left[\frac{p_1(x)}{p_2(x)} \right] dx, \tag{2.47}$$

and between $p_2(x)$ *versus* $p_1(x)$ *is given by:*

$$I(p_2 : p_1) = \int_{\mathbb{R}^{d_x}} p_2(x) \log \left[\frac{p_2(x)}{p_1(x)} \right] dx. \tag{2.48}$$

The KLI is an asymmetric measure (i.e., $I(p_1 : p_2) \neq I(p_2 : p_1)$) and non-negative (i.e., $I(p_1 : p_2) \geq 0$ and $I(p_2 : p_1) \geq 0$). Also, the equality $I(p_1 : p_2) = 0$ is possible only if the two distributions are strictly equal. The KLD distance represents the symmetric form of KLI [89] and is expressed as

$$KLD(p_1; p_2) = I(p_1 : p_2) + I(p_2 : p_1). \tag{2.49}$$

For two Gaussian distributions, $p_1 \sim \mathcal{N}(\mu_0, \sigma_0)$ and $p_2 \sim \mathcal{N}(\mu_1, \sigma_1)$, characterized respectively by their means μ_0, μ_1 and variances σ_0^2, σ_1^2, the KLD distance has the following analytical expression [94]:

$$\begin{aligned} KLD(p_1 \backslash\backslash p_2) &= \frac{1}{\sigma_0\sqrt{2\pi}} \int \exp\left(\frac{(x-\mu_0)^2}{2\sigma_0^2} \right) \\ &\quad \cdot \left[\log \frac{\sigma_1}{\sigma_0} - \frac{(x-\mu_0)^2}{2\sigma_0^2} + \frac{(x-\mu_1)^2}{2\sigma_1^2} \right] dx \\ &= \frac{(\mu_1-\mu_0)^2}{2\sigma_1^2} + \frac{1}{2}\left(\log \frac{\sigma_1^2}{\sigma_0^2} + \frac{\sigma_0^2}{\sigma_1^2} - 1 \right). \end{aligned} \tag{2.50}$$

In the case when anomalies occur only in the mean (i.e., $\sigma_0^2 = \sigma_1^2$), Eq. (2.50) can be expressed as

$$KLD(p_1 \backslash\backslash p_2) = \frac{(\mu_1-\mu_0)^2}{2\sigma_1^2}. \tag{2.51}$$

https://t.me/learningnets

From (2.50) we can see that the second term is strictly positive for any $\sigma_0 \neq \sigma_1$. We conclude that in the case of faults simultaneously affecting the mean and variance of the monitored Gaussian variable the value of KLD is larger than its value in the case of a change in the mean alone. Therefore, it is also clear that detecting anomalies in the mean and variance of process measurements simultaneously using the KLD is easier than detecting anomalies that occur in the process mean alone.

If we use KLD to characterize closeness of distributions $p_1(x)$ and $p_2(x)$, it turns out that its value becomes close to zero when the two distributions are similar; otherwise it deviates significantly from zero. Thus, KLD can be utilized for anomaly detection. The KLD-based test to choose between the null hypothesis $\mathcal{H}_0$ and the alternative $\mathcal{H}_1$ is given by

$$\delta(Y) = \begin{cases} \mathcal{H}_0 \text{ if } KLD(p_1 \backslash\backslash p_2) < H_D, \\ \mathcal{H}_1 \text{ else.} \end{cases} \tag{2.52}$$

From the distribution of the decision statistic, KLD, the threshold of the KLD mechanism can be computed nonparametrically as the $(1-\alpha)$th quantile of the approximated distribution of KLD statistic obtained by kernel density estimation (KDE). The nonparametric threshold is computed using anomaly-free data. We declare an anomaly when the KLD statistic exceeds the decision threshold.

2.4.2.2 Hellinger-based monitoring scheme

The test statistic considered here is based on the Hellinger distance (HD) metric. It was introduced by Ernest Hellinger and can be adapted to measure the similarity among two different probability vectors [95–97]. As with KLD, HD distance also plays a central role in several problems of mathematical statistics [98,96, 97]. The HD measure has been widely exploited in numerous disciplines, including pattern recognition [99], image processing [100], classification [101], and anomaly detection [102,97,103]. Additionally, the HD metric has been exploited in different applications including cybersecurity [103] and fraud detection in insurance [104].

Definition 2.2 (Hellinger distance). *The HD between two PDFs $p_1(x)$ and $p_2(x)$ is the value*

$$HD^2(p_1, p_2) = \frac{1}{2}\sum(\sqrt{p_1(x)} - \sqrt{p_2(x)})^2. \tag{2.53}$$

We can see that HD is the Euclidean norm of the difference between the square root vectors:

$$HD^2(p_1, p_2) = \frac{1}{\sqrt{2}}\|\sqrt{p_1} - \sqrt{p_2}\|_2. \tag{2.54}$$

https://t.me/learningnets

Note that the HD distance is a symmetric metric of p_2 and p_1 (i.e., $HD^2(p_1, p_2) = HD^2(p_2, p_1)$). It has all the properties of a metric and is bounded (i.e., the HD satisfies $0 \leq HD^2(p_1, p_2) \leq 1$), and the equality $HD^2(p_2, p_1) = 0$ is possible only if $p_1 = p_2$.

The analytical form of HD between normal distributions $p_1 \sim \mathcal{N}(\mu_0, \sigma_0)$ and $p_2 \sim \mathcal{N}(\mu_1, \sigma_1)$ can easily be obtained as in [94],

$$HD^2(p_1, p_2) = 1 - \sqrt{\frac{2\sigma_0\sigma_1}{\sigma_0^2 + \sigma_1^2}} \exp\left(-\frac{1}{4} \frac{(\mu_0 - \mu_1)^2}{\sigma_0^2 + \sigma_1^2} \right), \tag{2.55}$$

where μ_0, μ_1 are the means and σ_0^2, σ_1^2 the variances for p_1 and p_2, respectively. In the case of only mean shift (i.e., $\sigma_1^2 = \sigma_0^2$), HD is given as

$$HD^2(p_1, p_2) = 1 - \exp\left(-\frac{1}{8} \frac{(\mu_0 - \mu_1)^2}{\sigma_0^2} \right). \tag{2.56}$$

Note that a small HD value indicates high similarity between two distributions, and a high value of HD reflects a significant dissimilarity between them. Similar to KLD, the HD metric can be used as an anomaly indicator. To conclude this subsection, we note that the analytical form of the HD metric exists for certain non-Gaussian distributions such as Weibull, Poisson, and Beta distributions. When the HD measure has no closed form, it can be approximated from the data sets using approaches like KDE [105].

Note that the KLD and HD measures are a special case of the power divergence, which has been first defined in [106,107]. Also, it has been shown that these two measures are asymptotically identical, up to a constant factor, when the ratio between distributions p_1 and p_2, that is, p_1/p_2, is close to 1 [108].

2.4.2.3 Limitations of univariate monitoring schemes

Today, modern engineering and industrial processes have become more complex and contain several parts with an important number of measured variables. Despite the fact that these univariate monitoring schemes are successfully used in several applications, they are suited only when each monitored variable is independent of the others and thus such schemes ignore the interaction between correlated variables. However, in engineering and environmental processes, several variables require to be monitored simultaneously, which results in a misleading analysis when using univariate monitoring schemes. Applying a single monitoring scheme for each process variable is cumbersome and ineffective, in particular in the presence of a high number of variables. Of course, the use of multivariate schemes to monitor multivariate processes permits minimizing the total number of monitoring schemes compared to univariate schemes. For instance, assume that the inspected process comprises two variables X_1 and X_2. Fig. 2.6 illustrates the difficulty of using independent univariate monitoring schemes to monitor a bivariate process. It can be seen from Fig. 2.6 that the

https://t.me/learningnets

observations of X_1 and X_2 are within their corresponding control limits. Also, Fig. 2.6 clearly demonstrates the ability of a multivariate scheme to detect this fault (i.e., the red star in the upper right corner (outside the control ellipse)). This point clearly represents a mismatch between X_1 and X_2. Therefore, monitoring these two process variables separately can be very misleading.

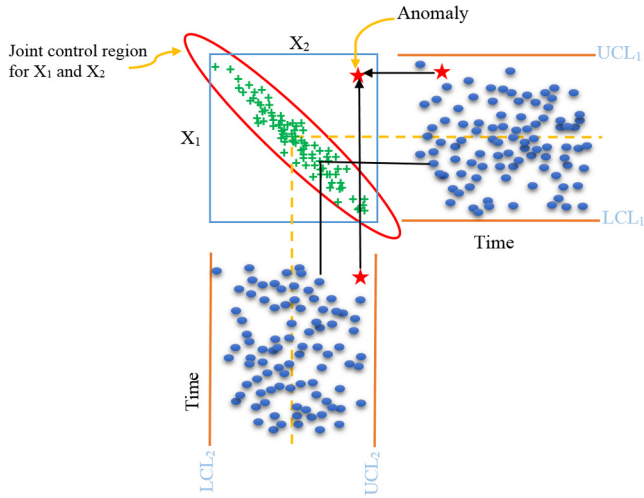


FIGURE 2.6 Illustration of the disadvantage of using univariate schemes for monitoring multivariate data.

Furthermore, the use of univariate charts for monitoring multivariate processes has at least two limitations, which are described briefly below:

- Once joint univariate monitoring schemes are applied for multivariate process monitoring, each univariate monitoring schemes has its own procedure parameters that should be tuned before it can be used. For example, if a univariate EWMA is utilized for monitoring a single process variable, then the univariate EWMA scheme has two parameters involved (i.e., γ and L). Thus, the joint monitoring scheme to monitor p process variables would have $2p$ parameters to determine.
- Suppose we have two variables, X_1 and X_2, which are monitored individually with a univariate scheme. For instance, 3-sigma thresholds are applied to each process variable. Each scheme will result in the value of the probability of a false alarm of $\alpha = 0,27\%$ (type I error). So the overall rate of false alarms for this case is $\acute{\alpha} = 0,54\%$. For instance, if 100 schemes are applied for monitoring simultaneously, the probability of a false alarm will reach 27% (i.e., 0.27% * 100). In general, if p statistically independent process variables and monitoring schemes with a false alarm rate are used, the overall false alarm rate $\acute{\alpha}$ is

$$\acute{\alpha} = 1 - (1 - \alpha)^p. \tag{2.57}$$

https://t.me/learningnets

However, when several process variables are inspected based on a single multivariate scheme, the overall rate of false alarms will remain at 0.27%.

Overall, the multivariate monitoring schemes that are constructed in the multivariate framework enable more effective fault detection than a joint monitoring scheme involving multiple univariate monitoring schemes (for details see [109, 59]).

2.4.3 Multivariate process monitoring schemes with parametric and nonparametric thresholds

To supervise multiple variables simultaneously, multivariate statistical monitoring schemes such as multivariate Shewhart, multivariate EWMA (MEWMA) and multivariate CUSUM (MCUSUM) have been designed as multivariate counterparts of their univariate schemes. In this section, the objectives of each monitoring schemes (i.e., multivariate Shewhart, MCUSUM, and MEWMA) and comment on their benefits and limitations are outlined.

2.4.3.1 Multivariate Shewhart schemes

The multivariate counterpart of the Shewhart monitoring scheme is one of the most popular multivariate monitoring techniques to inspect the mean shift of a normally distributed process. Suppose that we inspect a multivariate process $\mathbf{X}_t = (X_1, X_2, \ldots, X_p)^T$, where $X_i \in \mathbb{R}^m$, and the collected data are normally distributed with mean μ and covariance matrix Σ. A multivariate Shewhart scheme, also known as a T^2 or as a χ^2 scheme [110], to monitor the process mean in multivariate data is based on the following monitoring statistic:

$$T_t^2 = \left[(\mathbf{x}_i - \mu)^T \Sigma^{-1} (\mathbf{x}_t - \mu)\right] \tag{2.58}$$

where $\mathbf{x}$ is a vector of p variables, and μ is a vector of fault-free means of every variable. The decision threshold of this monitoring scheme, which is derived using fault-free data, is $H = \chi^2_{\alpha,m}$. For new test measurements, the T^2 scheme flags an anomaly if the value of T^2 surpasses a threshold value, $\chi^2_{\alpha,m}$ [25]. Since this scheme uses only the current observation and ignores all past data, it shows poor detection performance in the presence of small changes. Note that this scheme is designed under the assumptions of uncorrelated and multivariate Gaussian distributed measurements.

2.4.3.2 Multivariate cumulative sum scheme (MCUSUM)

As a remedy to the insensitivity of the Shewhart scheme to small changes, MCUSUM and MEWMA monitoring schemes have been designed [109]. The essence of the MCUSUM approach is to exploit information of all available data by computing the cumulative sum of the differences of each previously observed vector compared to the nominal value for monitoring the means of the

https://t.me/learningnets

multivariate process [111]. MCUSUM was first introduced in [112] as a natural version of the univariate CUSUM scheme to deal with multivariate data. Numerous extensions of MCUSUM have been developed [112–114]. In this subsection, to simplify the presentation, we only present the MCUSUM scheme, which received considerable consideration in the literature, developed in [112]. Let $\mathbf{X}_t = (X_1, X_2, \ldots, X_p)^T$, be a sequence of i.i.d. $\mathcal{N}_p(\mu, \Sigma)$, where $X_i \in \mathbb{R}^m$; m is the number of variables. The MCUSUM decision statistic to plot is

$$S_t = \sqrt{L_t^T \Sigma^{-1} L_t}, \tag{2.59}$$

where

$$L_t = \begin{cases} 0 \text{ if } C_t \leq k, \\ (L_{t-1} + X_t - \mu_0)(1 - \frac{k}{C_t}) \quad \text{otherwise}, \end{cases} \tag{2.60}$$

and

$$C_t = \sqrt{(L_{t-1} + X_t - \mu_0)^T \Sigma^{-1} (L_{t-1} + X_t - \mu_0)}. \tag{2.61}$$

Crosier [109], recommended using $L_0 = 0$ and $K = \frac{\sqrt{(\mu_1 - \mu_0)^T \Sigma^{-1} (\mu_1 - \mu_0)}}{2}$. The MCUSUM scheme signals faults when

$$S_t = \sqrt{L_t^T \Sigma^{-1} L_t} > H, \tag{2.62}$$

where the decision threshold H is computed via simulation to reach the desired probability of a false alarm.

2.4.3.3 *Multivariate exponentially weighted moving average scheme (MEWMA)*

The MEWMA scheme was designed by Lowry et al. [115] as a multivariate extension of EWMA for detection faults occurring in the mean of multivariate data. The desirable characteristic of MEWMA is its capacity to detect small changes in multivariate correlated process variables by incorporating all information from the available datasets in the decision function. It has been widely used in several applications such as renewable energy [116,117], air quality monitoring [118], industrial processes, and medical healthcare.

Let $\mathbf{X}_t = (X_1, X_2, \ldots, X_m)^T$ be an m-dimensional set of measurements at time t. The MEWMA scheme designed by Lowry et al. [115] is given by

$$\begin{cases} Z_0 = \mu_{\mathbf{0}}, \\ \mathbf{Z}_t = \mathbf{\Gamma} \mathbf{X}_t + (\mathbf{I}_{m \times m} - \mathbf{\Gamma}) \mathbf{Z}_{t-1} \quad t = 1, 2, \ldots, n, \end{cases} \tag{2.63}$$

where $\mathbf{\Gamma} = \text{diag}(\gamma_1, \gamma_2, \ldots, \gamma_m)$ is a diagonal matrix of smoothing parameters, and m is the number of process variables. Designing an MEWMA scheme needs

https://t.me/learningnets

attribution of weight $0 < \gamma \leq 1$ that is utilized to allocate importance to observations; Z_i is the ith EWMA vector, and X_i is the ith observation vector, $i = 1, 2, \ldots, n$. We see that if $\mathbf{\Gamma} = \mathbf{I}$, then the MEWMA scheme becomes equal to the T^2 scheme. Very often, in practice, in the absence of prior reason to assign a different weight for each component, for simplicity it is convenient to assume $\gamma_1 = \gamma_2 = \cdots = \gamma_m = \gamma$. Thus (2.63) can be written as

$$\mathbf{Z}_t = \gamma \mathbf{X}_t + (1 - \gamma)\mathbf{Z}_{t-1}. \tag{2.64}$$

For constructing the MEWMA chart, the MEWMA statistic can be computed recursively as [115]

$$\mathbf{V}_t^2 = \mathbf{Z}_t^T \Sigma_{\mathbf{Z}_t}^{-1} \mathbf{Z}_t, \tag{2.65}$$

where $\Sigma_{\mathbf{Z}_t}$ is the variance–covariance matrix of $\mathbf{Z}_t$,

$$\Sigma_{\mathbf{Z}_t} = \frac{\gamma}{(2-\gamma)}\left[1 - (1-\gamma)^{2n}\right]\Sigma, \tag{2.66}$$

where Σ is the covariance matrix of the input data. For large t, the covariance matrix converges to $\Sigma_{\mathbf{Z}_t} = (\frac{\gamma}{(2-\gamma)})\Sigma$.

Under the hypothesis of no change, the MEWMA statistic is Gaussian distributed with variance–covariance matrix Σ_{Z_i}, $Z \sim \mathcal{N}(0, \Sigma_{Z_i})$. In the presence of mean shift, μ_1, the distribution of MEWMA statistic is $Z \sim \mathcal{N}\Big(\gamma \sum_{j=1}^{n} \left[(1-\gamma)^{n-j}\theta\right], \Sigma_{Z_i}\Big)$. Hence, it follows that $\mathbf{V}_t^2$ is distributed as a χ^2 with P degrees of freedom in the absence of change. However, since the variables $\mathbf{V}_t^2$, $t = 1, 2, \ldots$, are correlated, the decision threshold of the MEWMA scheme cannot directly be chosen as the $(1 - \alpha)$th quantile of the χ_P^2 distribution. Numerous procedures have been suggested in the literature to compute the decision threshold h with respect to the parameters γ, p, and α, such as Monte Carlo simulation and Markov chain approximation [77,119]. In [120], a procedure implemented in Fortran to compute the MEWMA threshold is given to achieve a fixed number of false alarms and a given γ. Also, in [121] a method to compute the optimal γ and corresponding threshold h for a selected combination of p, the magnitude of the shift to be detected, and the targeted ARL0 is developed.

Let us comment further on weighting matrix Γ. In [122], it has been shown that the detection performance of the MEWMA scheme can be enhanced when using Γ with nonzero off-diagonal elements compared to using a single weighting parameter γ. However, in this case, the construction of the MEWMA scheme is not an easy task, in particular, to properly select the weighting matrix parameters. This is mainly due to the fact that fault direction and the correlation among the process variables should be considered in the design of the MEWMA.

Furthermore, to use the MEWMA monitoring schemes to detect anomalies in the process covariance matrix, there have been several MEWMA extensions in the literature; we refer to [123–125].

https://t.me/learningnets

To conclude this subsection, we note that utilizing the conventional multivariate monitoring schemes (i.e., multivariate Shewhart, MCUSUM, and MEWMA) for inspecting high-dimensional processes with collinearities may be ineffective.

2.5 Linear LVR-based process monitoring strategies

First, let us outline the reason why multivariate monitoring schemes including multivariate Shewhart, MCUSUM, and MEWMA schemes are not frequently used alone in applications with numerous correlated process variables. From the previous section, we note that the design of the decision statistics of these three multivariate schemes requires the calculation of the inverse covariance matrix. However, since collected multivariate data from engineering and environmental processes are correlated, the matrix inversion of the covariance matrix can frequently generate poor numerical results because of the ill-conditioning problem. Accordingly, it becomes obvious and particularly relevant to use projection-based monitoring methods, such as PCA and PLS, when the covariance matrix is ill-conditioned. As described in Sect. 2.2.3, LVR models are well suited for multivariate monitoring as they mitigate the problem of ill-conditioning in other multivariate schemes including multivariate Shewhart chart, MCUSUM, and MEWMA schemes. For instance, to avoid the ill-conditioning problem, projection-based methods such as PCA can be used to decompose the covariance matrix of the raw data and generate transformed variables having better numerical characteristics that enable process monitoring via the conventional multivariate schemes (e.g., MEWMA and MCUSUM). In this section, the general framework combining the LVR models and univariate or multivariate monitoring techniques is described.

2.5.1 Conventional LVR monitoring statistics

The general framework of the data-based monitoring approaches such as input-space models (e.g., PCA) and input–output models (e.g., PLS) consists of two phases: (i) model construction and (ii) online monitoring as indicated in Fig. 2.7. The first phase focuses on designing a reference model based on data gathered from the inspected process when running within nominal conditions. Then the decision thresholds of the monitoring scheme are computed (e.g., T^2 or another chart). Generally, thresholds are defined in the subspace of the retained PCs or the subspace of the residuals. These two subspaces usually serve as a monitoring subspaces for fault detection. For instance, in PCA, a low-dimensional empirical model that captures information from process data is built by picking the relevant number of principal components as that catching the most variability in data. In the second phase, new testing data are projected in the subspace model, residuals are computed using the designed model, and test statistics, such as T^2 and SPE statistics, are computed and compared with the previously computed

https://t.me/learningnets

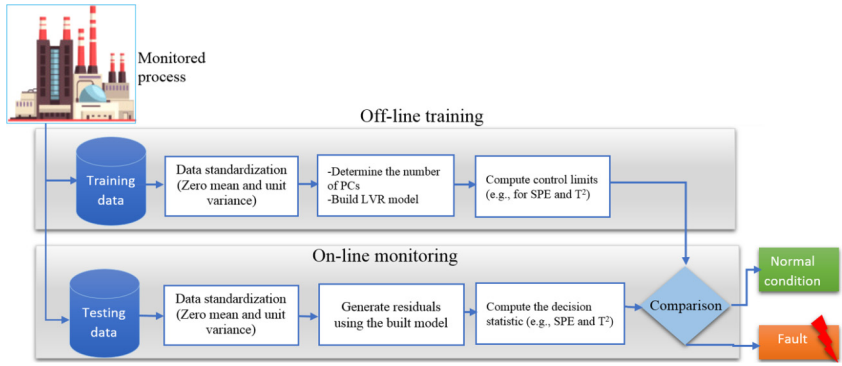


FIGURE 2.7 A flowchart representing the essence of data-based fault detection.

threshold for fault detection. The variation in the retained PCs and residuals are frequently verified respectively by the T^2 and SPE schemes to detect abnormal operations, but MCUSUM and MEWMA may also be used. After a reference model is built using PCA or another approach, numerous monitoring schemes can be applied for fault detection. To perform SPE and T^2 analysis, we may use any model identified using various applicable methods other than PCA.

2.5.1.1 *Hotelling's T^2 statistic*

The LVR methods (e.g., PCA and PLS) are usually followed by fault detection methods using SPE and T^2 statistics. Abnormalities in the process can be detected by checking whether the measurements are not beyond the region of the normal operation in the PCs subspace or in the residuals subspace. The T^2 statistic computes the fluctuations in the PCs alone at each time instance. More specifically, the T^2 value is defined as the sum of the squares of the retained PCs divided by the corresponding eigenvalue calculated from fault-free data characterizing normal operation as [25]

$$T^2 = x^T \widehat{P} \widehat{\Lambda}^{-1} \widehat{P}^T x = \sum_{i=1}^{l} \frac{t_i^2}{\lambda_i}, \tag{2.67}$$

where λ_i is the eigenvalue (variance) of the PC t_i. If the actual covariance matrix is determined from the sample matrix, the control limit of the T^2 scheme is given by [25]

$$T^2_{l,n,\alpha} = \frac{l(n-1)}{n-l} F_{l,n-l,\alpha}, \tag{2.68}$$

where α is the level of significance, and $F_{l,n-l}$ is the Fisher F distribution with l and $n-l$ degrees of freedom. In the case of multivariate Gaussian distributed data $\mathbf{X}$, and if the size of the sample is large enough, the T^2 statistic threshold

https://t.me/learningnets

can be computed from $T_\alpha^2 = \chi_{l,\alpha}^2$. The decision threshold value is computed using fault-free data.

2.5.1.2 Q statistic or squared prediction error (SPE)

To detect faults in the residual subspace, the SPE (also called Q statistic) is usually used [22],

$$Q = \mathbf{e}^T \mathbf{e}. \tag{2.69}$$

The decision threshold of the Q statistic is defined as [126]

$$Q_\alpha = \varphi_1 \left[\frac{h_0 c_\alpha \sqrt{2\varphi_2}}{\varphi_1} + 1 + \frac{\varphi_2 h_0 (h_0 - 1)}{\varphi_1^2} \right], \tag{2.70}$$

where c_α is the value of the normal distribution with α level of significance, $\varphi_i = \sum_{j=l+1}^{m} \lambda_j^i$ for $i = 1, 2, 3$, and $h_0 = 1 - \frac{2\varphi_1 \varphi_3}{3\varphi_2^2}$.

It should be noted that the Q scheme is sensitive to modeling errors and its detection quality mainly relies on the number of retained PCs [22].

Fig. 2.8 illustrates an example of a data point "b" with a large T^2 value representing the (horizontal) distance between the center of the PC plane and this point. Also, Fig. 2.8 shows another point with a significant Q value, which represents the vertical distance between the PC plan and this atypical data point "a". Generally speaking, the Q indicates the mismatch between the data sample and the model.

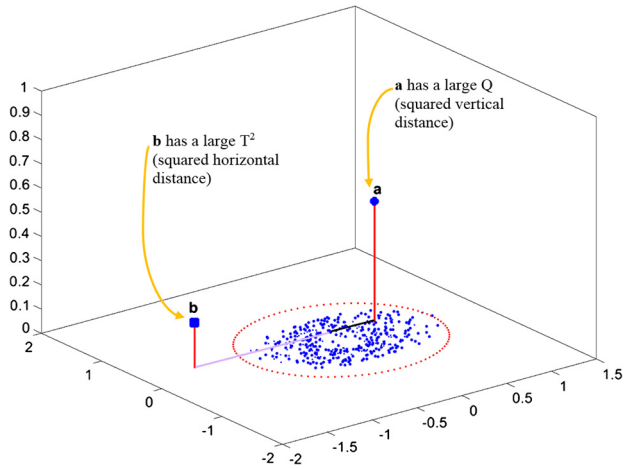


FIGURE 2.8 Projected measurements (blue [dark gray in print version]) and two observations "a" and "b" with values of their corresponding T^2 and Q statistics.

To conclude this section, note that the decision thresholds of Q and T^2 schemes are derived based on the fact that the data are uncorrelated and normally distributed. To remedy this limitation, an alternative solution based on

https://t.me/learningnets

kernel density estimation (KDE) can be used [127]. After estimating the distribution of the decision statistic (e.g., Q and T^2), the threshold can be computed nonparametrically as the $(1-\alpha)$th quantile of this estimated distribution.

As we explained in Sect. 2.5.1, in projection-based monitoring approaches, the PC and residual subspaces are frequently inspected via the Shewhart-type monitoring chart (i.e., Q and T^2) for detecting abnormal changes. However, these schemes are designed to detect large faults but they fail to detect small changes. Alternatively, other types of multivariate schemes, such as MEWMA and MCUSUM, can be used [118,128]. Furthermore, the capability to detect small faults can be enhanced by amalgamating univariate schemes (CUSUM or EWMA) with Q and T^2, but it needs a good selection of the parameters of CUSUM or EWMA schemes [129]. Instead of the conventional multivariate monitoring schemes with projection based methods, other approaches exploit the sensitivity of distribution-based schemes, such as KLD [130,92,131,89] and HD [102,132], to detect incipient faults.

2.5.2 Fault isolation

Today engineering and environmental processes have become far more complex due to advances in technology. Multiple relevant process variables require to be supervised all together at the same time. Once a fault is detected, fault isolation (attribution) is required to ensure safe operation of the process and to avoid the risk of process shutdowns by making the necessary correction before the fault propagates and contaminates the process [133]. Isolating the detected faults, which is necessary for a systematic diagnosis and maintenance, is not an easy task because of the large number of variables that need to be inspected. The aim of fault isolation is finding process variables that contributed to the abnormal change, thus allowing the operators to concentrate only on the subsystem where the fault happened. Accordingly, fault isolation is crucial to significantly reduce the losses in profitability and to facilitate maintenance tasks. In this section, two basic tools for fault isolation, namely latent variable contribution plots and RadViz visualizer, are presented. In Chap. 3, more details about fault attribution in multivariate statistical monitoring will be presented.

2.5.2.1 Fault isolation using modified contribution plots

The underlying intuitive idea behind the contribution plot is to find the process variables with large contributions that are potential indicators to locate the source of abnormalities. Hence, the contribution plot provides relevant information to operators for enabling efficient fault diagnosis by focusing their analysis on a small number of variables which may be the cause of the detected anomalies [134,21,135,136].

Several fault isolation techniques are developed within multivariate statistical monitoring approaches, including contributions to the T^2 statistic, contributions to SPE, and contribution to individual scores, which are the most popular

https://t.me/learningnets

approaches in fault diagnosis [21,137,138]. The essence of these contribution-based approaches is that the variables with faults are expected to give significant contributions to the fault detection statistics. Most of the contribution-based fault isolation techniques have been summarized and generalized in [136]. However, this approach investigates the process variables separately without considering the correlation between them, which make it inappropriate for isolating faults in multiple variables that simultaneously participate in the fault occurrence. Other researchers [139] use the GLR test for fault isolation, which can also provide an estimate of the size of the isolated sensor fault. In [140], a reconstruction-based method is introduced to identify single and multiple faults. Another approach combining the desirable benefits of the contribution analysis and the reconstruction-based approach has been designed to further enhance the fault isolation. However, the reconstruction-based contribution approach is suitable only for identifying unidimensional faults with relatively large magnitudes. Recently, in [141], an approach based on exponential smoothing reconstruction has been designed for isolation of incipient faults. Next, we will review two of the most commonly used techniques in fault isolation – the contribution methods.

T^2 contribution approach

The T^2 contribution-based approaches have been broadly applied in the industry to identify the process variables that significantly contribute to the fault detected by the T^2 monitoring scheme. The essence of this approach is based on computing the gradient of T^2 with regards to every variable. From Eq. (2.67) the gradient of the T^2 statistic provides the sensitivity to the variable vector $\mathbf{x}$ as

$$\mathbf{S}_{T^2} = \frac{\partial(T^2)}{\partial \mathbf{x}} = 2\mathbf{P}(\mathbf{P}^T \sum_{\mathbf{x}} \mathbf{P})^{-1}\mathbf{P}^T\mathbf{x} = 2\mathbf{P}\Lambda_p^{-1}\mathbf{P}^T\mathbf{x}. \tag{2.71}$$

The constant "2" can be ignored and the contribution from each variable can be computed as

$$T^2\text{cont}_{x_i} = \frac{\sum_{j=1}^{k}(p_j x_i)^2}{\lambda_j} = x_i^2 \sum_{j=1}^{k} \frac{p_j^2}{\lambda_j}. \tag{2.72}$$

If $T^2\text{cont}_{y_i}$ is the largest among all values calculated for $i = 1, \ldots, n$, then y_i is indicated as a potential cause of the fault.

SPE contribution approach

When a fault is flagged by the Q scheme, the Q contribution plot is usually applied to isolate the fault [135]. Let $\mathbf{E} = [\mathbf{e}_1\ \mathbf{e}_2 \ldots \mathbf{e}_m]$ the residual matrix and $\mathbf{e}_j$ be the jth columns of the residuals matrix. From (2.69), we have

$$Q_j = \mathbf{e}_j^T \mathbf{e}_j. \tag{2.73}$$

https://t.me/learningnets

Indeed, each row of the residual matrix $\mathbf{E}$, $\mathbf{e}_i$ is the Q contribution for this specific sample. The fractional contribution of the ith variables to the overall Q at sample instant j can be computed as

$$Q\text{cont}_{X_j} = \frac{e_{ij}^2}{Q_j}. \tag{2.74}$$

The Q contributions' plot is helpful to identify the variable making an important contribution to SPE as the suspected source of the fault.

2.5.2.2 Fault diagnosis using RadViz visualizer

RadViz visualization was first proposed by Patrick Hoffman et al. in [142,143] as a multivariate data visualization tool to classify DNA sequences. Indeed, RadViz enables nonlinear projecting of high dimensional data into a 2-dimensional space and allows direct interpretation of the position of the observation in the space. It can visualize high dimensional data comprising three or more variables in a 2-dimensional space. RadViz has been widely used to visualize, interpret, and cluster high dimensional datasets like microarray data and stock exchange trends.

Generally speaking, RadViz reaps the benefits of the dimensionality-reduction methods with scatterplots, where the value of each observation could be indicated from the distance to the anchors [144]. The anchor points are the visualized variables uniformly spaced in the circumference of a unit circle. Fig. 2.9 gives an illustrative example showing the basic concept of a RadViz Visualizer. In this method, each point is linked to every anchor by a virtual spring. Here, the position of each observation in the circle is calculated based on its relative influence by the anchors (i.e., spring force for each spring). The coordinates of each observation are at the point where the system of spring forces attains equilibrium. The influence of each anchor on the projected observation is relative to the amplitude of the coordinate for that anchor. The coordinates of each observation in RadViz are computed as

$$x_i = \frac{\sum_{i=1}^{d} x_i \cos\theta_j}{\sum_{i=1}^{d} x_i}, \tag{2.75}$$

$$y_i = \frac{\sum_{i=1}^{d} x_i \sin\theta_j}{\sum_{i=1}^{d} x_i}, \tag{2.76}$$

where x_i and y_i are the coordinate of the projected observation, and θ_j denotes the angular position on the circle for the anchor j; $a_{i,j}$ represents the value for dimension j for the observation i; d is the number of dimensions; and n the number of observations.

The main characteristics of RadViz method can be summarized as follows. In this 2D space, observations in the center have approximately equal dimensional

https://t.me/learningnets

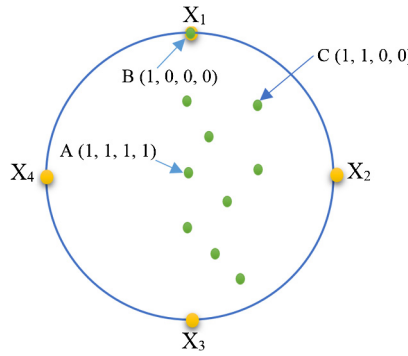


FIGURE 2.9 An illustrative example of the principle of RadViz Visualizer.

values. On the other hand, observations with some coordinate values significantly larger than the others are impacted or related to the dimensional anchors closer to these observations. In other words, the larger the value for a specific dimension, the closer the projected observation to the associating anchor, and vice versa. It is this desirable characteristic that makes RadViz a useful tool to assist fault diagnosis. In Fig. 2.9, RadViz is utilized for visualizing multidimensional data in a four-dimensional space (Fig. 2.9). The yellow points (light gray in print version) in the circumference of the circle, which are called anchor points, represent the dimension of the data (e.g., process variables or latent variables). In this example, observation B has coordinates $(1, 0, 0, 0)$ and it is placed at X_1, which means that this observation is fully related to this anchor (variable) without any ambiguity. Observation C with coordinates $(1, 1, 0, 0)$ has a large correlation with X_1 and X_2 and low correlation with X_3 and X_4. Finally, observation A with coordinates $(1, 1, 1, 1)$ is placed in the center of the circle, meaning that it is impacted in a similar way by all anchors.

Until now, the RadViz tool has not been well exploited in fault detection and diagnose to assist fault isolation. In Sect. 2.6.2, RadViz is used to diagnose anomalies in influent measurements at water resource recovery facilities.

2.6 Cases studies

2.6.1 Simulated example

Here, the prediction efficacy of the OLS, RR, and the LVR models (PLSR and PCR) is evaluated using synthetic data sets. The simulated data comprises ten input variables and one response variable. In this example, the input variables x_1 and x_2 represent "block" and "heavy-sine" signals, and the other seven variables are obtained as linear combinations of x_1 and x_2 (i.e., the input matrix $X = [x_1, x_2, \ldots, x_9]$ is of rank 2):

$$\mathbf{x}_3 = \mathbf{x}_1 + \mathbf{x}_2; \qquad \mathbf{x}_4 = 0.3\mathbf{x}_1 + 0.7\mathbf{x}_2; \qquad \mathbf{x}_5 = 0.3\mathbf{x}_3 + 0.2\mathbf{x}_4;$$

https://t.me/learningnets

$$\mathbf{x}_6 = 2.2\mathbf{x}_1 - 1.7\mathbf{x}_3; \qquad \mathbf{x}_7 = 2.1\mathbf{x}_6 + 1.2\mathbf{x}_5; \qquad \mathbf{x}_8 = 1.4\mathbf{x}_2 - 1.2\mathbf{x}_7;$$
$$\mathbf{x}_9 = 1.3\mathbf{x}_2 + 2.1\mathbf{x}_1; \qquad \mathbf{x}_{10} = 1.3\mathbf{x}_6 - 2.3\mathbf{x}_9.$$

Then, the output is obtained as a linear combination of all inputs variables as

$$\mathbf{y} = \sum_{i=1}^{10} b_i \mathbf{x}_i, \tag{2.77}$$

where $b_i = \{0.07, 0.03, -0.05, 0.04, 0.02, -1.1, -0.04, -0.02, 0.01, -0.03\}$ for $i = 1, \ldots, 10$. First, 128 noise-free samples are generated and then tainted with zero-mean Gaussian noise. A sample of output measurements with signal-to-noise ratio (SNR) equal to 20 is given in Fig. 2.10. After constructing the four models using training data, they are compared in terms of the output prediction MSE using unseen test measurements.

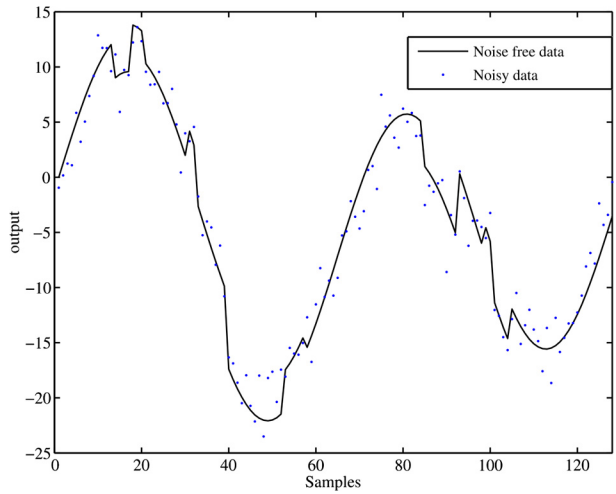


FIGURE 2.10 A sample of output measurements with SNR = 20.

To study the effect of noisy data on the prediction quality of the four models, the simulated noise-free data has been tainted with noise SNR levels equal to 10, 20, and 50. The simulated data are divided into two portions: testing and training data. For PLSR and PCR, the number of retained PCs is selected using the cross-validation technique. The developed model performances are assessed by the model prediction, i.e., MSE using the testing data set. To achieve statistically valid results, 1000 stochastic simulations have been carried out. Prediction results of the five models are presented in Table 2.1 and Fig. 2.11. Results in Fig. 2.11 indicate that RR provided better prediction compared to OLS. Also, it can be seen that PCR and PLS achieved higher performance compared to the full rank models (OLS and RR). The use of only a few principal components

https://t.me/learningnets

TABLE 2.1 The prediction MSEs of the four models.

Model	SNR=10	SNR=20	SNR=50
PLS	2.183	1.111	0.4510
PCR	2.223	1.127	0.4565
RR	2.288	1.148	0.4584
OLS	3.849	1.955	0.7853

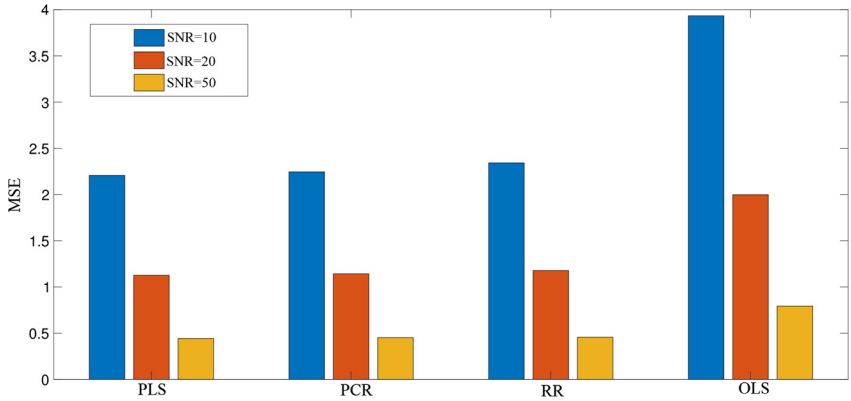


FIGURE 2.11 The prediction MSEs of the studied models under different SNR levels.

in LVR models allows removing a portion of the noise in the data and thus improving the prediction quality. From Table 2.1, we can see that the PCR and PLS models provide comparable prediction performance, which is in agreement with the literature [145,146].

2.6.2 Monitoring influent measurements at water resource recovery facilities

Wastewater treatment plants (WWTPs) provide sustainable solutions to water scarcity. As initial conditions offered to WWTPs, influent characteristics (ICs) affect treatment units' states, ongoing processes mechanisms, and product quality. Anomalies in ICs, often raised by abnormal events, need to be monitored and detected promptly to improve system resilience. In this section, the ability of the PCA-based approach to monitor influent measurements at WWTPs is investigated. For this purpose, historical ICs from the WWTP based in King Abdullah University of Science and Technology (KAUST), Thuwal, Saudi Arabia (Fig. 2.12) were engaged.

In this dataframe, operators have kept seven years of records of 21 features (Table 2.2). Measurements were performed on samples taken from the headwork of the WWTP in order to maintain compliance with regulations and standards. This plant has a sustainability mission to provide all treated effluents for irriga-

https://t.me/learningnets



FIGURE 2.12 (A) The pipeline from the campus, (B) the grit chamber and grit classifier inside the headwork, (C) the neutralization pit.

TABLE 2.2 List of inspected influent variables in KAUST WWTP.

No.	Variable name	Measurement scopes	Limit
1	InFlow-LS1	Wastewater inflow, from the whole campus area, in m^3/day	–
2	InFlow-LS8	Wastewater inflow, from a desalination plant, in m^3/day	–
3	InFlow-DP	Wastewater inflow, recycled from WRRF itself, in m^3/day	–
4	InFlow-Total	Wastewater inflow, from the whole university, in m^3/day	2500–6000
5	Temp	Temperature, in Celsius	–
6	pH	potential of hydrogen, unitless	6–9
7	Conductivity	Conductivity, in μS/cm	< 2850
8	TDS	Total dissolved solid, in mg/L	< 2000
9	TSS	Total suspended solid, in mg/L	< 312
10	CaHardness	Calcium hardness, in mg/L	–
11	MgHardness	Magnesium hardness, in mg/L	–
12	TotalAlkalinity	Total alkalinity, in mg/L	< 200
13	BOD_5	5-day Biochemical Oxygen Demand, in mg/L	< 264
14	COD	chemical oxygen demand, in mg/L	< 527
15	FOG	Fat, oils and grease, in mg/L	–
16	TKN	Total Kjeldahl Nitrogen, in mg/L	< 40
17	NH_3N	Ammonia, in mg/L	< 25
18	NO_3N	Nitrate, in mg/L	< 10
19	PO_4P	Phosphate, in mg/L	–
20	Cl	Chloride, in mg/L	–
21	Boron	Boron, in mg/L	< 2.5

tion reuse across the campus, which greatly reduces the potable water demand of the university. Abnormal events have occurred, such as intensive rainfalls, seawater intrusion into the lift station, discharge from construction area, and hypochlorite dosage. All of them caused effects on the WWTP operation (downstream processes compared to the inflow), and therefore were identified and reported by operators. R package Amelia was imported to impute a few missing data (132/63,950, less than 1%) during the preprocessing step. By involving data-driven anomaly detection based monitoring techniques, we aimed to recognize anomalies in the influent, make decisions, and take action before they flow into the process, and upgrade sustainability of the plant.

https://t.me/learningnets

Descriptive statistics of ICs time series data are given in Table 2.3. To statistically characterize each variable, i.e., mean value, standard deviation, extremes, and quartiles are computed. Also, symmetry and shape of the distribution of each variable are checked by computing skewness and kurtosis statistics.

The training data collected from WWTP in the absence of anomalies is autoscaled and utilized to design the reference PCA model. Here, the retained PCs in the PCA model are selected using the CPV technique. Seven PCs catching 80.01% of the variability in the data have been kept in the development of the PCA model (Fig. 2.13).

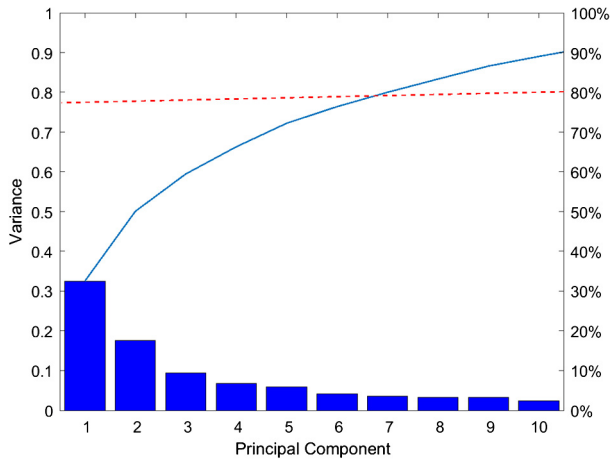


FIGURE 2.13 Illustration of the cumulative percent variance. Seven PCs explained 80.01% of the total variance.

Fig. 2.14 displays a heatmap of the transformation matrix between the original variable and the retained PCs. This heatmap enables visual identification of relations between the original IMs variables and the PCs. Fig. 2.15 shows a correlation heatmap of Pearson correlation results for the IMs variables. The dominant PC, which explains 32.54% of total dataset variance, is linked to the "Flow" and "CNP" blocks. The second dominant PC capturing 17.55% of the total variance is related to the calcium hardness and the "TDS" block. The third PC accounts for 9.39% of the variability in the data and is related to the "CNP" and "Flow" blocks. The fourth PC, which captures 6.78% of the total variability, is impacted by the "pH" and "miscellaneous" blocks. The other retained PCs account for 5.94%, 4.20%, and 3.61% of variance, respectively.

To show the quality of the designed PCA model, Fig. 2.16 presents the plots of the typical variables with their predictions from the PCA model. Fig. 2.16 indicates that the PCA model presents relatively acceptable predictions of the IMs time series. However, for some variables, such as BOD5, some modeling mismatch is noticed. This modeling mismatch is reflected in residuals, which are used as an indicator of fault detection, and can impact the detection quality.

https://t.me/learningnets

TABLE 2.3 A summary of statistics quantitatively describing the training dataframe.

	mean	std	min	0.25	0.5	0.75	max	skewness	kurtosis
InFlow-LS1	3021.61	535.45	2228.00	2660.00	2851.00	3244.00	5249.00	1.33	1.89
InFlow-LS8	279.31	156.08	64.00	156.00	228.00	366.00	867.00	1.13	0.92
InFlow-DP	47.43	95.79	0.00	9.00	10.00	36.00	749.00	3.80	18.67
InFlow-Total	3512.92	611.00	2558.00	3036.00	3389.00	3853.00	5642.00	0.86	0.34
Temp	29.31	1.59	25.99	28.10	29.20	30.60	32.50	0.27	−0.85
pH	7.40	0.25	6.59	7.25	7.36	7.52	8.56	0.91	3.20
Conductivity	669.37	284.69	264.00	537.00	625.00	719.00	2466.00	3.19	15.37
TDS	459.48	209.08	174.00	363.00	429.00	492.00	1809.00	3.37	16.69
TSS	68.66	27.29	12.00	49.00	64.00	82.00	187.00	1.14	2.10
CaHardness	72.75	30.78	20.00	52.00	72.00	94.00	176.00	0.49	0.17
MgHardness	41.91	27.46	6.00	24.00	36.00	48.00	156.00	1.81	3.59
TotalAlkalinity	120.88	24.98	68.00	100.00	120.00	136.00	196.00	0.22	−0.33
BOD_5	99.03	36.95	27.00	71.00	92.00	123.00	224.00	0.58	0.10
COD	152.99	61.83	42.00	102.00	157.00	189.00	329.00	0.50	−0.25
FOG	54.36	53.05	2.90	14.30	37.10	77.10	351.40	1.86	5.25
TKN	17.91	6.18	2.10	13.80	17.30	21.90	37.90	0.30	0.45
NH_3N	11.84	4.10	0.94	9.30	12.00	14.50	23.60	−0.06	0.47
NO_3N	4.17	1.68	0.10	2.90	4.20	5.10	9.80	0.49	0.47
PO_4P	8.25	2.86	1.30	6.40	8.10	10.00	23.50	1.41	6.59
Cl	126.09	75.62	45.00	91.00	107.00	137.00	654.00	4.29	23.37
Boron	1.15	0.33	0.50	0.90	1.10	1.30	2.50	1.40	2.63

https://t.me/learningnets

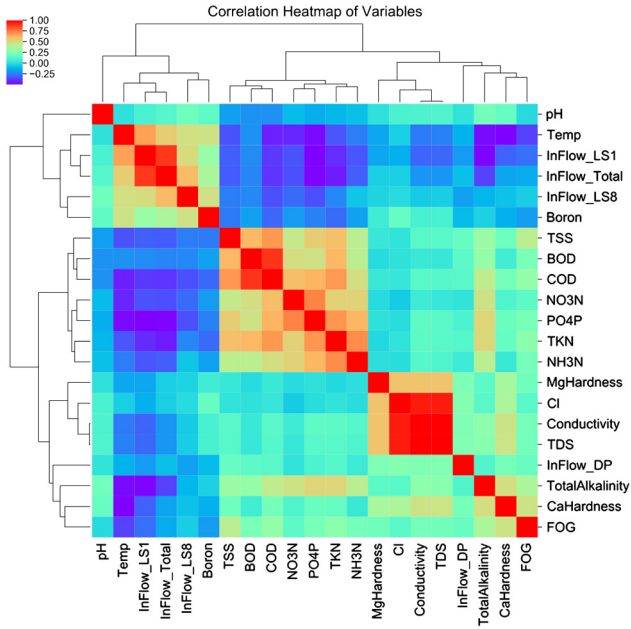


FIGURE 2.14 Heatmap of the Pearson correlation coefficients computed for the IC variables.

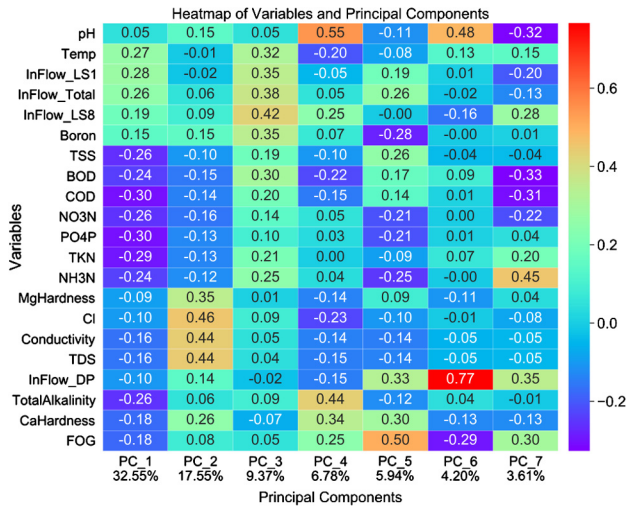


FIGURE 2.15 Heatmap of variables and the retained PCs.

The designed PCA model that reflects the nominal behavior of the IMs at the WWTP is used for fault detection purposes. IM testing data collected from May 15, 2011, to September 1, 2017, were used to test the PCA-based moni-

https://t.me/learningnets

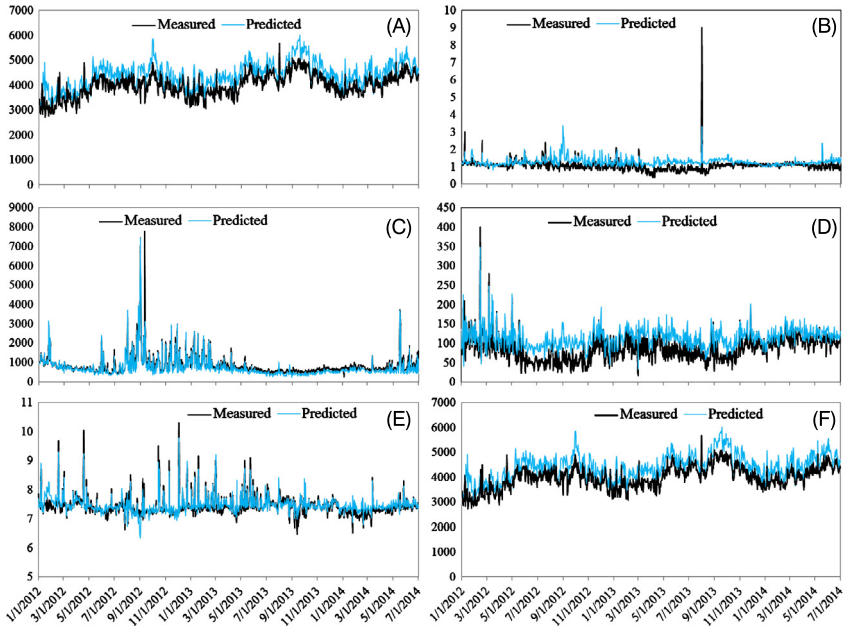


FIGURE 2.16 Time series of typical variables with their predictions produced by PCA modeling: (A) total inflow, (B) boron, (C) conductivity, (D) BOD5, (E) pH, and (F) total alkalinity.

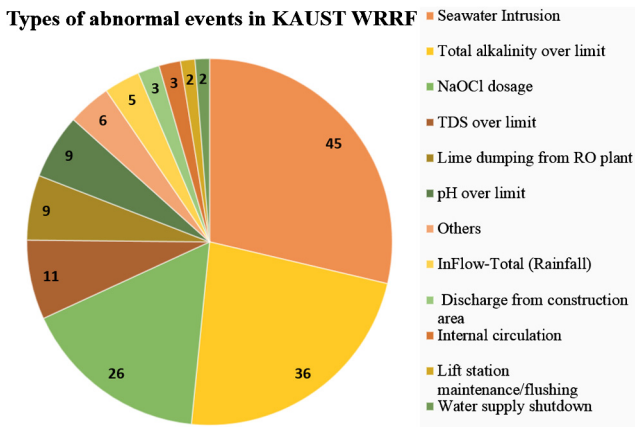


FIGURE 2.17 Anomalies occurred in KAUST WWTP.

toring scheme. Several abnormal events have been reported during this period, including seawater intrusion and discharge from construction area over the limit and leakage (Fig. 2.17).

The designed PCA model that reflects the healthy behavior of the WWTP will be used for anomaly detection purposes. Anomalies in ICs, often raised

https://t.me/learningnets

by abnormal events, need to be monitored and detected promptly to improve system resilience. Here, the capacity of the PCA-based SPE, T^2, and residuals-based univariate chart in monitoring ICs are tested. For each scheme, parametric and nonparametric thresholds are adopted to improve anomaly detection. The nonparametric threshold is computed as the $(1-\alpha)$th quartile of the estimated distribution of the decision statistic (e.g., Q and T^2) using the KDE technique. To quantitatively assess the detection efficiency of the proposed procedures, the following metrics will be used: true positive rate (TPR), false positive rate (FPR), accuracy, and area under the curve (AUC) [147]. Detection results of the five monitoring schemes are summarized in Table 2.4.

TABLE 2.4 Accuracy of the PCA-KNN vs others.

Algorithm	TPR	FPR	Precision	AUC
PCA-SPE^{np}	0.853	0.057	0.939	0.898
PCA-T_p^2	0.657	0.059	0.928	0.799
PCA-T_{np}^2	0.402	0.005	0.969	0.699
PCA-Residuals$_{np}$	0.765	0.533	0.480	0.616
PCA-SPE$_p$	0.196	0.001	0.964	0.598

From the detection results in Table 2.4, we realize that the PCA-based residuals scheme is ineffective for monitoring multivariate IM data. In this approach, each single PCA residual is monitored individually by a nonparametric univariate monitoring scheme. The joint monitoring scheme declares the presence of an abnormal event if at least one individual scheme signals' anomalies. This approach does not consider the correlation among variables and thus results in weak detection results.

As clearly shown, the detection capacity is greatly enhanced by using the PCA-based SPE with nonparametric threshold compared to the corresponding Gaussian distribution-based thresholds. Indeed, the decision thresholds of SPE and T^2 schemes are computed with the assumption that residuals are Gaussian distributed, which is invalid for IMs data (Table 2.4). Indeed, the PCA-based SPE scheme with nonparametric threshold outperforms the other methods (Table 2.4) by achieving an AUC up to 0.898, which flags its ability to detect the vast majority of abnormal events reported by the operator, while avoiding raising false alarms at the same time. The lowest prediction performance was obtained for the PCA-based SPE scheme with parametric thresholds since SPE statistic, which is sensitive to model errors.

The anomalies that are challenging to detect based on the overall outcome of our algorithms are related to "Water supply shutdown" and "Lift station maintenance/flushing". This is because they may raise multiple latent effects in multiple variables (unlike others that would cause significant single variable shifting) and therefore were not easy to be captured.

https://t.me/learningnets

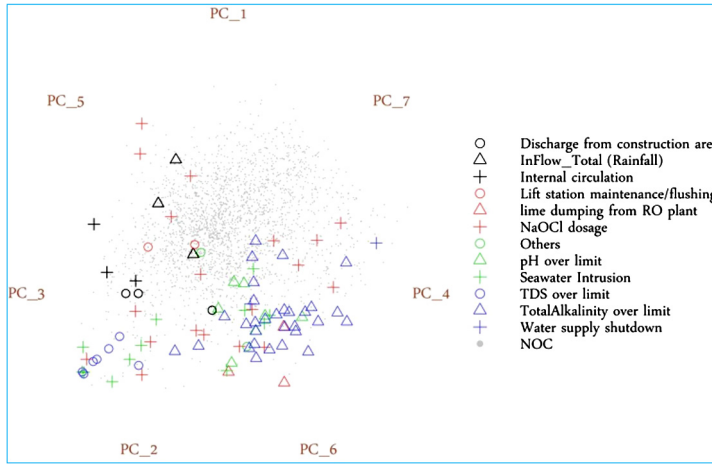


FIGURE 2.18 RadViz of IMs datasets with anomalies.

It has been shown in [148] that the PCA-based fault detection can be further improved by using k-nearest neighbors (kNN) algorithm to separate between normal and abnormal features in IMs dataset. The use of kNN is motivated by its remarkable success in quantifying the similarity between normal and abnormal features due to its capability to deal with nonlinear features, and does not involve assumptions on the underlying data distribution.

It should be clear that the PCA-based monitoring schemes are affected by the prediction quality of the designed model. The detection results in this application highlight that building a nonlinear PCA model may enhance the detection abilities.

Fig. 2.18 displays a RadViz visualization of IMs datasets with their scores on principal components set as anchors. Here, the dimensionality reduction benefits of PCA and the desirable proprieties of multivariate data visualization via RadViz have been combined. Moreover, prior knowledge of PCs composition from Fig. 2.15 provided a clear interpretation and diagnosis of detected anomalies. Observations without anomalies (in gray) are scattered in the center region, and numerous anomalies from the second to the sixth PC surrounded them in a "V" shape manner. RadViz is used here to assist analysis of detected anomalies and identifying their possible sources. From Fig. 2.15, it can be seen that anomalies generated by "discharge from construction area" and "TDS over the limit" are close to the second and third anchors, exactly matching their roles as "TDS" and "Inflow" block-indicators. Also, we can see that observations from "Rainfall", "Internal circulation", and "Lift station flushing" are near to the "Inflow" and the PC_5 which is dominated by FOG and recycled flow. IM abnormalities caused by "Lime dumping from RO plant", "pH over the limit", "Total alkalinity over the limit", and "Water supply shutdown" are closer to PC_6, which is dominated by

https://t.me/learningnets

pH, PC_2 anchor and PC_4, which is dominated by pH, alkalinity, and calcium hardness anchor.

Note that RadViz is helpful to understand and identify the source of anomalies.

2.7 Discussion

Prediction, fault detection, and diagnosis using LVR methods are effective for handling massive and high-dimensional data. As explained in this chapter, the linear LVR models represent an important tool and have a good capacity to extract relevant information from multivariate data. These models are amalgamated with univariate and multivariate monitoring schemes for detecting anomalies in multivariate data. However, these models are designed to model linear relationships, and in practice, modern industry and environmental processes exhibit nonlinear behaviors. This has been shown in the WWTP case study, where the data collected is non-Gaussian distributed and nonlinear. The linear PCA-based fault detection approach can achieve a detection rate of around 90%, which may be improved by using nonlinear monitoring schemes. Accordingly, the nonlinear LVR approach techniques that can describe and capture nonlinearity in multivariate processes is needed. Chapter 4 will focus on nonlinear process monitoring.

Although Q and T^2 and their contribution plots are frequently employed in fault detection and isolation, they frequently lead to the incorrect isolation of faults. Indeed, Q reduces the degrees of freedom of the test statistics. Commonly one-half to two-thirds of the diagnosis information (redundancy) in the data is lost using Q. The Q contribution plot method does not possess a convincing statistical basis and, in fact, sometimes gives an ambiguous diagnosis. Numerous lasso and forward variable selection methods have been proposed for fault attribution. The next chapter will review these techniques and present a new fused adaptive lasso fault attribution approach that is designed for nonstationary, temporally dependent multivariate processes.

References

[1] V. Venkatasubramanian, R. Rengaswamy, K. Yin, S.N. Kavuri, A review of process fault detection and diagnosis: part I: quantitative model-based methods, Computers & Chemical Engineering 27 (3) (2003) 293–311.

[2] V. Venkatasubramanian, R. Rengaswamy, S.N. Kavuri, K. Yin, A review of process fault detection and diagnosis: part III: process history based methods, Computers & Chemical Engineering 27 (3) (2003) 327–346.

[3] H. Haimi, M. Mulas, F. Corona, R. Vahala, Data-derived soft-sensors for biological wastewater treatment plants: an overview, Environmental Modelling & Software 47 (2013) 88–107.

[4] S. Yin, X. Li, H. Gao, O. Kaynak, Data-based techniques focused on modern industry: an overview, IEEE Transactions on Industrial Electronics 62 (1) (2015) 657–667.

[5] F. Harrou, M. Madakyaru, Y. Sun, S. Khadraoui, Improved detection of incipient anomalies via multivariate memory monitoring charts: application to an air flow heating system, Applied Thermal Engineering 109 (2016) 65–74.

https://t.me/learningnets

[6] I. Nimmo, Adequately address abnormal operations, Chemical Engineering Progress 91 (9) (1995) 36–45.
[7] D.J. Hill, B.S. Minsker, Anomaly detection in streaming environmental sensor data: a data-driven modeling approach, Environmental Modelling & Software 25 (9) (2010) 1014–1022.
[8] A.G. Capodaglio, H.V. Jones, V. Novotny, X. Feng, Sludge bulking analysis and forecasting: application of system identification and artificial neural computing technologies, Water Research 25 (10) (1991) 1217–1224.
[9] M. Madakyaru, M.N. Nounou, H.N. Nounou, Linear inferential modeling: theoretical perspectives, extensions, and comparative analysis, Intelligent Control and Automation 3 (04) (2012) 376.
[10] S. Yin, S.X. Ding, X. Xie, H. Luo, A review on basic data-driven approaches for industrial process monitoring, IEEE Transactions on Industrial Electronics 61 (11) (2014) 6418–6428.
[11] I. Frank, J. Friedman, A statistical view of some chemometric regression tools, Technometrics 35 (2) (1993) 109–148.
[12] S. Yin, S.X. Ding, A. Haghani, H. Hao, P. Zhang, A comparison study of basic data-driven fault diagnosis and process monitoring methods on the benchmark Tennessee Eastman process, Journal of Process Control 22 (9) (2012) 1567–1581.
[13] F. Harrou, Y. Sun, M. Madakyaru, B. Bouyedou, An improved multivariate chart using partial least squares with continuous ranked probability score, IEEE Sensors Journal 18 (16) (2018) 6715–6726.
[14] M. Stone, R.J. Brooks, Continuum regression: cross-validated sequentially constructed prediction embracing ordinary least squares, partial least squares and principal components regression, Journal of the Royal Statistical Society, Series B 52 (2) (1990) 237.
[15] J. Fox, Applied Regression Analysis and Generalized Linear Models, Sage Publications, 2015.
[16] N.R. Draper, H. Smith, Applied Regression Analysis, vol. 326, John Wiley & Sons, 2014.
[17] A. Hoerl, R. Kennard, Ridge regression based estimation for nonorthogonal problems, Technometrics 8 (1970) 27–52.
[18] A.E. Hoerl, R.W. Kennard, Ridge regression iterative estimation of the biasing parameter, Communications in Statistics. Theory and Methods 5 (1) (1976) 77–88.
[19] A.E. Hoerl, R.W. Kannard, K.F. Baldwin, Ridge regression: some simulations, Communications in Statistics. Theory and Methods 4 (2) (1975) 105–123.
[20] B.R. kowalski, M.B. Seasholtz, Recent developments in multivariate calibration, Journal of Chemometrics 5 (1991) 129–145.
[21] J. MacGregor, T. Kourti, Statistical process control of multivariate processes, Control Engineering Practice 3 (3) (1995).
[22] S. Qin, Statistical process monitoring: basics and beyond, Journal of Chemometrics 17 (8/9) (2003) 480–502.
[23] B.M. Wise, N.B. Gallagher, The process chemometrics approach to process monitoring and fault detection, Journal of Process Control 6 (6) (1996) 329–348.
[24] K. Pearson, LIII. On lines and planes of closest fit to systems of points in space, The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science 2 (11) (1901) 559–572.
[25] H. Hotelling, Analysis of a complex of statistical variables into principal components, Journal of Educational Psychology 24 (6) (1933) 417.
[26] H. Abdi, L.J. Williams, Principal component analysis, Wiley Interdisciplinary Reviews: Computational Statistics 2 (4) (2010) 433–459.
[27] D. Báscones, C. González, D. Mozos, Hyperspectral image compression using vector quantization, PCA and JPEG2000, Remote Sensing 10 (6) (2018) 907.
[28] A. Subasi, M.I. Gursoy, EEG signal classification using PCA, ICA, LDA and support vector machines, Expert Systems with Applications 37 (12) (2010) 8659–8666.
[29] F. Harrou, L. Fillatre, M. Bobbia, I. Nikiforov, Statistical detection of abnormal ozone measurements based on constrained generalized likelihood ratio test, in: 52nd IEEE Conference on Decision and Control, IEEE, 2013, pp. 4997–5002.

https://t.me/learningnets

[30] P. Ralston, G. DePuy, J.H. Graham, Computer-based monitoring and fault diagnosis: a chemical process case study, ISA Transactions 40 (1) (2001) 85–98.
[31] M. Zhu, A. Ghodsi, Automatic dimensionality selection from the scree plot via the use of profile likelihood, Computational Statistics & Data Analysis 51 (2) (2006) 918–930.
[32] I. Jolliffe, Principal Component Analysis, Springer, 2011.
[33] B. Li, J. Morris, E.B. Martin, Model selection for partial least squares regression, Chemometrics and Intelligent Laboratory Systems 64 (1) (2002) 79–89.
[34] S. Wold, Cross-validatory estimation of the number of components in factor and principal components models, Technometrics 20 (4) (1978) 397.
[35] P. Geladi, B.R. Kowalski, Partial least square regression: a tutorial, Analytica Chimica Acta 185 (1986) 1–17.
[36] A. Phatak, S. De Jong, The geometry of partial least squares, Journal of Chemometrics: A Journal of the Chemometrics Society 11 (4) (1997) 311–338.
[37] Å. Jansson, J. Röttorp, M. Rahmberg, Development of a software sensor for phosphorus in municipal wastewater, Journal of Chemometrics: A Journal of the Chemometrics Society 16 (8–10) (2002) 542–547.
[38] P. Geladi, B.R. Kowalski, Partial least-squares regression: a tutorial, Analytica Chimica Acta 185 (1986) 1–17.
[39] S.J. Qin, Survey on data-driven industrial process monitoring and diagnosis, Annual Reviews in Control 36 (2) (2012) 220–234.
[40] Y. Wang, Y. Wei, T. Liu, T. Sun, K.T. Grattan, TDLAS detection of propane/butane gas mixture by using reference gas absorption cells and partial least square approach, IEEE Sensors Journal 18 (20) (2018) 8587–8596.
[41] Y. Hiroyuki, Y.B. Hideki, F.C.E.O. Hiromu, F. Hideki, Canonical correlation analysis for multivariate regression and its application to metabolic fingerprinting, Biochemical Engineering Journal 40 (2008) 199–204.
[42] S. Wold, Soft Modeling: The Basic Design and Some Extensions, Systems under Indirect Observations, Elsevier, Amsterdam, 1982.
[43] S.J. Qin, Y. Dong, Data distillation, analytics, and machine learning, in: Proceedings of the 2017 CPC/FOCAPO, Jan. 8–12, 2017, Tucson, Arizona, 2017.
[44] L. Sun, S. Ji, S. Yu, J. Ye, On the equivalence between canonical correlation analysis and orthonormalized partial least squares, in: Twenty-First International Joint Conference on Artificial Intelligence, 2009.
[45] S.J. Qin, Y. Zheng, Quality-relevant and process-relevant fault monitoring with concurrent projection to latent structures, AIChE Journal 59 (2) (2013) 496–504.
[46] W. Ku, R.H. Storer, C. Georgakis, Disturbance detection and isolation by dynamic principal component analysis, Chemometrics and Intelligent Laboratory Systems 30 (1) (1995) 179–196.
[47] F. Tsung, Statistical monitoring and diagnosis of automatic controlled processes using dynamic PCA, International Journal of Production Research 38 (3) (2000) 625–637.
[48] Y. Dong, S.J. Qin, A novel dynamic PCA algorithm for dynamic data modeling and process monitoring, Journal of Process Control 67 (2018) 1–11.
[49] N.L. Ricker, The use of biased least-squares estimators for parameters in discrete-time pulse-response models, Industrial & Engineering Chemistry Research 27 (2) (1988) 343–350.
[50] M.H. Kaspar, W.H. Ray, Dynamic PLS modelling for process control, Chemical Engineering Science 48 (20) (1993) 3447–3461.
[51] S. Lakshminarayanan, S.L. Shah, K. Nandakumar, Modeling and control of multivariable processes: dynamic PLS approach, AIChE Journal 43 (9) (1997) 2307–2322.
[52] Y. Park, A statistical process control approach for network intrusion detection, PhD dissertation, Georgia Institute of Technology, 2005.
[53] M. Frisén, Optimal sequential surveillance for finance, public health, and other areas, Sequential Analysis 28 (3) (2009) 310–337.

https://t.me/learningnets

[54] F. Kadri, F. Harrou, S. Chaabane, Y. Sun, C. Tahon, Seasonal ARMA-based SPC charts for anomaly detection: application to emergency department systems, Neurocomputing 173 (2016) 2102–2114.
[55] W. Shewhart, Economic quality control of manufactured product, The Bell System Technical Journal 2 (1930) 364–389.
[56] D.C. Montgomery, Introduction to Statistical Quality Control, John Wiley&Sons, New York, 2005.
[57] E. Page, Continuous inspection schemes, Biometrika 41 (1–2) (1954).
[58] A. Cinar, A. Palazoglu, F. Kayihan, Chemical Process Performance Evaluation, 1st ed., CRC Press, Boca Raton, FL, 2007.
[59] D.C. Montgomery, Introduction to Statistical Quality Control, John Wiley & Sons, 2007.
[60] J.M. Lucas, M.S. Saccucci, Exponentially weighted moving average control schemes: properties and enhancements, Technometrics 32 (1) (1990) 1–12.
[61] S.W. Roberts, Control chart tests based on geometric moving averages, Technometrics 1 (3) (1959) 239–250.
[62] J.S. Hunter, The exponentially weighted moving average, Journal of Quality Technology 18 (4) (1986) 203–210.
[63] G. Capizzi, G. Masarotto, An adaptive exponentially weighted moving average control chart, Technometrics 45 (3) (2003) 199–207.
[64] W. Jiang, L. Shu, D.W. Apley, Adaptive CUSUM procedures with EWMA-based shift estimators, IIE Transactions 40 (10) (2008) 992–1003.
[65] R.S. Sparks, CUSUM charts for signalling varying location shifts, Journal of Quality Technology 32 (2) (2000) 157–171.
[66] F. Harrou, L. Fillatre, I. Nikiforov, Anomaly detection/detectability for a linear model with a bounded nuisance parameter, Annual Reviews in Control 38 (1) (2014) 32–44.
[67] M. Basseville, I.V. Nikiforov, et al., Detection of Abrupt Changes: Theory and Application, vol. 104, Prentice Hall, Englewood Cliffs, 1993.
[68] T.S. Ferguson, Mathematical Statistics: A Decision Theoretic Approach, Vol. 1, Academic Press, 2014.
[69] E.L. Lehmann, J.P. Romano, Testing Statistical Hypotheses, Springer Science & Business Media, 2006.
[70] T.A. Severini, An approximation to the modified profile likelihood function, Biometrika 85 (2) (1998) 403–411.
[71] Y. Pawitan, In All Likelihood: Statistical Modelling and Inference using Likelihood, Oxford University Press, 2001.
[72] G. Capizzi, G. Masarotto, Practical design of generalized likelihood ratio control charts for autocorrelated data, Technometrics 50 (3) (2008) 357–370.
[73] L.C. Alwan, Effects of autocorrelation on control chart performance, Communications in Statistics. Theory and Methods 21 (4) (1992) 1025–1049.
[74] L.C. Alwan, H.V. Roberts, Time-series modeling for statistical process control, Journal of Business & Economic Statistics 6 (1) (1988) 87–95.
[75] D.G. Wardell, H. Moskowitz, R.D. Plante, Run-length distributions of residual control charts for autocorrelated processes, Journal of Quality Technology 26 (1994) 308–317.
[76] D. Montgomery, C. Mastrangelo, F.W. Faltin, W.H. Woodall, J.F. MacGregor, T.P. Ryan, Some statistical process control methods for autocorrelated data, Journal of Quality Technology 23 (3) (1991).
[77] G.C. Runger, S.S. Prabhu, A Markov chain model for the multivariate exponentially weighted moving averages control chart, Journal of the American Statistical Association 91 (436) (1996) 1701–1706.
[78] J.N. Dyer, B.M. Adams, M.D. Conerly, The reverse moving average control chart for monitoring autocorrelated processes, Journal of Quality Technology 35 (2) (2003) 139–152.
[79] E.G. Schilling, P.R. Nelson, The effect of non-normality on the control limits of $\bar{X}$ charts, Journal of Quality Technology 8 (4) (1976).

https://t.me/learningnets

[80] S.A. Yourstone, W.J. Zimmer, Non-normality and the design of control charts for averages, Decision Sciences 23 (5) (1992) 1099–1113.
[81] P.M. Burrows, $\bar{X}$ control schemes for a production variable with skewed distribution, Journal of the Royal Statistical Society. Series D. The Statistician 12 (4) (1962) 296–312.
[82] B. Laungrungrong, C.M. Borror, D.C. Montgomery, EWMA control charts for multivariate Poisson-distributed data, International Journal of Quality Engineering and Technology 2 (3) (2011) 185–211.
[83] B. Laungrungrong, C.M. Borror, D.C. Montgomery, A one-sided MEWMA control chart for Poisson-distributed data, International Journal of Data Analysis Techniques and Strategies 6 (1) (2014) 15–42.
[84] C. Çiflikli, Development of univariate control charts for non-normal data, PhD thesis, İzmir Institute of Technology, 2006.
[85] N. Singh, R. Agrawal, Combination of Kullback–Leibler divergence and Manhattan distance measures to detect salient objects, Signal, Image and Video Processing 9 (2) (2015) 427–435.
[86] A. Karine, A. Toumi, A. Khenchaf, M. El Hassouni, Target recognition in radar images using weighted statistical dictionary-based sparse representation, IEEE Geoscience and Remote Sensing Letters 14 (12) (2017) 2403–2407.
[87] A. Zeroual, F. Harrou, Y. Sun, N. Messai, Integrating model-based observer and Kullback–Leibler metric for estimating and detecting road traffic congestion, IEEE Sensors Journal 18 (20) (2018) 8605–8616.
[88] D. Olszewski, Fraud detection in telecommunications using Kullback–Leibler divergence and latent Dirichlet allocation, in: International Conference on Adaptive and Natural Computing Algorithms, Springer, 2011, pp. 71–80.
[89] F. Harrou, Y. Sun, M. Madakyaru, Kullback–Leibler distance-based enhanced detection of incipient anomalies, Journal of Loss Prevention in the Process Industries 44 (2016) 73–87.
[90] J. Harmouche, C. Delpha, D. Diallo, Y. Le Bihan, Statistical approach for nondestructive incipient crack detection and characterization using Kullback–Leibler divergence, IEEE Transactions on Reliability 65 (3) (2016) 1360–1368.
[91] A.S. Leonard, D.B. Weissman, B. Greenbaum, E. Ghedin, K. Koelle, Transmission bottleneck size estimation from pathogen deep-sequencing data, with an application to human influenza A virus, Journal of Virology 91 (14) (2017) e00171-17.
[92] L. Aggoune, Y. Chetouani, T. Raïssi, Fault detection in the distillation column process using Kullback–Leibler divergence, ISA Transactions 63 (2016) 394–400.
[93] J. Zeng, U. Kruger, J. Geluk, X. Wang, L. Xie, Detecting abnormal situations using the Kullback–Leibler divergence, Automatica 50 (11) (2014) 2777–2786.
[94] L. Pardo, Statistical Inference Based on Divergence Measures, Chapman and Hall/CRC, 2005.
[95] I. Csiszár, P.C. Shields, et al., Information theory and statistics: a tutorial, Foundations and Trends® in Communications and Information Theory 1 (4) (2004) 417–528.
[96] G. Ditzler, R. Polikar, Hellinger distance based drift detection for nonstationary environments, in: 2011 IEEE Symposium on Computational Intelligence in Dynamic and Uncertain Environments (CIDUE), IEEE, 2011, pp. 41–48.
[97] C. Li, B. Huang, F. Qian, Hellinger distance based probability distribution approach to performance monitoring of nonlinear control systems, Chinese Journal of Chemical Engineering 23 (12) (2015) 1945–1950.
[98] M. Basseville, Divergence measures for statistical data processing—an annotated bibliography, Signal Processing 93 (4) (2013) 621–633.
[99] D.I.R. González, J.-B. Hayet, Fast human detection in RGB-D images with progressive SVM-classification, in: Pacific-Rim Symposium on Image and Video Technology, Springer, 2013, pp. 337–348.
[100] A.P. Korostelev, A.B. Tsybakov, Minimax Theory of Image Reconstruction, vol. 82, Springer Science & Business Media, 2012.
[101] V. González-Castro, R. Alaiz-Rodríguez, E. Alegre, Class distribution estimation based on the Hellinger distance, Information Sciences 218 (2013) 146–164.

https://t.me/learningnets

[102] L. Aggoune, Y. Chetouani, H. Radjeai, Change detection in a distillation column using non-linear auto-regressive moving average with exogenous input model and Hellinger distance, IET Science, Measurement & Technology 10 (1) (2016) 10–17.
[103] J. Tajer, A. Makke, O. Salem, A. Mehaoua, A comparison between divergence measures for network anomaly detection, in: 2011 7th International Conference on Network and Service Management, IEEE, 2011, pp. 1–5.
[104] K. Yamanishi, J.-I. Takeuchi, G. Williams, P. Milne, On-line unsupervised outlier detection using finite mixtures with discounting learning algorithms, Data Mining and Knowledge Discovery 8 (3) (2004) 275–300.
[105] D. Romano, M. Kinnaert, Robust fault detection and isolation based on the Kullback divergence, IFAC Proceedings Volumes 39 (13) (2006) 426–431.
[106] J. Havrda, F. Chárvat, Quantification method of classification processes. The concept of structural α-entropy, Kybernetika 3 (1967) 30.
[107] P.N. Rathie, P. Kannappan, A directed-divergence function of type β, Information and Control 20 (1) (1972) 38–45.
[108] A.A. Borovkov, Mathematical Statistics, Gordon and Breach, Amsterdam, 1998.
[109] R.B. Crosier, Multivariate generalizations of cumulative sum quality-control schemes, Technometrics 30 (3) (1988) 291–303.
[110] H. Hotelling, Multivariate Quality Control Illustrated by the Air Testing of Sample Bomb Sights, Techniques of Statistical Analysis, Chap. II, McGraw-Hill, New York, 1947.
[111] J.J. Pignatiello Jr, G.C. Runger, Comparisons of multivariate CUSUM charts, Journal of Quality Technology 22 (3) (1990) 173–186.
[112] R.B. Crosier, A new two-sided cumulative sum quality control scheme, Technometrics 28 (3) (1986) 187–194.
[113] D. Hawkins, Multivariate quality control based on regression-adjusted variables, Technometrics 33 (1) (1991) 61–75.
[114] J. Healy, A note on multivariate CUSUM procedures, Technometrics 29 (4) (1987) 409–412.
[115] C.A. Lowry, W.H. Woodall, C.W. Champ, S.E. Rigdon, A multivariate exponentially weighted moving average control chart, Technometrics 34 (1) (1992) 46–53.
[116] H.-H. Yang, M.-L. Huang, C.-M. Lai, J.-R. Jin, An approach combining data mining and control charts-based model for fault detection in wind turbines, Renewable Energy 115 (2018) 808–816.
[117] F. Harrou, Y. Sun, B. Taghezouit, A. Saidi, M.-E. Hamlati, Reliable fault detection and diagnosis of photovoltaic systems based on statistical monitoring approaches, Renewable Energy 116 (2018) 22–37.
[118] F. Harrou, F. Kadri, S. Khadraoui, Y. Sun, Ozone measurements monitoring using data-based approach, Process Safety and Environmental Protection 100 (2016) 220–231.
[119] S.E. Rigdon, An integral equation for the in-control average run length of a multivariate exponentially weighted moving average control chart, Journal of Statistical Computation and Simulation 52 (4) (1995) 351–365.
[120] K.M. Bodden, S.E. Rigdon, A program for approximating the in-control ARL for the MEWMA chart, Journal of Quality Technology 31 (1) (1999) 120–123.
[121] S.S. Prabhu, G.C. Runger, Designing a multivariate EWMA control chart, Journal of Quality Technology 29 (1) (1997) 8–15.
[122] D.M. Hawkins, S. Choi, S. Lee, A general multivariate exponentially weighted moving-average control chart, Journal of Quality Technology 39 (2) (2007) 118–125.
[123] D.M. Hawkins, E.M. Maboudou-Tchao, Multivariate exponentially weighted moving covariance matrix, Technometrics 50 (2) (2008) 155–166.
[124] C. Matrix, M.R. Reynolds Jr, G.-Y. Cho, Multivariate control charts for monitoring the mean vector and covariance matrix, Journal of Quality Technology 38 (3) (2006) 230–253.
[125] M.R. Reynolds Jr, Z.G. Stoumbos, Combinations of multivariate Shewhart and MEWMA control charts for monitoring the mean vector and covariance matrix, Journal of Quality Technology 40 (4) (2008) 381–393.

https://t.me/learningnets

[126] J. Jackson, G. Mudholkar, Control procedures for residuals associated with principal component analysis, Technometrics 21 (1979) 341–349.
[127] E. Martin, A. Morris, Non-parametric confidence bounds for process performance monitoring charts, Journal of Process Control 6 (6) (1996) 349–358.
[128] F. Harrou, F. Kadri, S. Chaabane, C. Tahon, Y. Sun, Improved principal component analysis for anomaly detection: application to an emergency department, Computers & Industrial Engineering 88 (2015) 63–77.
[129] F. Harrou, Y. Sun, S. Khadraoui, Amalgamation of anomaly-detection indices for enhanced process monitoring, Journal of Loss Prevention in the Process Industries 40 (2016) 365–377.
[130] A. Youssef, C. Delpha, D. Diallo, An optimal fault detection threshold for early detection using Kullback–Leibler divergence for unknown distribution data, Signal Processing 120 (2016) 266–279.
[131] J. Harmouche, C. Delpha, D. Diallo, Incipient fault detection and diagnosis based on Kullback–Leibler divergence using principal component analysis: part I, Signal Processing 94 (2014) 278–287.
[132] F. Harrou, M. Madakyaru, Y. Sun, Improved nonlinear fault detection strategy based on the Hellinger distance metric: plug flow reactor monitoring, Energy and Buildings 143 (2017) 149–161.
[133] R. Isermann, Fault-Diagnosis Systems: an Introduction From Fault Detection to Fault Tolerance, Springer Science & Business Media, 2006.
[134] J.F. MacGregor, C. Jaeckle, C. Kiparissides, M. Koutoudi, Process monitoring and diagnosis by multiblock PLS methods, AIChE Journal 40 (5) (1994) 826–838.
[135] P. Miller, R.E. Swanson, C.E. Heckler, Contribution plots: a missing link in multivariate quality control, Applied mathematics and computer science 8 (4) (1998) 775–792.
[136] C.F. Alcala, S.J. Qin, Analysis and generalization of fault diagnosis methods for process monitoring, Journal of Process Control 21 (3) (2011) 322–330.
[137] J. McGregor, T. Kourti, J. Kresta, Multivariate identification: a study of several methods, in: IFAC ADCHEM Proc., Toulouse, France, vol. 4(2), 1991, pp. 145–156.
[138] T. Kourti, Application of latent variable methods to process control and multivariate statistical process control in industry, International Journal of Adaptive Control and Signal Processing 19 (4) (2005) 213–246.
[139] S. Narasimhan, R. Mah, Generalized likelihood ratio method for Gross error identification, AIChE Journal 33 (9) (1987) 1514–1521.
[140] H.H. Yue, S.J. Qin, Reconstruction-based fault identification using a combined index, Industrial & Engineering Chemistry Research 40 (20) (2001) 4403–4414.
[141] H. Ji, X. He, J. Shang, D. Zhou, Exponential smoothing reconstruction approach for incipient fault isolation, Industrial & Engineering Chemistry Research 57 (18) (2018) 6353–6363.
[142] P. Hoffman, G. Grinstein, D. Pinkney, Dimensional anchors: a graphic primitive for multidimensional multivariate information visualizations, in: Proceedings of the 1999 Workshop on New Paradigms in Information Visualization and Manipulation in Conjunction with the Eighth ACM International Conference on Information and Knowledge Management, ACM, 1999, pp. 9–16.
[143] P. Hoffman, G. Grinstein, K. Marx, I. Grosse, E. Stanley, DNA visual and analytic data mining, in: Proceedings. Visualization'97 (Cat. No. 97CB36155), IEEE, 1997, pp. 437–441.
[144] J. Sharko, G. Grinstein, K.A. Marx, Vectorized RadViz and its application to multiple cluster datasets, IEEE Transactions on Visualization and Computer Graphics 14 (6) (2008) 1444–1451.
[145] O. Yeniay, A. Goktas, A comparison of partial least squares regression with other prediction methods, Hacettepe Journal of Mathematics and Statistics 31 (2002) 99–111.
[146] P.D. Wentzell, L.V. Montoto, Comparison of principal components regression and partial least square regression through generic simulations of complex mixtures, Chemometrics and Intelligent Laboratory Systems 65 (2003) 257–279.

https://t.me/learningnets

[147] F. Harrou, A. Dairi, B. Taghezouit, Y. Sun, An unsupervised monitoring procedure for detecting anomalies in photovoltaic systems using a one-class support vector machine, Solar Energy 179 (2019) 48–58.

[148] T. Cheng, F. Harrou, Y. Sun, T. Leiknes, Monitoring influent measurements at water resource recovery facility using data-driven soft sensor approach, IEEE Sensors Journal 19 (1) (2019) 342–352.

https://t.me/learningnets

Chapter 3

Fault isolation

3.1 Introduction

Monitoring univariate data does not require fault isolation because if a fault is detected, then the only variable affected is the one being monitored. However, multiple features are often monitored jointly in complex systems or processes, and multivariate fault detection does not necessarily provide the user with a list of the subset of variables that have been affected by the fault. Identifying the *shifted* variables is the second in a four-step process that begins with fault detection and ends with returning a system back to its in control (IC) state. First, *fault detection* is used to detect irregularities or anomalies in the system. The performance of a fault detection method is usually measured in terms of detection speed and occurs on the order of seconds, minutes, or hours depending on the temporal frequency of the measurements. *Fault isolation* is used to identify the variables within the system that have been affected by the fault. Oftentimes, the ability of methods to correctly identify the shifted variables is only loosely assessed, but we will review some of the common metrics in this chapter. Determining the cause of the fault occurs in the *fault diagnosis* step, and this could require expert input, especially in a new, understudied system. Finally, *process recovery* is a set of actions needed to return the system to its previous IC state. In some cases, manual intervention is necessary, for example, to make mechanical repairs, but in other cases, adjustments to the control of the system can compensate for the fault.

In this chapter, we focus on presenting fault isolation methods, but fault isolation can occur in multiple contexts, often generated by distinct goals or available data, so the context and goals should be clarified during the initial analysis. For example, fault isolation can be done retrospectively, looking back in time at a period of historical data, or prospectively, analyzing observations as they are made available in real-time. These two goals are often called *Phase I* and *Phase II* monitoring, respectively. Phase I methods in the fault detection context are usually used to test if data are fault-free so that the fault-free data can be used to establish monitoring thresholds and optimal parameter settings for Phase II monitoring. Similar distinctions can be used in fault isolation, but a third option exists in which a period of historical data with a known fault(s) may be examined to determine the cause(s) of the fault without the purpose of carrying information forward into a Phase II analysis. This may occur im-

https://doi.org/10.1016/B978-0-12-819246-7.00009-7
Copyright © 2021 Elsevier Inc. All rights reserved.
https://t.me/learningnets

mediately after a fault has been observed or can be applied to historical data archives.

In terms of available data, we have observed two common settings. In the first, a process may not be well-studied, and the types, number, and frequency of faults that will occur are unknown. This is the classic *unsupervised* setting in which multivariate observations are available, but the observations are not labeled as faulty or not, and the variables are also certainly not labeled as shifted or not. Alternatively, a very well-studied process may be monitored in which a wealth of historical data exists, and multiple copies of every type of fault that could occur have been observed. In such settings, observations are labeled as faulty or not, and the variables associated with each fault are also known. This is a *supervised* setting, so when a new fault is detected, the goal is to classify the new fault as closely as possible to one of the existing catalogued faults.

Throughout this chapter, we attempt to standardize notation and nomenclature. For example, we termed variables that have been affected by the fault as shifted, but many other labels exist in the literature, such as faulty [1], OC [2], changed [3], responsible [4], abnormal [5], suspicious [6], and altered [7]. The variables themselves may simply reflect a change in an external input, so the terms faulty, OC, responsible, suspicious, and abnormal imply that the blame for the fault lies on those particular variables, which may not be the case. We prefer the label shifted, which implies that the variable has changed or altered in some way without assigning any diagnostic responsibility to the variable. Similarly, some methods that claim to perform fault isolation truly focus more on fault detection by removing unimportant variables, so we are very careful with terminology, and the vocabulary that we use may not match that of the original source material, but our goal is to be consistent within this text.

Both statistical and machine learning approaches have been applied in fault isolation. In this chapter, we will present some of the classical approaches to fault isolation along with some more modern tools. Then, we will present some common metrics for evaluating fault isolation methods. Finally, we present two case studies to illustrate some of these methods in practice. However, before we delve into the details of fault isolation, we first point out some fundamental issues that often go overlooked in fault isolation.

3.1.1 Pitfalls of standardizing data

Standardizing variables prior to performing fault detection and isolation (FD&I) is necessary to ensure that those variables with greater ranges or variabilities do not overwhelm the others. Multiple approaches can be taken to standardize variables. We let $\mathbf{X}$ be an $n \times p$ dimensional matrix with n observations on the rows, and p variables on the columns. We may refer to either $\mathbf{x}_i$ for $i = 1, \ldots, n$, which is a p-dimensional set of variables for one observation, or $\mathbf{x}_j$ for $j =$

https://t.me/learningnets

$1, \ldots, p$, which is an n-dimensional set of observations for one variable. The standardization can be either univariate or multivariate. The classical univariate approach is simply

$$\frac{\mathbf{x}_j - \hat{\mu}_j}{\hat{\sigma}_j},$$

for $\hat{\mu}_j$ and $\hat{\sigma}_j$ the sample mean and standard deviation of variable j, respectively, for $j = 1, \ldots, p$. The classic multivariate approach is

$$\hat{\boldsymbol{\Sigma}}^{-1/2}(\mathbf{X} - \hat{\boldsymbol{\mu}}),$$

where $\hat{\boldsymbol{\Sigma}}$ is the estimated variance–covariance matrix, and $\hat{\boldsymbol{\mu}}$ is the estimated vector of means. Robust standardizations can also be considered, as in [8,9], and these utilize robust estimates of the mean and covariance.

When the variables are related, regardless of whether the relationship is linear or nonlinear, then the standardization method used can dramatically change which variables appear to be shifted. In other words, the shifted variables in the raw data may not be preserved in the transformed data. The stronger the relationship between two variables, the stronger the propagation of the fault from one variable to another can be. To illustrate, Fig. 3.1 shows two sets of three variables. The first set of variables, $\mathbf{x}_1$, $\mathbf{x}_2$, and $\mathbf{x}_3$, are linearly related and are simulated from a multivariate normal distribution with mean $\boldsymbol{\mu} = (0, 0, 0)'$ and covariance

$$\boldsymbol{\Sigma} = \begin{bmatrix} 1.00 & 0.15 & -0.30 \\ 0.15 & 1.00 & 0.85 \\ -0.30 & 0.85 & 1.00 \end{bmatrix}. \tag{3.1}$$

The second set of variables are nonlinearly related, governed by the following equation set, as introduced by [10]:

$$\mathbf{y}_1 = \mathbf{s} + \mathbf{e}_1, \quad \mathbf{y}_2 = \mathbf{s}^2 - 3\mathbf{s} + \mathbf{e}_2, \quad \mathbf{y}_3 = -\mathbf{s}^3 + 3\mathbf{s}^2 + \mathbf{e}_3, \tag{3.2}$$

where the noise factors are $\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3 \sim N(\mathbf{0}, 0.01 \times \mathbf{I})$ with $\mathbf{I}$ a 3×3 identity matrix and underlying latent factor $\mathbf{s} \sim \text{Unif}(0.01, 2)$.

A fault is introduced at the vertical dashed line in Figs. 3.2 and 3.3. For the linearly related variables, Fig. 3.2 shows a strong drift fault in the first variable, and in the second column wherein a multivariate standardization has been applied to the data, a slight drift is also apparent in both $\mathbf{x}_2$ and $\mathbf{x}_3$. The first variable is most strongly correlated with the third, so a stronger drift is observed in $\mathbf{x}_3$ versus $\mathbf{x}_2$. However, no such contamination of $\mathbf{x}_2$ and $\mathbf{x}_3$ is present for the univariate standardization. Similarly, Fig. 3.3 shows a sharp downward shift in the second variable of the nonlinearly related variable set. In the multivari-

https://t.me/learningnets

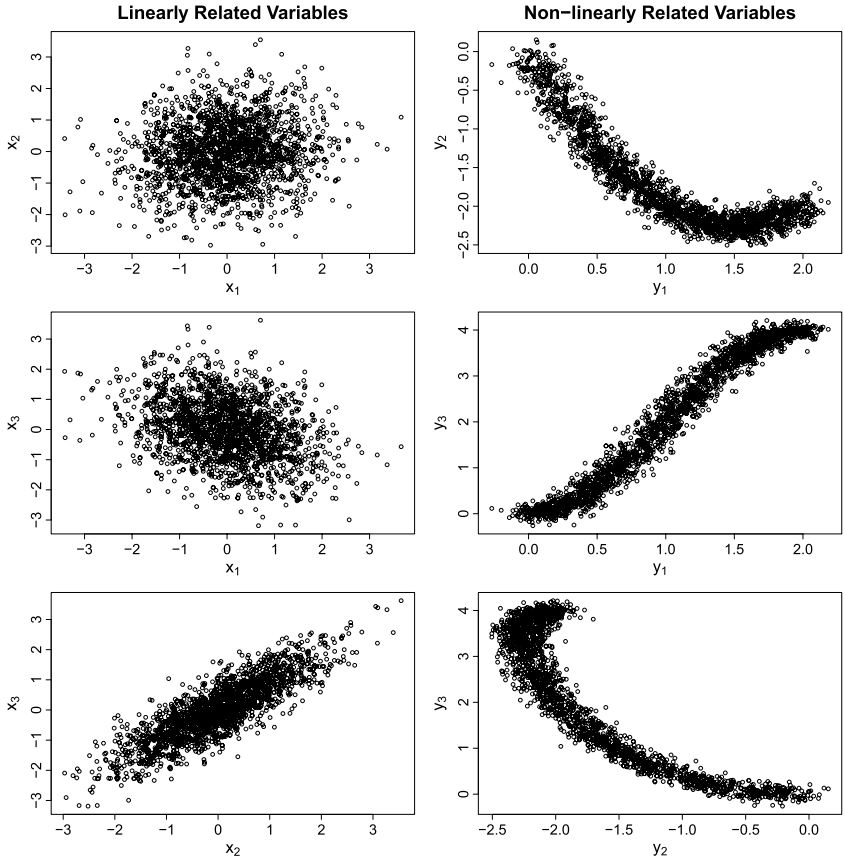


FIGURE 3.1 Pairwise scatterplots of linearly (left) and nonlinearly (right) related variable sets, each from an IC process. On the left, the variables $\mathbf{x}_1$, $\mathbf{x}_2$, and $\mathbf{x}_3$ are simulated from a multivariate normal distribution, and on the right, a set of nonlinear equations is used to generate $\mathbf{y}_1$, $\mathbf{y}_2$, and $\mathbf{y}_3$.

ate standardized data, the strength of the shift in $\mathbf{y}_2$ is dampened, and a shift in $\mathbf{y}_1$ also appears. On the other hand, the univariate standardization preserves the fault and its strength in $\mathbf{y}_2$. Thus, if fault isolation is a primary goal, caution must be taken to ensure that the shifted variables are not distorted by the standardization.

In fact for the multivariate standardization, when the unique symmetric square-root is computed with the eigenvector-eigenvalue decomposition, it makes $\hat{\mathbf{\Sigma}}^{-1/2}$ dense, so a fault affecting only one variable in the original data shifts the mean of all of the variables. If the Cholesky factorization is used, a fault affecting variable k shifts the mean of variables $k, k+1, \ldots, p$. Either way, the fault is not preserved in the original variable unless the fault occurs in only variable p, and the Cholesky factorization is used to obtain $\hat{\mathbf{\Sigma}}^{-1/2}$.

https://t.me/learningnets

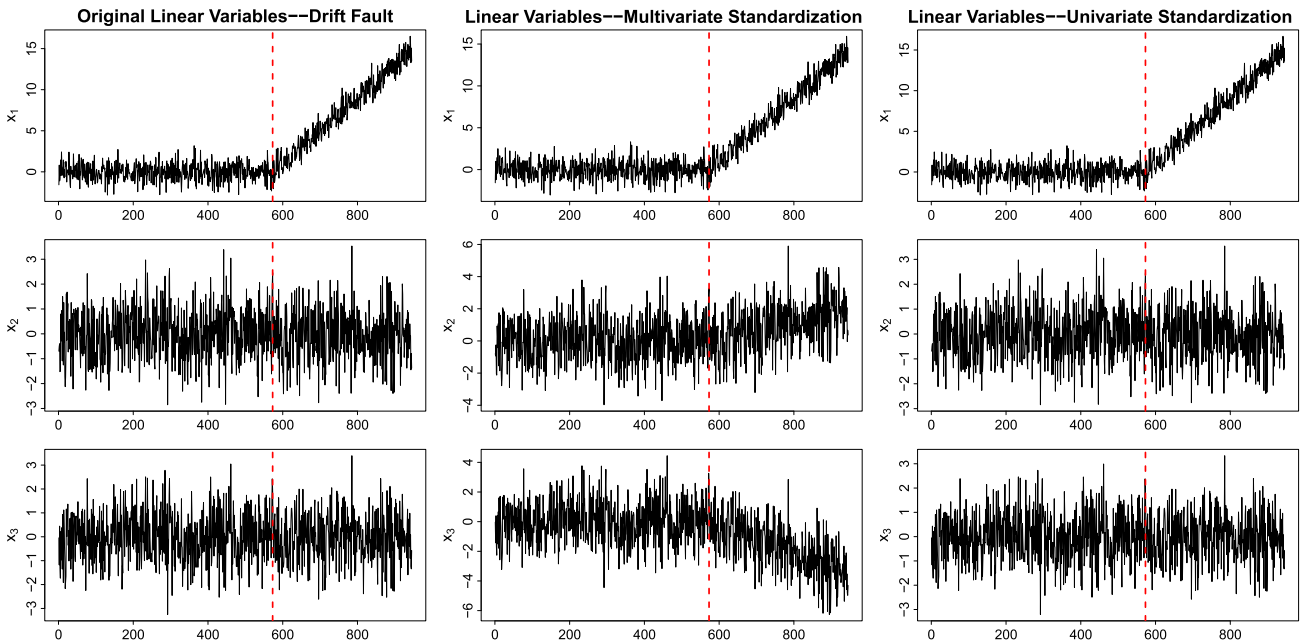


FIGURE 3.2 Linearly related variables with a drift fault in $\mathbf{x}_1$ introduced at vertical dashed line (left). The center and right columns show the same variables after multivariate and univariate standardization, respectively.

https://t.me/learningnets

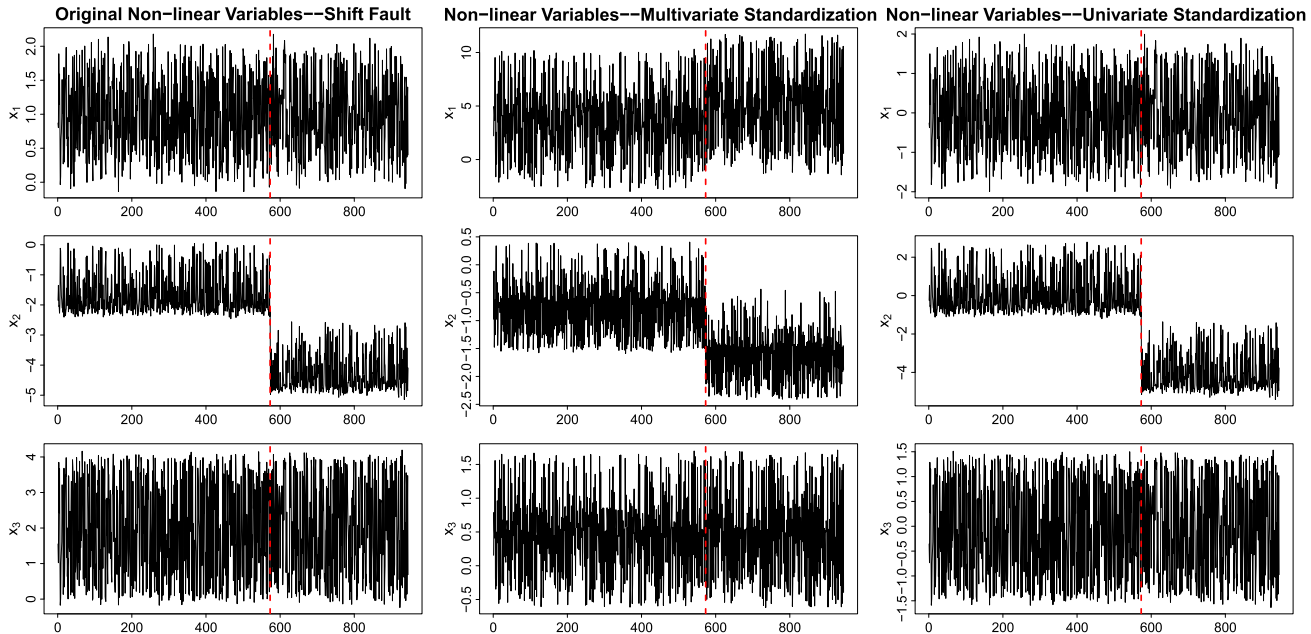


FIGURE 3.3 Nonlinearly related variables with a shift fault in $\mathbf{y}_2$ introduced at vertical dashed line (left). The center and right columns show the same variables after multivariate and univariate standardization, respectively.

https://t.me/learningnets

3.1.2 Shortcomings of contribution plots/scores

A very common first approach to isolating the variables associated with a fault is to use contribution scores or plots that quantify the contribution of each variable to the monitoring statistic [11,12]. Then, those variables with the largest contribution are deemed shifted. With a small number of features and a single shifted variable, this approach is simple to implement and interpret, but it is not advisable for multiple reasons. First, contribution scores are subject to variable smearing. This occurs when variable i shifts, and its shift contaminates the contribution scores of all other variables. Secondly, even under IC conditions, the contributions of all variables are unequal, so misdiagnoses can occur when variables whose contributions are small under IC conditions still do not have the largest contribution in OC conditions.

We will show the variable smearing effect for the Q statistic, using notation adapted from [13]. First, the covariance of $\mathbf{X}$ is estimated with $\mathbf{S} = (1/(n-1))\mathbf{X}'\mathbf{X}$. Then, principal component analysis (PCA) decomposes $\mathbf{S}$ as follows:

$$\mathbf{S} = \mathbf{P}\boldsymbol{\Lambda}\mathbf{P}' + \tilde{\mathbf{P}}\tilde{\boldsymbol{\Lambda}}\tilde{\mathbf{P}}' = \hat{\mathbf{S}} + \tilde{\mathbf{S}},$$

where $\hat{\mathbf{S}} = \mathbf{P}\boldsymbol{\Lambda}\mathbf{P}'$; $\tilde{\mathbf{S}} = \tilde{\mathbf{P}}\tilde{\boldsymbol{\Lambda}}\tilde{\mathbf{P}}'$; $\mathbf{P}$ is a matrix of size $p \times l$ of eigenvectors termed principal loadings; $\tilde{\mathbf{P}}$ is a matrix of size $p \times (p-l)$ of eigenvectors termed residual loadings; l is the number of principal components (PCs) retained to explain the variability in $\mathbf{S}$; and $\boldsymbol{\Lambda}$ and $\tilde{\boldsymbol{\Lambda}}$ are diagonal matrices containing the principal and residual eigenvalues, respectively. An individual measurement, $\mathbf{x}_i$, can be rewritten as $\mathbf{x}_i = \hat{\mathbf{x}}_i + \tilde{\mathbf{x}}_i = \mathbf{P}\mathbf{P}'\mathbf{x}_i + \tilde{\mathbf{P}}\tilde{\mathbf{P}}'\mathbf{x}_i = \mathbf{C}\mathbf{x}_i + \tilde{\mathbf{C}}\mathbf{x}_i$, where $\mathbf{C} = \mathbf{P}\mathbf{P}'$ is the projection matrix into the principal component subspace (PCS), and $\tilde{\mathbf{C}} = \tilde{\mathbf{P}}\tilde{\mathbf{P}}'$ is the projection matrix into the residual subspace (RS).

The Q statistic, also commonly known as the Squared Prediction Error (SPE), measures the goodness of fit in the lower dimensional model (i.e., $l < p$) via the squared norm of the residual vector, $\tilde{\mathbf{x}}_i$, computed as

$$Q = ||\tilde{\mathbf{x}}_i||^2 = \mathbf{x}_i'\tilde{\mathbf{C}}\tilde{\mathbf{C}}\mathbf{x}_i = \mathbf{x}_i'\tilde{\mathbf{C}}\mathbf{x}_i = ||\tilde{\mathbf{C}}^{1/2}\mathbf{x}_i||^2.$$

The contribution of a variable to the Q statistic is called that variable's *contribution score*, with variables with the largest contribution scores presumed to be those most likely to be shifted. The contribution scores for the Q statistic are

$$c_j^Q = \left(\mathbf{d}_j'\tilde{\mathbf{C}}\mathbf{x}_i\right)^2,$$

where $\mathbf{d}_j$ is a vector of zeroes of length p except for a one in the jth row and is referred to as the direction vector.

Now, consider that variable j is shifted, resulting in a faulty measurement with $\mathbf{x}_i = \mathbf{x}_i^* + \mathbf{d}_j\delta$, where $\mathbf{x}_i^*$ is the fault-free part of the measurement, and $\mathbf{d}_j\delta$ is the faulty part. The faulty part is composed of the fault direction, $\mathbf{d}_j$, and δ is

https://t.me/learningnets

the scalar magnitude of the fault. Then, the contribution of variable k to the Q statistic when there is a fault in variable j for the ith observation is as follows:

$$\begin{aligned} c_k^Q &= \left(\mathbf{d}_k'\tilde{\mathbf{C}}\mathbf{x}_i\right)^2 = \left[\mathbf{d}_k'\tilde{\mathbf{C}}(\mathbf{x}_i^* + \mathbf{d}_j\delta)\right]^2 \\ &= \left[\mathbf{d}_k'\tilde{\mathbf{C}}\mathbf{x}_i^* + \mathbf{d}_k'\tilde{\mathbf{C}}\mathbf{d}_j\delta\right]^2 = \left[\mathbf{d}_k'\tilde{\mathbf{C}}\mathbf{x}_i^* + \tilde{c}_{kj}\delta\right]^2, \end{aligned}$$

where the first term in the brackets on the right-hand side of the equation depends on the observation itself and the contribution of variable k to the score, but the second term includes $\tilde{c}_{kj}$, the (k, j)th element of $\tilde{\mathbf{C}}$. The term $\tilde{c}_{kj}$ is not zero for $j \neq k$, and therefore, the jth variable's effect is *smeared* into the contribution of variable k.

For example, taking the covariance matrix from (3.1) and performing the eigenvalue-eigenvector decomposition results in

$$\tilde{\mathbf{C}} = \begin{bmatrix} 0.10 & -0.21 & 0.22 \\ -0.21 & 0.43 & -0.45 \\ 0.22 & -0.45 & 0.47 \end{bmatrix}.$$

Thus, if the contribution score for variable 3 is desired, but a fault is present in variable 1, then $\tilde{c}_{31} = 0.22$, which would inflate the contribution score for variable 3.

Some proposals exist that reduce the smearing effect, such as [13,14] for reconstruction-based contribution (RBC) scores for PCA and kernel PCA (KPCA), respectively. For example, the RBC of variable k to Q for PCA is defined as

$$c_k^{Q^{\text{RBC}}} = \frac{\left(\mathbf{d}_k'\tilde{\mathbf{C}}\mathbf{x}_i\right)^2}{\tilde{c}_{kk}} = \frac{c_k^Q}{\tilde{c}_{kk}},$$

where $\tilde{c}_{kk}$ is the kth diagonal element of $\tilde{\mathbf{C}}$. The RBC only differs from the contribution score by scaling it with $\tilde{c}_{kk}$ and still clearly exhibits the smearing effect. The question becomes whether the contribution scores will be largest for c_j^Q or $c_j^{Q^{\text{RBC}}}$ for a fault in variable j. Alcala and Qin [13] prove that $c_j^{Q^{\text{RBC}}}$ is the largest RBC for a fault in variable j, but the same does not hold for c_j^Q. Even in the case of single sensor faults, correct variable isolation by c_j^Q cannot be guaranteed. Furthermore, the authors of [13] show in simulation that the combined index statistic, ϕ, which is a weighted linear combination of T^2 and Q, yields more accurate fault isolation than either T^2 or Q alone with $c_j^{Q^{\text{RBC}}}$.

While we have defined $\mathbf{d}_j$ as a vector of zeros with a one in the jth entry, it could also be a matrix of zeroes with a one in each column, representing a multidimensional fault in which multiple variables are shifted simultaneously.

https://t.me/learningnets

Consequently, δ could be a vector in which each entry corresponds to the magnitude of the fault in the corresponding direction column. Thus, to compute the RBC, *a priori* information about the specific combination of potentially shifted variables is required. Simply testing a single fault direction at a time does not present this problem (such as a single sensor fault), but oftentimes complex faults will affect multiple variables simultaneously. Thus, one of the drawbacks of the RBC is that having knowledge of the fault directions is required to assess variable contribution. In spite of this drawback, RBC guarantees that the faulty variable for a single variable fault has the largest score and is a rapid approach for simple fault isolation. A paper by Yan and Yao [15] improves reconstruction-based contribution plots by incorporating variable selection.

3.2 Fault isolation

In this section, we will first discuss the premise of using variable selection as a tool for improving fault detection. Then, we review some of the traditional and historic approaches for fault isolation. More modern methods using various penalized regression techniques are covered in Sect. 3.2.3. Examples of both Phase I and Phase II methods will be presented.

3.2.1 Variable thinning

In some settings, what is referred to as variable isolation is primarily focused on *variable thinning*. Meaning, the greater the number of monitored variables, the more challenging it becomes to detect faults. Typically, only a small subset of the monitored variables actually shift, so including additional noisy features diminishes a method's ability to detect faults. Thus, to improve the power of fault detection, variable thinning is employed to remove those variables that are unlikely to have changed [2,6,16–24]. Many of these works do not just perform variable thinning but also isolate the variables affected by the fault, estimating the direction and magnitude of the shift in each variable. However, the variable isolation abilities of these methods are oftentimes not evaluated, instead focusing on the improvement in fault detection that results when reducing the number of monitored variables.

Similar to a simulation study in [2], we illustrate the effect that including additional unshifted variables has on detecting a fault. First, we simulate $n = 150$ observations of p variables from a multivariate normal distribution with mean zero and an identity variance–covariance matrix. A downward fault of 1 unit is introduced at observation number 101 and continues through the end of the sequence. The T^2 monitoring statistic is computed with the mean and variance–covariance matrix assumed known. We record (i) whether the fault is detected and (ii) the number of OC observations (out of 50) that are flagged. This process is repeated for 200 simulations as p is varied from 1 to 12. Results presented in Table 3.1 illustrate that the detection probability decreases rapidly from over

https://t.me/learningnets

0.90 to less than 0.30 as the number of variables increases. Furthermore, the strength of the detection, as measured by the average number of observations flagged across the simulated datasets, also decreases dramatically as p increases. This illustrates that weak faults become nearly impossible to detect, even with a very modest number of monitored variables.

TABLE 3.1 Detection probability and average number of observations flagged in 200 simulations as p varies. The first variable contains a downward shift of size one.

p	Detection Probability	Average Flags	p	Detection Probability	Average Flags
1	0.930	2.580	7	0.455	0.615
2	0.865	1.825	8	0.385	0.460
3	0.725	1.295	9	0.360	0.450
4	0.620	0.990	10	0.315	0.380
5	0.575	0.840	11	0.290	0.365
6	0.500	0.660	12	0.280	0.330

3.2.2 Iterative traditional isolation

Some of the first approaches to fault isolation were to apply a separate procedure after a control chart signaled a fault. These procedures require a decomposition of the T^2 statistic or similar step-down procedures. A few popular approaches are reviewed here [25]. Nearly all of these approaches first require that an OC signal has been issued by the fault detection method. In particular, the first two approaches presented assume that $\mathbf{x}_i \sim N_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, and the T^2 statistic is of course $T^2 = (\mathbf{x}_i - \bar{\mathbf{x}})'\mathbf{S}^{-1}(\mathbf{x}_i - \bar{\mathbf{x}})$. When $\mathbf{x}_i$ is independent of $\bar{\mathbf{x}}$ and $\mathbf{S}$, then the distribution of T^2 is

$$\frac{n(n-p)}{p(n+1)(n-1)}T^2 \sim F_{p,n-p},$$

and the upper control limit (UCL) uses the upper α quantile of the $F_{p,n-p}$ distribution.

3.2.2.1 Mason–Young–Tracy method

The Mason–Young–Tracy method (MYT) is a widely used approach that decomposes the T^2 statistic into its orthogonal components, which represent the contribution of each individual variable to the overall T^2 [26,27]. For p variables, T^2 can be rewritten as follows:

$$T^2 = T_1^2 + T_{2.1}^2 + T_{3.1,2}^2 + \cdots + T_{p.1,2,\ldots,p-1}^2. \tag{3.3}$$

https://t.me/learningnets

The first term in Eq. (3.3) is

$$T_1^2 = \left(\frac{x_{i,1} - \bar{x}_1}{s_1} \right)^2, \tag{3.4}$$

where $x_{i,1}$ is the first variable in the observation vector of $\mathbf{x}_i$, and $\bar{x}_1$ and s_1 are the mean and standard deviation of the first variable, respectively. Then, the additional terms can be written in general as

$$T^2_{j.1,\ldots,j-1} = \left(\frac{x_{i,j} - \bar{x}_{j.1,\ldots,j-1}}{s_{j.1,\ldots,j-1}} \right)^2, \tag{3.5}$$

where

$$\bar{x}_{j.1,\ldots,j-1} = \bar{x}_j + \mathbf{b}'_j \left(\mathbf{x}_i^{(j-1)} - \bar{\mathbf{x}}^{(j-1)} \right),$$

with $\bar{x}_j$ being the sample mean of n observations of the jth variable, and $\mathbf{b}_j$ a $(j-1)$-dimensional vector estimating the regression coefficients of the jth variable regressed on the prior $j-1$ variables. Furthermore, $\mathbf{x}_i^{(j-1)}$ is a $(j-1)$-dimensional vector that excludes the $j, \ldots, p$ variables, and $\bar{\mathbf{x}}^{(j-1)}$ is the sample mean of n observations of the first $j-1$ variables.

Because the order of the p variables is not unique, there exist $p!$ possible orderings of the variables in Eq. (3.3), each with its own distinct partition. Thus, as p gets large, the number of partitions to examine becomes prohibitively large. Mason et al. [26] do note that the terms of greatest interest tend to be the terms of the unadjusted contribution of a single variable and the term containing the adjusted contribution of one of the variables after adjusting for the other $p-1$ variables.

When an observation's T^2 value exceeds the UCL, the following steps can be followed to isolate the shifted variables, as described by [25]:

1. Compute the individual T_j^2 statistic for each variable, as in Eq. (3.4).
   - **a.** Classify variables whose T_j^2 exceed the UCL as shifted.
   - **b.** Remove shifted variables from the observation.
   - **c.** Recompute T^2 with $k \leq p$, and test it for significance. If it is still significant, proceed to the next step.
2. Compute all $T^2_{j.j'}$ terms for the remaining k variables, as in Eq. (3.5).
   - **a.** If $T^2_{j.j'}$ is high, then this indicates that the pairwise relationship between variables j and j' is unusual.
   - **b.** Remove the pairs of variables with a significant $T^2_{j.j'}$, leaving k' variables.
   - **c.** Recompute T^2 with $k' \leq k$, and test for significance. If it is still significant, proceed to the next step.
3. Repeat the prior step for higher-order terms, beginning with three-way terms, until either no variables are left in the reduced set or T^2 based on the reduced set of variables is no longer significant.

https://t.me/learningnets

Under the null hypothesis of no shift, the UCL for the partial terms, $T^2_{j.1,\ldots,j-1}$, is based on the distribution

$$T^2_{j.1,\ldots,j-1} \sim \frac{n+1}{n} F_{1,n-1}.$$

3.2.2.2 Murphy method

Murphy [28] proposed discriminant analysis to identify shifted variables. Once a fault has been detected with T^2, the following steps can be taken:

1. Calculate the individual charting statistic, $T^2_j(x_{i,j})$, for $j = 1, \ldots, j$,

$$T^2_j(x_{i,j}) = \frac{(x_{i,j} - \bar{x}_j)^2}{s^2_j}.$$

   a. Calculate the difference between the overall T^2 and each individual one, $D_1(j) = [T^2 - T^2_j(x_{i,j})]$.
   b. Choose the smallest difference, $\min_j D_1(j) = D_1(r)$. Then, the rth variable makes the greatest contribution to the overall T^2. If $D_1(r)$ is significant, then classify variable r as shifted and proceed to the next step.
2. Calculate the pairwise charting statistics, $T^2_2(x_{i,r}, x_{i,j})$ for all j variables except the rth one.
   a. Calculate the $p-1$ differences $D_2(r, j) = [T^2 - T^2_2(x_{i,r}, x_{i,j})]$ for all j variables except the rth one.
   b. Choose the smallest difference, $\min_j D_2(r, j) = D_2(r, s)$, and test if it is significant. If so, then also classify variable s as shifted.
3. Continue until either no variables' differences from T^2 are significant, or all of them are significant, in which case all variables would be classified as shifted.

The threshold for significance in each step depends upon the number of variables used in the reduced subset and whether $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are estimated or known. In the case that the parameters are known, the difference statistics $D_k(\cdot)$ are compared to a $\chi^2_{\alpha,k+1}$; otherwise, a scaled F-distribution is required.

3.2.2.3 Artificial neural network methods

Several proposals that incorporate Artificial Neural Networks (ANN) into variable isolation have been made [29–31]. Some, similar to the prior methods, require a fault detection method to detect a fault prior to initiating the fault isolation. In some cases, only those variables that have shifted are isolated, and in other cases, the goal is to both isolate the shifted variables and estimate the size and direction of the shifts. Psarakis [32] gives a short review of the use of ANN for fault detection in process monitoring and reviews several proposals for ANN for fault isolation.

As with all ANN, many configurations for building such models are possible, namely the size of the training data, the number of layers, the number of nodes

https://t.me/learningnets

within each layer, and the type of activation function. Thus, ANN require a large computational commitment to test many variations of models. Bersimis et al. [25] found that the performance of ANN in variable isolation was highly variable, depending on the strength and type of correlation present among the variables.

3.2.2.4 Discussion

There are some drawbacks of these traditional isolation techniques. For example, recomputing the monitoring statistics for all possible subsets of the monitored variables can become computationally infeasible when p is large. Furthermore, it is still difficult to associate the alarms triggered by MEWMA or MCUSUM charts with specific variables. Nevertheless, these methods still prove useful in applied problems [33]. Bersimis et al. [25] provide an overview of many of these methods and conduct an extensive comparison simulation study of them across several types of simulated data. However, this study still displays several shortcomings that should motivate additional research: (i) the study only simulates $p = 3$ or 5 variables; (ii) the only metric of interest reported is whether the method detected at least one of the shifted variables (which does not reveal if all shifted variables are isolated or if any unshifted variables are also flagged); and (iii) the overall percentages of identification of at least one shifted variable are still relatively low in most settings.

3.2.3 Variable selection methods

Some of the more recent works that propose methods to perform fault detection and isolation simultaneously frame the FD&I problem in terms of a multiple linear regression model and then use penalized regression variable selection methods [2,6,16]. To briefly review penalized regression, a multiple linear regression model takes the following form:

$$y_i = \mathbf{x}_i \boldsymbol{\beta} + \epsilon_i, \quad i = 1, \ldots, n,$$

where y_i, $\mathbf{x}_i$, $\boldsymbol{\beta}$, and ϵ_i are the response variable, vector of predictors, vector of coefficients, and iid random errors with distribution $N(0, \sigma^2)$, respectively. To perform variable selection and parameter estimation simultaneously, the following penalized least squares objective function is minimized with respect to $\boldsymbol{\beta}$:

$$\sum_{i=1}^{n} (y_i - \mathbf{x}_i \boldsymbol{\beta})^2 + \sum_{j=1}^{p} g_{\gamma_j} \left(|\beta_j| \right), \tag{3.6}$$

where β_j is the jth coefficient of $\boldsymbol{\beta}$, γ_j are regularization parameters, and $g_{\gamma_j}(\cdot)$ is the penalty function. For an L_q penalty function with $q \leq 1$, the variable selection is performed automatically by setting β_j equal to zero for those predictors with small estimated coefficients.

https://t.me/learningnets

Commonly chosen options for $g_{\gamma_j}(\cdot)$ include

- L_0 *penalty.* The L_0 penalty is $g_{\gamma_j}(\cdot) = \gamma \cdot I(|\beta_j| \neq 0)$, making the second term in Eq. (3.6) equal to $\gamma \sum_{j=1}^{p} I(\beta_j \neq 0)$. This penalty constrains the number of predictors that can enter the model to be less than a prespecified number, denoted s.
- L_1 *penalty.* The L_1 penalty is $g_{\gamma_j}(\cdot) = \gamma \cdot |\beta_j|$, making the second term in Eq. (3.6) equal to $\gamma \sum_{j=1}^{p} |\beta_j|$. This then corresponds to the least absolute shrinkage and selection operator (lasso) proposed by [34]. As $\gamma \to 0$, the estimator converges to the least squares estimator of $\boldsymbol{\beta}$.
- *Weighted* L_1 *penalty.* This penalty weights the coefficients by an initial estimate of the coefficients, as follows:

$$g_{\gamma_j}(\cdot) = \gamma \frac{|\beta_j|}{|\tilde{\beta}_j|^{\alpha}},$$

where $\tilde{\beta}_j$ is the initial estimate of β_j, which can be the least squares estimate, and $\alpha > 0$ is a prespecified constant. It is often recommended to set $\alpha = 1$ [35, 36]. This weighted penalty corresponds to what is called *adaptive lasso*, and the weight allows for different shrinkage to be applied to different predictors, forcing those with already small coefficients to be driven to zero even faster. Adaptive lasso produces asymptotically unbiased coefficient estimates and has oracle properties, meaning it identifies the correct subset of variables and has an optimal estimation rate [35,37].

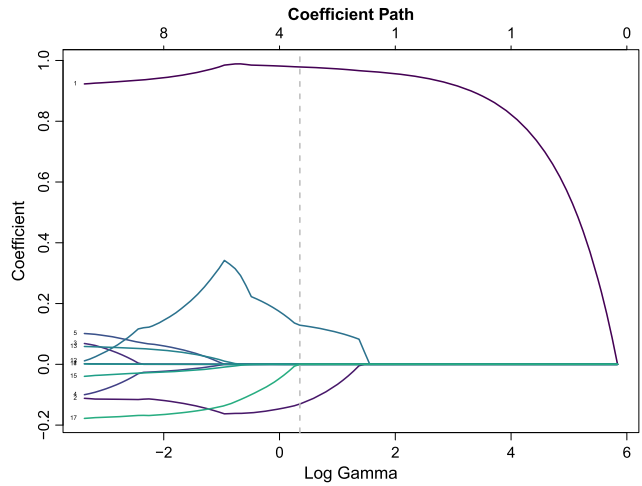


FIGURE 3.4 Values of the coefficients in a linear model as γ increases.

An excellent overview of lasso and its extensions is given in [38], and a coefficient path as γ changes is shown in Fig. 3.4 to illustrate the impact of γ on

https://t.me/learningnets

the coefficients. Choosing γ can be done through cross-validation. In Fig. 3.4, the dashed vertical line shows γ_{se}, which is the largest value of γ such that the squared error is within one standard error of the minimum. At this point, three predictors remain in the model out of the initial 17 that were included in the fit.

To employ variable selection techniques for fault isolation, the multiple linear regression model is reframed so that the mean vector is being monitored and those variables whose mean is unchanged are filtered out. Several variations of the penalties presented here have also been employed, and a few specific methods will be presented in the following subsections.

3.2.3.1 Phase I variable selection

One of the primary goals of a Phase I analysis is to test whether a set of observations could reasonably be expected to come from an IC process. As such, data used in a Phase I analysis could be contaminated by multiple faults simultaneously. A secondary goal may be to retrospectively analyze a fault or multiple faults that occurred so that operators may begin to diagnosis them. We present an example of each type of analysis in this section.

Example method 1, mphase1. Capizzi and Masarotto [8] developed a distribution-free retrospective change point detection method designed to test for multiple change-points over a given period of time, called mphase1. Their approach is designed to first test whether the period of time under consideration is IC; if not, then the shifted variables can be identified, along with their shift sizes and directions. Their method assumes that the observations are iid when the process is IC. It also allows for subgrouped data, in which multiple realizations of the process are present at a single time points. This allows the user to detect outlying observations at a single isolated time point in addition to detecting step shifts. For simplicity, we ignore the potential presence of subgrouped observations in our description of the method.

A user-friendly `R` package called `dfphase1` contains code to implement mphase1 [8]. It is not assumed that data from an IC period exists. The following steps are performed:

1. *Standardization.* The data are standardized and transformed to the multivariate signed ranks using a robust estimate of the center and spread. For the location estimate, $\boldsymbol{\ell}$, the transformation–retransformation spatial median is applied to subgroup means [39]. For the spread, a robust scatter matrix, $\mathbf{S}_r$, is constructed as

$$\mathbf{S}_r = \frac{1}{2(n-1)} \sum_{i=1}^{n} (\mathbf{x}_i - \mathbf{x}_{i-1}) (\mathbf{x}_i - \mathbf{x}_{i-1})'.$$

Then, they use $\boldsymbol{\ell}$ and $\mathbf{S}_r$ to standardize the data, $\mathbf{z}_i = \mathbf{S}_r^{-1/2}(\mathbf{x}_i - \boldsymbol{\ell})$, and compute the multivariate signed ranks $\mathbf{u}_i$ as follows:

https://t.me/learningnets

$$\mathbf{u}_i = \begin{cases} \mathbf{0}, & \text{if } \mathbf{z}_i = \mathbf{0}, \\ F^{-1}_{\chi^2_p}\left(\frac{g_i}{1+n}\right)\frac{\mathbf{z}_i}{\|\mathbf{z}_i\|}, & \text{if } \mathbf{z}_i \neq \mathbf{0}, \end{cases}$$

where $\|\mathbf{v}\| = \sqrt{\mathbf{v}'\mathbf{v}}$ is the Euclidean norm of $\mathbf{v}$; g_i is the rank of $\{\|\mathbf{z}_1\|, \ldots, \|\mathbf{z}_n\|\}$; and $F^{-1}_{\chi^2_p}(\cdot)$ is the quantile function of a χ^2 random variable with p degrees of freedom. A version for subgrouped data is given in [8,40].

2. *Screening.* Next, an initial screening process using forward selection identifies K locations where a fault is most likely to have occurred, where K is either selected by the user or is chosen to be the minimum of 50 and the integer closest to $\sqrt{n}$. For the kth iteration, $k = 1, 2, \ldots, K$, of the forward selection process, an additional shift at time step $i = 2, \ldots, n-1$ is added to the model. This time step is chosen as that with the minimum residual sum of squares conditional on the previously identified $k-1$ shifts, namely $\sum_{i=1}^{n} ||\mathbf{u}_i - \hat{\mathbf{u}}_i^{(k)}||^2$, where $\hat{\mathbf{u}}_i^{(k)}$ comes from a model fitted with a potentially different mean vector at each time point. Shifts within $l_{\min}$ time steps of the $k-1$ shifts already present are not considered. After each step, the explained variance is computed as

$$T_k = \sum_{i=1}^{n} ||\hat{\mathbf{u}}_i^{(k)}||^2 - n||\bar{\mathbf{u}}||^2,$$

where $\bar{\mathbf{u}}$ is the overall mean of the signed ranks.

3. *Testing.* Then, $T_k, k = 1, \ldots, K$, test statistics are aggregated to form a single test statistic as follows:

$$W_{\text{OBS}} = \max_{k=1,\ldots,K} \frac{T_k - \text{E}_0(T_k)}{\sqrt{\text{Var}_0(T_k)}}.$$

The expected value and variance of T_k are estimated with the sample mean and variance of T_k, respectively. A permutation-based p-value is computed by permuting the iid observations to test the null hypothesis that the process does not display any step or isolated shifts and is stable in its location.

4. *Post-signal diagnostic.* If the null hypothesis is rejected, fault isolation is performed via adaptive lasso to further prune the estimated coefficients from the forward selection screening process, which are used as initial estimates [35]. Adaptive lasso drives some of the small shifts to zero. The particular model fit is as follows: $\mathbf{u}_i = \mathbf{S}_r^{-1/2}\boldsymbol{\delta}_0 + \mathbf{S}_r^{-1/2}\boldsymbol{\delta}_1\boldsymbol{\xi}_i^{(1)} + \cdots + \mathbf{S}_r^{-1/2}\boldsymbol{\delta}_K\boldsymbol{\xi}_i^{(K)} + \epsilon_i$. Here, $\boldsymbol{\delta}_0$ is the intercept while $\boldsymbol{\delta}_1, \ldots, \boldsymbol{\delta}_K$ are the directions of the potential location shifts to be estimated, and $\boldsymbol{\xi}_i^{(k)} = I(i \geq \tau^{(k)}), k = 1, \ldots, K$, corresponds to a step shift at time $\tau^{(k)}$. Then, the penalized objective function to minimize is

$$\sum_{i=1}^{n} ||\mathbf{u}_i - \mathbf{S}_r^{-1/2}\boldsymbol{\delta}_0 - \sum_{k=1}^{K} \mathbf{S}_r^{-1/2}\boldsymbol{\delta}_k\boldsymbol{\xi}_i^{(k)}||^2 + \gamma \sum_{k=1}^{K}\sum_{h=1}^{g} \left|\frac{\delta_{k,h}}{\hat{\delta}^{ls}_{k,h}}\right|, \qquad (3.7)$$

https://t.me/learningnets

where $\delta_{k,h}$ is the hth element of $\boldsymbol{\delta}_k$, and $\hat{\delta}^{ls}_{k,h}$ is its least squares estimate. The regularization parameter is selected using the EBIC criterion proposed by [41],

$$\text{EBIC}_{\phi}(\gamma) = np \log\left(\frac{\text{RSS}(\gamma)}{np}\right) + \nu(\gamma)\log(np) + 2\phi \log\binom{np-2}{\nu(\gamma)}, \quad (3.8)$$

where $\text{RSS}(\gamma)$ is the residual sum of squares of the model for a given γ; $\nu(\gamma)$ is the number of change points detected; $np-2$ is the dimension of the searched parameter space, or the number of potential change point locations; and $\phi \in [0, 1]$ is a user-defined parameter that controls the false signal rate. Setting $\phi = 1$ controls the number of false signals to be low.

Capizzi and Masarotto [8] study the performance of mphase1 extensively in simulation and find that it performs very well for a variety of underlying multivariate distributions and for a variety of fault types. It achieves the nominal false alarm probability in IC data regardless of distribution, and for data contaminated with multiple faults, it identifies all of the faults exactly or approximately (within five time steps). While the authors do not provide a performance assessment of the method's ability to correctly identify the shifted variables, in practice, the method appears to properly isolate the correct variables when data meet the method's assumptions.

To illustrate mphase1, we simulate 150 observations of ten variables, both from a multivariate normal with an identity variance–covariance matrix. In the first case, data are IC, and Fig. 3.5 shows the p-value of 0.359 for testing the null hypothesis that the mean of the observations is constant. Thus, there is no significant evidence that the mean differs from zero, and this dataset could then be used to estimate parameters and rejection thresholds for a subsequent Phase II analysis. In a second case, the data are OC with shifts in the mean as shown in Fig. 3.6. Variables $\mathbf{x}_1$, $\mathbf{x}_2$, and $\mathbf{x}_3$ exhibit step shifts, some of which coincide. Variable $\mathbf{x}_{10}$ has a drift fault beginning at observation 100. Fig. 3.7 shows the results. The p-value to test that the mean has shifted for any of the variables is less than 0.001, indicating significant evidence of at least one shift. The shifts in $\mathbf{x}_1$, $\mathbf{x}_2$, and $\mathbf{x}_3$ are estimated nearly perfectly with the shifts sometimes detected one or two time steps earlier or later than when they actually occur. The drift fault in variable $\mathbf{x}_{10}$ is treated as if multiple step shifts occur, so if a drift is present, it is often manifested with multiple consecutive shifts in the same direction.

While the mphase1 method provides an approach to directly test if a set of observations can be used for estimating parameters for Phase II analysis, another goal may be to isolate variables to diagnose faults that occurred in the past. Klanderman et al. [42] develop a retrospective change-point detection method that is also distribution-free, but relies on fused lasso to drive the difference in adjacent means to zero. Their method is called adjusted flexible fault isolation (aFFI).

Example method 2, aFFI. Most proposed methods assume that the mean of the process is stationary and constant, but it is often the case in practice that

https://t.me/learningnets

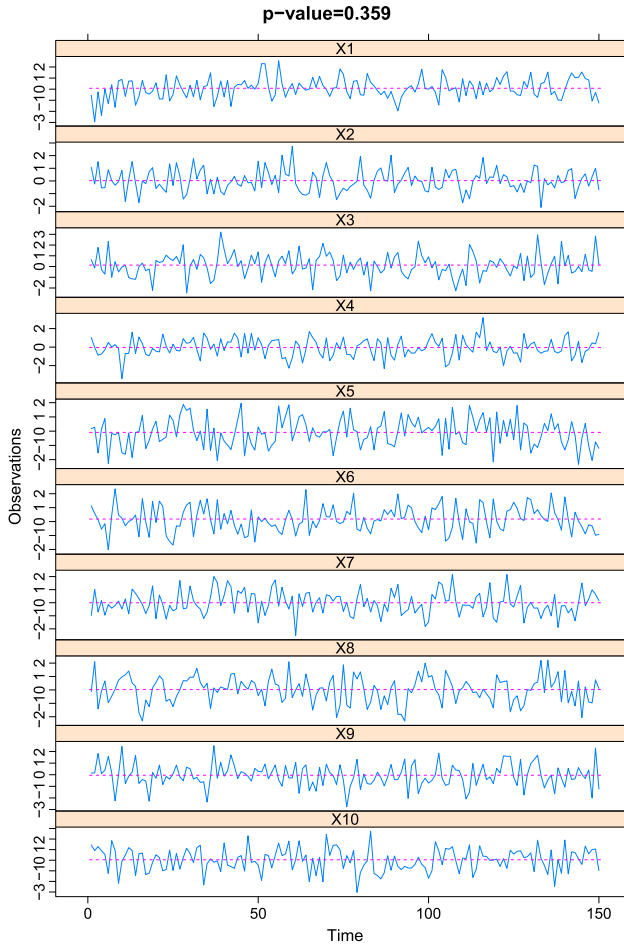


FIGURE 3.5 Plot of ten variables simulated under IC conditions. The p-value at the top of the figure is for the mphase1 test of the null hypothesis that the observations are location stationary.

the mean is nonzero. A common approach is to remove the trend by fitting a model to data that are known to be IC, with several examples provided in dynamic screening systems [43–46]. In Sect. 3.5.1, an example of detrending in a complex system is given followed by application of the method in [42], termed *adjusted Flexible Fault Isolation* (aFFI), which is described here. This method does not assume that the observations are stationary, and in addition, temporal dependence in the observations is also allowed. The detrending occurs for a training period, and the estimates of the parameters in the trend model are used to detrend a period for which monitoring is desired. Assuming that the observations, $\mathbf{x}_i$, $i = 1, \ldots, p$, have been suitably standardized and detrended, the set

https://t.me/learningnets

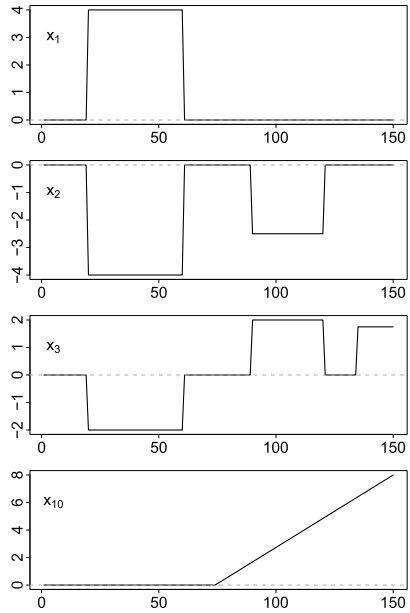


FIGURE 3.6 Plot of the mean of the four variables exhibiting OC conditions. The first three variables have transient or sustained shift faults while the last has a drift fault beginning at observation 100.

of p linear regression models are fit as

$$\mathbf{x}_j^{\mathrm{M}} = \boldsymbol{\beta}_j + \boldsymbol{\epsilon}_j, \quad j = 1, \ldots, p, \tag{3.9}$$

where $\mathbf{x}_j^{\mathrm{M}}$, $\boldsymbol{\beta}_j$, and $\boldsymbol{\epsilon}_j$ are vectors of length n, and the errors $\boldsymbol{\epsilon}_j$ have mean zero and constant variance. The $\mathbf{x}_j^{\mathrm{M}}$ is the set of observations for variable j in the monitoring period. The mean of the response for the jth variable at time step t is $\beta_{j,t}$, and a change in the mean of the process is indicated when $\beta_{j,t} \neq \beta_{j,t+1}$. When $\beta_{j,t} \neq \beta_{j,t+1}$, time $t+1$ is referred to as a *change point* for variable j. A variable is classified as *shifted* at time $t+1$ if there is a change point at time $t+1$ versus *unshifted* if no change occurs at $t+1$. The $\boldsymbol{\beta}_j$ for $j = 1, \ldots, p$, are estimated with fused lasso.

Fused lasso is used when there is a natural ordering to the parameters, such as parameters that are indexed by time [47]. Fused lasso penalizes the difference between adjacent coefficients rather than penalizing the coefficients themselves, driving the small changes between adjacent coefficients to zero [38]. Zhang et al. [48] also use fused lasso in a Bayesian framework to perform fault isolation by estimating the fault probability for each variable. The theoretical properties of change point detection using fused lasso are in [49].

An algorithm to perform fused lasso [50] is implemented in the `R` package `genlasso` [51]. An estimate for $\boldsymbol{\beta}_j$ for $j = 1, \ldots, p$ is obtained by minimizing

https://t.me/learningnets

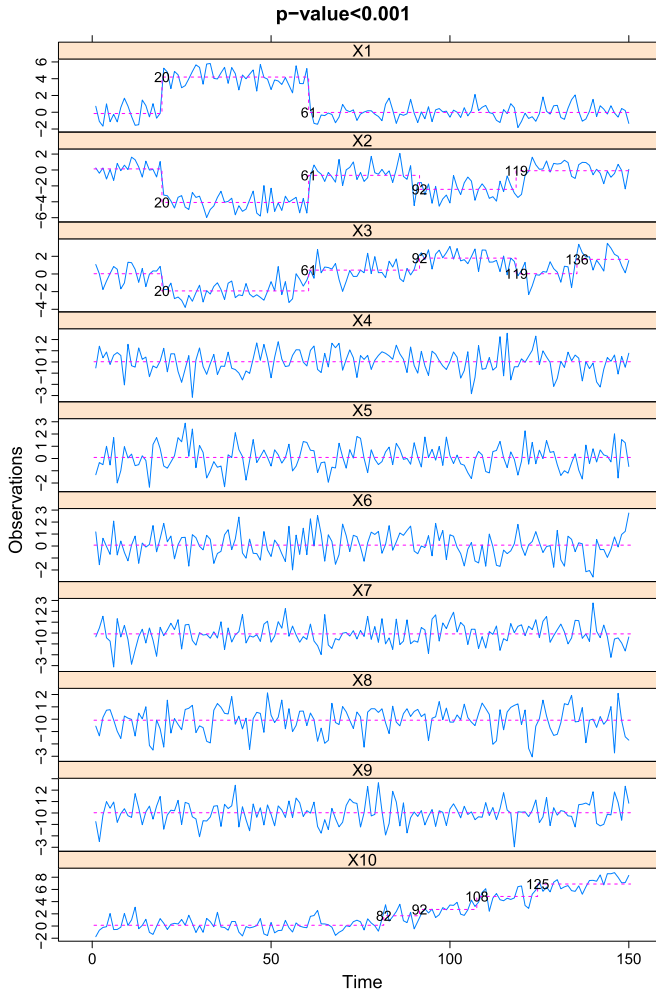


FIGURE 3.7 Plot of ten variables simulated under OC conditions. The p-value at the top of the figure is for the mphase1 test of the null hypothesis that the observations are location stationary.

the following:

$$\frac{1}{2}\sum_{j=1}^{p}\sum_{t=1}^{n}\left(x_{j,t}-\beta_{j,t}\right)^2+\gamma\sum_{j=1}^{p}\sum_{t=1}^{n-1}|\beta_{j,t+1}-\beta_{j,t}|.$$

In order to choose γ, the EBIC in Eq. (3.8) can be used again. Within this framework, the EBIC is

$$\text{EBIC}_{\phi}(\gamma)=np\log\left(\frac{\text{RSS}(\gamma)}{np}\right)+\nu(\gamma)\log(np)+2\phi\log\binom{np-p}{\nu(\gamma)}, \quad (3.10)$$

https://t.me/learningnets

where the primary differences are that RSS(λ) is the residual sum of squares of Eq. (3.9), and $np - p$ is the dimension of the searched parameter space, or the number of potential change point locations. There cannot be a change point detected at the first observation, $t = 1$, so change points can only be detected at a maximum of $np - p$ locations.

One problem with the EBIC in Eq. (3.10) is that using the sample size, n, in it implies that the observations are independent. However, if $\mathbf{x}_j$ are autocorrelated, then the effective sample size (ESS), or the equivalent number of independent observations, differs from n. If positive autocorrelation is present, the ESS will be smaller than the actual sample size, and the stronger the positive autocorrelation, the smaller the ESS. Therefore, the ESS is calculated, which is denoted n', as follows:

$$n' = \frac{n}{1 + \frac{1}{n}\sum_{h>0} R(h)}, \tag{3.11}$$

where $R(h)$ is a user-selected function that models the autocorrelation when the process is IC between two distinct observations (i.e., $h \neq 0$) that are h time steps apart [52]. Confirming the presence of autocorrelation and determining the most appropriate $R(h)$ can be done by investigating the sample autocorrelation function (ACF) and partial autocorrelation function (PACF) plots of each $\mathbf{x}_j$ in the IC training period. This assumes that the strength and structure of the temporal dependence are the same for each variable. Replacing n with n' leads to the adjusted EBIC (aEBIC)

$$\text{aEBIC}_\phi(\gamma) = n' p \log\left(\frac{\text{RSS}(\gamma)}{n' p}\right) + \nu(\gamma)\log(n' p) + 2\phi \log\binom{np - p}{\nu(\gamma)}.$$

Then, γ is chosen such that aEBIC is minimized, yielding the estimated coefficients $\hat{\boldsymbol{\beta}}_j$ for $j = 1, \ldots, p$. Clearly, when the standardized residuals are independent, $R(h) = 0$ for $h > 0$, then $n' = n$, so aEBIC can be used to select γ even in the absence of autocorrelation.

Finally, the estimated coefficients must be converted into a list of times at which faults may have occurred and a list of shifted variables associated with each fault. A change point at time $t + 1$ is identified in variable j when the estimated $\beta_{j,t} \neq \beta_{j,t+1}$. To reduce the number of faults detected within a few time steps of each other, change points are ordered from first to last and, if any additional change points are identified within κ time steps of the previous change point, they are combined into a single fault. The parameter κ is set to be relatively small so that shifts in multiple variables detected within a small window in time are associated with the same fault, but two distinct faults are not combined into a single fault.

An estimate of the shift size and direction in the standardized residuals, $\delta_{j,t}$, for the jth variable at time t is

$$\hat{\delta}_{j,t} = \max_{t_1, t_2 \in \{t, t+1, \ldots, t+\kappa\}} \left|\hat{\beta}_{j,t_1} - \hat{\beta}_{j,t_2}\right| \cdot \text{sign(max difference)}.$$

https://t.me/learningnets

The size of the shift in the standardized residuals is not the same as the size of the shift in the original data. Because of the nonstationarity present in the original data and the difference in scale of the features monitored, the shift sizes should be compared in the standardized residuals rather than the original data. The shift sizes may be used by the researcher to assess the size of the contribution of each of the shifted variables to the fault and to identify which variables to prioritize in the fault diagnosis step.

In simulation, aFFI is able to accurately detect different types of faults (shift, drift, or latent fault), and it also accurately isolates the shifted variables under a variety of conditions (i.e., varying number of variables, nonstationarity, and temporal dependence). Unlike some methods, there is also no *a priori* estimate of the number or structure of the faults required. Such a method could be used to catalogue and characterize faults from a historical data archive. An example of this approach applied to a real problem is given in Sect. 3.5.1.

3.2.3.2 Phase II variable selection

Many more options for Phase II fault isolation using variable selection methods are possible. Here, we will describe three important developments that are more powerful than the decomposition approaches described in Sect. 3.2.2, particularly when the number of variables is large. We give references for additional works for more specialized cases.

Example method 1, VS-MSPC. In [2], the monitored observation, $\mathbf{x}_i$, is assumed to follow a p-dimensional $N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ distribution. A test of the hypotheses

$$\begin{cases} H_0 : \boldsymbol{\mu} \in \boldsymbol{\Omega}_0, \\ H_1 : \boldsymbol{\mu} \in \boldsymbol{\Omega}_1 \end{cases} \tag{3.12}$$

tests whether the process is IC for $\boldsymbol{\Omega}_0 = (0, 0, \ldots, 0)'$ versus OC for $\boldsymbol{\Omega}_1 = \{\boldsymbol{\mu} : \boldsymbol{\mu} = \delta\mathbf{d}, \delta > 0\}$. As before, $\mathbf{d}$ is a direction vector such that $\sqrt{\mathbf{d}'\boldsymbol{\Sigma}^{-1}\mathbf{d}} = 1$, and δ is a scalar indicating the size of the shift. A generalized likelihood ratio test statistic is formed by writing out the likelihood under both hypotheses as

$$\lambda(\mathbf{x}_i) = \frac{\max_{\boldsymbol{\mu}\in\boldsymbol{\Omega}_0} L(\mathbf{x}_i, \boldsymbol{\mu})}{\max_{\boldsymbol{\mu}\in\boldsymbol{\Omega}_1} L(\mathbf{x}_i, \boldsymbol{\mu})},$$

where $L(\mathbf{x}_i, \boldsymbol{\mu})$ is the likelihood of $\mathbf{x}_i$. Substituting $\boldsymbol{\Omega}_0 = (0, 0, \ldots, 0)'$ in the numerator, taking the log, and using the assumption of normality for the likelihoods, the rejection region of the test can be written as

$$\Lambda(\mathbf{x_i}) = \min_{\boldsymbol{\mu}\in\Omega_1} \left(\mathbf{x}_i'\boldsymbol{\Sigma}^{-1}\mathbf{x}_i + (\mathbf{x}_i - \boldsymbol{\mu})'\boldsymbol{\Sigma}(\mathbf{x}_i - \boldsymbol{\mu}) \right) < \log c,$$

for c some constant. However, the first term does not depend on $\boldsymbol{\mu}$, so the focus rests on the second term, as follows:

$$S^2 = \min_{\boldsymbol{\mu}\in\Omega_1} \left((\mathbf{x}_i - \boldsymbol{\mu})'\boldsymbol{\Sigma}(\mathbf{x}_i - \boldsymbol{\mu}) \right), \tag{3.13}$$

https://t.me/learningnets

with the solution of Eq. (3.13), denoted $\boldsymbol{\mu}^*$, being an estimator of $\boldsymbol{\mu}$. Assuming that most of the elements of $\boldsymbol{\mu}$ are zero, penalization can be used to force small coefficients to zero.

The penalized version of Eq. (3.13) used by [2] uses the L_0 penalty in order to constrain the absolute number of nonzero coefficients. Then, decomposing the positive-definite $\boldsymbol{\Sigma}$ using the Cholesky decomposition as $\boldsymbol{\Sigma} = \mathbf{L}\mathbf{L}'$, Eq. (3.13) with penalization can be rewritten as

$$\begin{cases} \min_{\boldsymbol{\mu}\in\boldsymbol{\Omega}_1} \left((\mathbf{z}_i - \mathbf{L}\boldsymbol{\mu})'(\mathbf{z}_i - \mathbf{L}\boldsymbol{\mu})\right) \\ \text{s.t.} \sum_{j=1}^{p} I\left(|\mu_j| \neq 0\right) \leq s, \end{cases}$$

where $\mathbf{z}_i = \mathbf{L}\mathbf{x}_i$. Wang and Jiang [2] advocate the use of forward selection, starting with no predictors and adding the predictor that produces the largest decrease in the sum of squared errors. They use the "F-to-enter" rule whereby the following F-value is calculated for every predictor, and the one with the highest value enters the model,

$$F = \frac{(R_{k+1}^2 - R_k^2)(n-k-1)}{1 - R_{k+1}^2},$$

where R_k^2 is the R^2 of the model with k predictors. No formal hypothesis test is performed, and once s variables have entered the model, no additional variables are added. When no prior knowledge of the number of shifted variables exist, the authors of [2] suggest the use of penalized or sparse PCA to find s, which takes linear combinations of only a subset of variables [9,53,54].

The control charting scheme proceeds as follows:

1. As each new observation is collected, variable selection is performed, as described above, finding the solution, $\boldsymbol{\mu}^*$, to Eq. (3.13).
2. Variables with nonzero coefficients identified in step 1 are charted using

$$\Lambda(\mathbf{x}_i) = 2\mathbf{x}_i'\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}^* - \boldsymbol{\mu}^*\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}^*.$$

   Values that exceed the preset control limit are deemed OC.
3. After an OC alarm is issued, those variables with nonzero coefficients identified in the first step are classified as shifted.

Wang and Jiang [2] test the Average Run Length (ARL) performance of VS-MSPC in both IC and OC settings with variable dimension ranging from $p = 10$ to $p = 100$ and find that it performs better than Hotelling's T^2 for detecting medium and large shifts. However, even if s is misspecified, the control chart still performs well. The performance of the method in detecting the shifted variables is not evaluated, but the method works well for a particular example.

Example method 2, LEWMA. Zou and Qiu [16] propose an exponentially weighted moving average (EWMA) based charting statistic that gives less weight to older observations and uses the adaptive lasso penalty. Named

https://t.me/learningnets

LEWMA (Lasso EWMA), this approach tests the same overall hypotheses as the prior method, but they initially formulate a test statistic that is computed across many values of the tuning parameter, γ. The penalized likelihood function is written as

$$L(\boldsymbol{\mu}) = n(\bar{\mathbf{x}} - \boldsymbol{\mu})'\boldsymbol{\Sigma}^{-1}(\bar{\mathbf{x}} - \boldsymbol{\mu}) + n\gamma \sum_{j=1}^{p} \frac{1}{|\bar{x}_j|^a}|\mu_j|,$$

in which each variable is shrunk by its respective sample mean, and, as is often recommended, $a = 1$. The estimator that minimizes the likelihood is denoted $\boldsymbol{\mu}_\gamma^*$, and a lasso-based test statistic for the hypotheses in Eq. (3.12) is

$$T_\gamma = \frac{n((\boldsymbol{\mu}_\gamma^*)'\boldsymbol{\Sigma}^{-1}\bar{\mathbf{x}})^2}{(\boldsymbol{\mu}_\gamma^*)'\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_\gamma^*}.$$

The selection of the correct subset of variables depends on the correct selection of γ, which can be chosen in a variety of ways, such as cross-validation or BIC. However, the best γ for estimation does not always produce a powerful hypothesis test; therefore, Zou and Qiu [16] borrow from the nonparametric testing framework and create a single test statistic based on several values of γ, as follows:

$$T = \max_{j=1,\ldots,q} \frac{T_{\gamma_j} - \mathrm{E}(T_{\gamma_j})}{\sqrt{\mathrm{Var}(T_{\gamma_j})}},$$

where $\mathrm{E}(T_{\gamma_j})$ and $\mathrm{Var}(T_{\gamma_j})$ are the sample mean and variance of T_{γ_j} under the null hypothesis. The authors of [16] recommend setting $q = r + 1$ or $q = r + 2$ if prior knowledge indicates that shifts occur in at most r variables. When such knowledge is unavailable, they show that $q = p$ also performs well in practice.

To construct the LEWMA, the multivariate EWMA sequence is defined as

$$\mathbf{u}_i = \lambda\mathbf{x}_i + (1 - \lambda)\mathbf{u}_{i-1}, \quad i = 1, 2, \ldots, \tag{3.14}$$

where $\mathbf{u}_0 = 0$ and $\lambda \in (0, 1]$ is a weight. Then, the EWMA is combined with the lasso, and q lasso estimators of the following penalized likelihood function for various values of γ are computed as

$$(\mathbf{u}_i - \boldsymbol{\mu})'\boldsymbol{\Sigma}^{-1}(\mathbf{u}_i - \boldsymbol{\mu}) + \gamma \sum_{j=1}^{p} \frac{1}{|\mathbf{u}_j|}|\mu_j|.$$

Then, using

$$W_{j,\gamma} = \frac{2 - \lambda}{\lambda[1 - (1 - \lambda)^{2j}]} \frac{(\mathbf{u}_j'\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_\gamma^*)^2}{(\boldsymbol{\mu}_\gamma^*)'\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_\gamma^*},$$

https://t.me/learningnets

the control chart signals a shift when

$$Q_j = \max_{k=1,\ldots,q} \frac{W_{j,k} - \mathrm{E}(W_{j,\gamma})}{\sqrt{\mathrm{Var}(W_{j,\gamma})}} > L,$$

where L is a control limit chosen to give a particular IC ARL. This L can be determined through simulation and must be determined by a one-time computation prior to performing Phase II monitoring. In practice, the $\frac{2-\lambda}{\lambda[1-(1-\lambda)^{2j}]}$ factor in $W_{j,\gamma}$ can be replaced with the asymptotic form $(2-\lambda)/\lambda$.

LEWMA is evaluated through simulation, particularly with respect to its ARL in OC settings, for $p = 5$ and $p = 15$. It is compared to two other EWMA charting statistics across multiple shift sizes in varying numbers of variables. The results show that LEWMA provides well-rounded protection against a variety of shifts in terms of ARL. Furthermore, the authors assess the ability of the method to correctly identify all shifted variables in a set of 15 variables. For a single shifted variable out of the 15, LEWMA correctly identifies the shifted variable in approximately 55 to 60% of simulations; however, as the number of shifted variables increases from 1 to 2 to 3, the highest proportion of simulations in which the correct shifted variables are identified drops from 60% to 37%, to 24%. Conversely, the proportion of simulations in which an error is made in identifying shifted variables (either a shifted variable is not identified or an unshifted variable is selected) increase as the number of shifted variables in the simulation increase from 6% for 1 shifted variable to 22% for 2 shifted variables, to 34% for three shifted variables. Nevertheless, LEWMA still performs better than its competitors. This study is one of the first to assess the performance of fault isolation in addition to fault detection, and similarly to [25], the results indicate that fault isolation is a very challenging problem, and methods have room for improvement.

Example method 3, APC-PCSR. Adaptive PC and PC-based Signal Recovery (APC-PCSR) adaptively chooses principal components (PCs), allowing them to vary over time, and then incorporates a diagnostic approach that exploits adaptive lasso [55]. This method tends to works better in high-dimensional settings than [2,6,16,17], which are computationally burdensome as the dimension grows. It can be used when variables are dependent, and knowledge of the fault direction, size, and/or number of affected variables are not needed.

To reduce the dimension of the data, PCA is performed. Here, $\mathbf{A}$ is the $p \times p$ matrix of eigenvectors of $\mathbf{\Sigma}$, and $\mathbf{y}_i = \mathbf{A}\mathbf{x}_i$ denotes the PC scores. The eigenvectors in $\mathbf{A}$ are ordered such that the corresponding eigenvalues are in decreasing order with $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p$. Monitoring just the top l PCs (usually those with the largest eigenvalues) can sometimes miss the fault because the variance of the top PCs is larger, so the bottom PCs can be more sensitive to the fault. To account for this, the standardized PC scores are computed as

$$\tilde{y}_{i,j} = \frac{y_{i,j}}{\sqrt{\lambda_j}}, \quad j = 1, \ldots, p.$$

https://t.me/learningnets

The IC observations are assumed to be distributed as $\mathbf{x}_i \sim N(\mathbf{0}, \mathbf{\Sigma})$, but under OC conditions, the distribution becomes $\mathbf{x}_i \sim N(\boldsymbol{\mu}, \mathbf{\Sigma})$. Thus, the OC process' PC scores will have distribution $\mathbf{y}_i \sim N(\mathbf{A}'\boldsymbol{\mu}, \mathbf{\Lambda})$, where $\mathbf{\Lambda} = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_p)$. Then, monitoring is conducted using an EWMA statistic, similar to the one presented in Eq. (3.14) but modified to apply to the standardized PC scores as follows:

$$u_{i,j} = \lambda \tilde{y}_{i,j} + (1-\lambda)u_{i-1}, \quad i = 1, 2, \ldots, \quad j = 1, 2, \ldots, p,$$

with $u_0 = 0$, and $\lambda \in (0, 1]$, as before. Note, a λ with no subscript denotes the weight in the EWMA statistic, but with a subscript, it denotes an eigenvalue. Then, an IC process will have $u_{i,j} \sim N\left(0, \sigma^2 = \frac{\lambda}{1-\lambda}\right)$, and its squared standardized value is

$$d_{i,j} = \left(\frac{u_{i,j}}{\sqrt{\frac{\lambda}{1-\lambda}}} \right)^2 \sim \chi^2_{(1)}.$$

In an OC process with a mean shift, some of the $d_{i,j}$ will become large, but Ebrahimi et al. [55] recommend filtering out some of the PCs. They simply threshold the $d_{i,j}$ using the α upper-tail quantile from the $\chi^2_{(1)}$ distribution and monitor the new statistic,

$$R_i = \sum_{j=1}^{p} \left(d_{i,j} - \chi^2_{(1),\alpha} \right)_+,$$

where $(\cdot)_+ = \max(0, \cdot)$. When R_i exceeds some threshold, then an alarm is issued, and this threshold is obtained through simulation under IC conditions to achieve a prespecified ARL.

Once a fault is detected, the second step is to integrate the PCs into the fault isolation. Assuming that each observation is measured with noise, the OC state can be represented as $\mathbf{x}_i = \boldsymbol{\mu} + \boldsymbol{\epsilon}_i$ with $\boldsymbol{\epsilon} \sim (\mathbf{0}, \mathbf{\Sigma})$. Then, the OC PC scores are $\mathbf{y}_i = \mathbf{A}\mathbf{x}_i = \mathbf{A}\boldsymbol{\mu} + \tilde{\boldsymbol{\epsilon}}$, with $\tilde{\boldsymbol{\epsilon}} = \mathbf{A}\boldsymbol{\epsilon}$. The goal is to estimate a sparse $\boldsymbol{\mu}$, so this is done with adaptive lasso

$$\boldsymbol{\mu}^* = \text{argmin}_{\boldsymbol{\mu}} \left((\mathbf{y}_i - \mathbf{A}\boldsymbol{\mu})'(\mathbf{y}_i - \mathbf{A}\boldsymbol{\mu}) + \gamma \sum_{j=1}^{p} \frac{1}{\bar{x}_j} |\mu_j| \right),$$

where $\bar{x}_j$ the least squares mean of the jth variable. However, the covariance of each variable in $\mathbf{y}_i$ varies, so the standardized PC scores and PCs are used, written in matrix form, as follows:

$$\tilde{\mathbf{y}}_i = \mathbf{\Lambda}^{-1/2}\mathbf{y}_i, \text{ and } \tilde{\mathbf{A}} = \mathbf{\Lambda}^{-1/2}\mathbf{A}.$$

https://t.me/learningnets

Incorporating these into the adaptive lasso estimation yields the following objective function:

$$\boldsymbol{\mu}^* = \text{argmin}_{\boldsymbol{\mu}} \left((\tilde{\mathbf{y}}_i - \tilde{\mathbf{A}}\boldsymbol{\mu})'(\tilde{\mathbf{y}}_i - \tilde{\mathbf{A}}\boldsymbol{\mu}) + \gamma \sum_{j=1}^{p} \frac{1}{\bar{x}_j} |\mu_j| \right).$$

Ebrahimi et al. [55] test their method in simulation with values of p ranging from 100 to 10,000, so they are able to test very high-dimensional settings. In IC settings, the percentage of false detections is reported, and this reaches the nominal setting of $\alpha = 0.05$ when $p \geq 5000$. The ARL is computed for OC settings for varying shift magnitudes, and the APC strongly approach outperforms the benchmark comparison methods. The ability of the PCSR method to isolate shifted variables is also evaluated via several metrics, including

- false negative percentage (FN%)—the percentate of shifted variables not detected out of the total number of shifted variables;
- false positive percentage (FP%)—the percentage of the nonshifted variables identified out of the total number of nonshifted variables; and
- parameter selection score (PSS)—total number of variables incorrectly classified (either as shifted or nonshifted).

While the FN% is relatively low across multiple shift sizes and simulation settings, the FP% and PSS can be large for small shifts, but both decline rapidly as the shift size increases. PCSR performs similarly to [3], another lasso-based fault isolation method, for processes with nonsparse covariance matrices. However, PCSR is better if the process has a sparse covariance matrix, and the authors of [55] argue that such settings are common in practice since variables are usually only correlated with a small subset of the total monitored variables. Thus, APC-PCSR is a promising method to apply in high-dimensional settings, and its use is illustrated in a case study in Sect. 3.5.2. Jiang et al. [56] also reduce variables for use in PCA-based monitoring but with an optimization algorithm.

Additional Phase II Fault Isolation Methods. Many methods for Phase II fault isolation have been proposed for specific purposes. Jiang et al. [6] propose a variable selection multivariate EWMA (VS-MEWMA) chart that continues to use the L_0 penalty of the VS-MSPC chart of [2]. Yan et al. [57] propose a VS-based control chart for high-dimensional Gaussian mixture models. Kuang et al. [58] incorporate VS into discriminant analysis for multivariate fault isolation. Various types of Kalman filters are used in [59] and [60] to detect and isolate sensor faults.

The Phase II VS-based control charts described herein assume that the mean and covariance matrix are known, so Li et al. [61] introduce a robust self-starting approach that reduces the number of IC observations needed to initiate the VS-MEWMA chart. Liang et al. [62] propose a version of the LEWMA [16] that can be applied to any continuous process generated within the family of elliptical distributions, performing better for heavy-tailed and skewed multivariate

https://t.me/learningnets

distributions. Li et al. [63] develop a robust self-starting monitoring approach for multivariate categorical processes that incorporates VS. Zhao and Gao [64] develop a process control method that monitors changes in the multivariate distribution of variables with VS in regression to isolate variables associated with changes in the distribution. Zarzo and Ferrer [65] use VS combined with partial least squares (PLS) in monitoring and isolating variables associated with changes in variability of a final output variable.

Monitoring the mean and variance are important for processes inside of control loops. When faults occur in a control loop, the control strategy seeks to maintain the variables at their target levels and may mask any changes in the mean of the variable, but correlations among variables may change substantially, making monitoring of the covariance in addition to the mean very important in such settings [66]. Capizzi and Masarotto [17] use VS to jointly monitor the mean and total dispersion. Wang et al. [67] also propose a method to monitor both the mean and covariance with penalized likelihood estimation. Wang and Tsung [22] use VS for adaptive dimension reduction, with a special focus on feedback-controlled processes.

Batch processes are those that operate on a feed input for a given period of time before switching to the next batch. Some methods for variable isolation in batch processes have also been developed. Yan et al. [68] incorporate VS into PLS discriminant analysis. Chu et al. [24] propose a method for batch processes combined with a bootstrap-based variable selection. Yao et al. [69] use two-dimensional PCA for batch processes with a reduced support region.

Some methods have been designed for very high-dimensional settings with variable numbers in the thousands. Kim et al. [70] developed a penalized likelihood-based fault detection approach that uses the L_2-norm regularization with the goal of shrinking all of the variable means to zero. While it was not originally designed for fault isolation, it does perform well in isolating faulty variables in a case study with ten monitored variables. Turkoz et al. [71] develop a Bayesian approach designed for nonnormal and high-dimensional processes that is computationally efficient. Abdella et al. [72] propose a multivariate CUSUM chart that is designed to detect very small shifts in the mean and isolate variables in high dimensional settings.

Bayesian methods have been used to handle a variety of problem-specific challenges. Yan et al. [73] adopt a Bayesian framework to handle the uncertainty in parameter estimation and to handle the choice of a tuning parameter. They apply the method to three faults in the Tennessee Eastman (TE) process and can show when the fault occurs, how long the fault persists, and which variables are shifted. Ge et al. [74,75] propose Bayesian methods to monitor processes with nonlinear and multimode features, with both applied to the TE process. Jiang and Huang [76] use a Bayesian fault diagnosis method for plant-wide process monitoring via local monitors. The TE process will be described in the next section.

https://t.me/learningnets

3.3 Fault classification

As mentioned in the introduction, a different data structure of interest is when data streams from multiple copies of the same fault in the same system have been recorded and labeled. This produces a catalogue of data produced from historical faults. In new or unstudied systems, such a catalogue is unlikely to exist, but oftentimes, in processes that are very well-studied, a catalogue is available. Then, the goal is to take a new set of faulty observations and determine to which historical fault the new fault is most similar. This is the well-known *classification* problem.

These methods are often only applied to a single set of data and are not conducive to repeated simulation studies and their corresponding metrics, such as those discussed in Sect. 3.4. The most commonly used dataset to illustrate the performance of these methods is the Tennessee Eastman (TE) chemical engineering process, and every work mentioned in this section uses data generated from the TE simulation model [77]. With well over 2,000 citations, this model simulates a large set of variables (over 50) that are nonlinearly related and change dynamically over time. Some variables are controlled to meet predetermined thresholds. Matlab® code is available to simulate data under various control strategies and with many different types of faults [78].

A multiclass support vector machine (SVM) is used in [79] to determine to which among a catalogue of historical faults the present set of observations is the closest. The dimension of the data is first reduced with PCA, and parameters in the SVM are optimized using a grid search. Zhang et al. [80] use kernel entropy component analysis (KECA) with one KECA classifier trained for each type of fault. The authors also seek to accommodate multiscale properties, so they implement a multiscale PCA as well. Ragab et al. [81] use Logical Analysis of Data (LAD) to classify faults, comparing their results to many other machine learning methods, such as ANN, decision trees, random forests, k nearest neighbors, quadratic discriminant analysis, and SVM,

Reducing the dimensionality of variables input into classification methods has been shown to reduce errors here as well [20,79]. De Assis Boldt et al. [20] test several cascade feature selection methods to reduce the input features, and then they apply the Extreme Learning Machine (ELM), which is a feedforward NN with a single hidden layer and a linear activation function at the output. Zhao et al. [82] uses a type of recurrent NN, specifically a long short-term memory (LSTM) neural network, that will directly process the raw data without needing to perform feature extraction and classifier design. This model can also account for autocorrelation in the observations. Penalized regression is used in [58] showing that discriminant analysis is equivalent to penalized regression.

While classification of the faults tends to be the focus of these methods, isolating variables is often done as a byproduct of fault identification. Some examples are in [83], where Chiang et al. use genetic algorithms combined with Fisher discriminant analysis to isolate variables, and de Assis Boldt et al. [20] identify variables that are frequently selected in their feature selection step.

https://t.me/learningnets

3.4 Fault isolation metrics

The metrics used to assess the performance of fault detection methods are standardized and well-known. For example, there are several variations of Average Run Length (ARL) that measure the average time that a method requires before signaling a fault when a process is OC. Metrics for IC processes and both Phase I and Phase II settings also exist. Metrics for assessing fault isolation (FI) methods are not as standardized. FI is a multivariate binary classification problem, aiming to correctly classify variables as shifted or unshifted for a fault occurring at a particular time point. We let $\mathcal{P}$ be the set of all variables; $\mathcal{S}$ be the set of shifted variables; and $\mathcal{U}$ be the set of unshifted variables. We denote t^* and $\widehat{t^*}$ as the time steps at which the fault occurs and is detected, respectively. Then, $\widehat{\mathcal{S}}$ is the set of variables identified as shifted, and $\widehat{\mathcal{U}}$ is the set of variables identified as unshifted at $\widehat{t^*}$.

TABLE 3.2 Summary of FI notation for a specific time step at which the fault occurs and is detected, t^* and $\widehat{t^*}$, respectively.

	Classified as Shifted	Classified as Unshifted	p
Truly Shifted	s^+	s^-	$\lvert\mathcal{S}\rvert$
Truly Unshifted	u^-	u^+	$\lvert\mathcal{U}\rvert$
$p^- = s^- + u^-$	$\lvert\widehat{\mathcal{S}}\rvert$	$\lvert\widehat{\mathcal{U}}\rvert$	$p^+ = s^+ + u^+$

FI metrics can be framed in terms of the overlap between $\mathcal{S}$ and $\widehat{\mathcal{S}}$ or between $\mathcal{U}$ and $\widehat{\mathcal{U}}$. Table 3.2 summarizes the notation for a specific time step at which the fault occurs and is detected. Boxes shaded in blue (mid gray shaded in print version) indicate a correctly identified set, and boxes shaded in red (dark gray shaded in print version) indicate an incorrectly identified set. The total number of variables correctly identified as shifted is denoted s^+, and the total number of variables incorrectly identified as shifted is denoted u^-. Similarly, the total number of variables correctly identified as unshifted is denoted u^+, and the total number of variables incorrectly identified as unshifted is denoted s^-. Together, $s^+ + u^- = |\widehat{\mathcal{S}}|$ and $s^- + u^+ = |\widehat{\mathcal{U}}|$ represent the total number of variables identified as shifted and unshifted, respectively. The total number of correctly identified variables is $p^+ = s^+ + u^+$, and the total number of incorrectly identified variables is $p^- = s^- + u^-$.

The choice of FI metric depends on the number of monitored variables. In the low-dimensional setting, it is reasonable to expect every variable to be identified correctly, so some metrics only give a method credit if all shifted and unshifted variables are correctly identified [1,3,7,16,21,46,84,85]. The correctness rate (CR), or simply Correctness, is

$$\text{CR} = \frac{1}{B}\sum_{k=1}^{B} I\left[\widehat{\mathcal{S}}_k = \mathcal{S}\right],$$

https://t.me/learningnets

where B is the number of simulated datasets; $\widehat{\mathcal{S}}_k$ is the set of variables identified as shifted for the kth dataset; and $I[\cdot]$ is an indicator function taking the value one if its argument is true and zero otherwise. In a high-dimensional setting, CR may be close to zero because it is difficult to perfectly identify all shifted variables.

Some metrics give "partial credit" to a method when some but not all of the variables are correctly classified, which is often necessary when p is large. Often, we are interested in the total number of incorrectly identified variables. In this case, Zou et al. [3] use the expected number of errors (ENE), which is defined as

$$\text{ENE} = \frac{1}{B}\sum_{k=1}^{B} p_k^- .$$

A related metric is the expected error rate (*EER*), which is simply ENE/p [1,3,55,85].

3.4.1 Fault isolation errors

Typically, it is beneficial to determine *which* type of errors have been made. As in FD, there are two types of errors: (1) identifying unshifted variables as shifted and (2) failing to identify shifted variables as shifted. First, we present a collection of metrics related to the first type of error. The expected number of false positives (EFP) is the average number of unshifted variables identified as shifted [86], which can be written as

$$\text{EFP} = \frac{1}{B}\sum_{k=1}^{B} u_k^- .$$

In addition to averaging the *number* of unshifted variables identified as shifted, Li et al. [86] also report the expected number of false positives as a *proportion* of all variables, or EFP/p. However, this number may be close to zero if the number of shifted variables is small relative to p, which can suggest that the FI method performs better than it does in reality. A preferable metric is the false positive percentage (FP%) [1,55], which is EFP/$|\mathcal{U}|$. [4] replace $\mathcal{U}$ in the denominator with $\mathcal{S}$ and call their metric the ratio of correct variable selection (RCVS).

The second type of error is failing to identify shifted variables as shifted. The false negative percentage (FN%) is the proportion of shifted variables classified as unshifted [1,55], represented as

$$\text{FN\%} = \frac{1}{B}\sum_{k=1}^{B} \frac{s_k^-}{|\mathcal{S}|}.$$

https://t.me/learningnets

A variant of the FN% is the weighted missed detection rate (wMDR), which also takes into account the *cost* of the error associated with misclassifying each variable using a severity function [86], given as

$$\text{wMDR} = \frac{\sum_j I(j \in \mathcal{S} \cap \widehat{\mathcal{U}}) \cdot s_j(\delta_j)}{\sum_j I(j \in \mathcal{S}) \cdot s_j(\delta_j)},$$

where $s_j(\delta_j)$ is a user-defined severity function of the shift size for the jth variable. The wMDR weights each shifted variable classified as unshifted by the severity of the error, where severe errors are weighted more heavily than minor errors. Li et al. [86] require that $s_j(0) = 1$ and that $s_j(\delta_j)$ be a nondecreasing function as δ_j moves away from zero. The severity function may be set based on knowledge of the system such as economical considerations, external covariates, or prior research [87–90]; if all variables are equally important or there is no information on the variables *a priori*, then the same severity function $s(\cdot)$ may be used for all variables. Setting $s_j(\delta_j) = 1$ for all j is equivalent to the FN%.

3.4.2 Precision and recall

Instead of measuring the frequency of errors, an alternative metric that measures the correct identification of shifted variables is $1 - \text{FN}\%$, which we term Recall. This is also referred to as probability of correct identification and ratio of correct variable selection [6,81,91,4]. Using $\widehat{\mathcal{S}}$ rather than $\mathcal{S}$ in the denominator, the Precision is the proportion of variables *identified* as shifted that are truly shifted [81,91], or

$$\text{Precision} = \frac{1}{B} \sum_{k=1}^{B} \frac{s_k^+}{|\widehat{\mathcal{S}}|}.$$

The harmonic average of the Precision and Recall is termed the F_1-score [55, 91], defined as

$$F_1\text{-score} = 2 \cdot \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}.$$

3.4.3 Phase I FI metrics

All of the aforementioned metrics are useful when performing FI in Phase II settings, but Capizzi and Masarotto [8] propose a Phase I method that simultaneously performs FD and FI. Their simulation study introduces multiple faults, some of which involve multiple shifted variables. They consider the average number of exact and approximate detections, where each shifted variable at each time step at which a shift occurs is considered to be a separate detection. An *exact* detection is correctly identifying a shifted variable at exactly the true time step of the fault, while an *approximate* detection is identifying a shifted variable

https://t.me/learningnets

as shifted within a certain window of the true time step of the fault. For both of these metrics, the closer the value to the true number of detections in the data set, the better the performance of the method. While the focus of [8] is FD&I, these two metrics focus on detection ability of their method.

The detection accuracy and isolation rate (DAIR) proposed by [42] incorporates the accuracy of FD into an FI metric. The DAIR is defined as

$$\text{DAIR} = \frac{1}{B}\sum_{k=1}^{B} K_{\boldsymbol{\eta}}(t^*, \hat{t}_k^*)\left(\frac{p_k^+}{p}\right),$$

which has two components: (1) the detection accuracy, which penalizes a method as $\widehat{t^*}$ gets farther from t^* using a kernel function $K_{\boldsymbol{\eta}}(t^*, \hat{t}^*)$, and (2) the isolation rate, which measures the proportion of correctly classified variables, or p_k^+/p. A simple kernel function is the *rectangular* kernel function, or

$$K_\eta(t^*, \hat{t}^*) = \begin{cases} 1, & |t^* - \hat{t}^*| < \eta, \\ 0, & |t^* - \hat{t}^*| \geq \eta, \end{cases}$$

which gives a method "full credit" if the fault is detected within η time steps of the true fault and "no credit" otherwise. Therefore, if the fault is detected more than η time steps from the true fault location, then DAIR will be zero even if all variables are correctly classified as shifted or unshifted. This metric assesses a method's ability to both detect the fault within the correct timeframe and to isolate the correct variables.

3.4.4 Discussion

It is important to choose metrics that accurately reflect a method's performance and that are based on knowledge of a system, including the number of variables being monitored and the economic costs of errors in FI. Poorly chosen metrics may provide an incomplete picture of a method's performance. For example, a large ENE does not reveal whether false negatives or false positives are more prevalent; a small FP% is incomplete without knowledge of a method's FN%; and even a small FN% is misleading if a shifted variable that the method failed to identify as shifted could cause catastrophic damage to a system or result in hefty fines and lost revenue. Careful consideration should be given so that sufficient insight is provided by the chosen metrics in a simulation study or case study.

3.5 Case studies

As the demand for water resources grows, a new paradigm of decentralized, potable reuse focuses on treating and reusing water at smaller, satellite facilities that collect water close to the source where it is produced. The historical solution has been to expand and upgrade existing, aging centralized treatment systems

https://t.me/learningnets

that collect water and wastewater from an extensive geographic region, which requires costly maintenance and pumping. Furthermore, centralized facilities simply release treated wastewater back into the environment, but decentralized facilities can reuse the water locally for a variety of reuse applications, such as irrigation or drinking. However, decentralized facilities experience higher variability in the quality and quantity of the influent, and it is not cost-effective to have an operator on-site full-time to monitor and maintain the system. Moreover, there are inherent risks in advanced wastewater treatment for potable reuse, including a potential release of contaminants into water supplies, making precise and robust process monitoring imperative to protect human health, the environment, and the facility. Such processes have rarely been a focus of FD&I methods; instead, most such studies are concerned with full-scale wastewater treatment facilities with only a few monitored variables or data from simulation models [11,12,92,93].

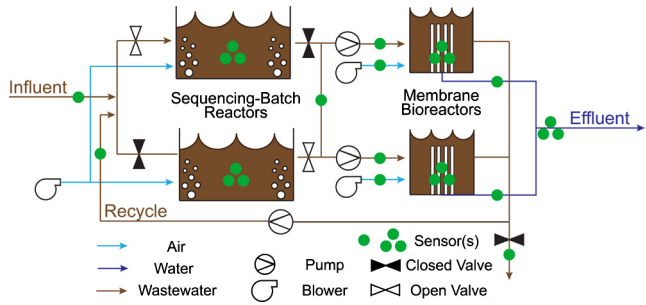


FIGURE 3.8 A schematic of the SBR-MBR operating at Mines Park. Single and triple sensor dots indicate that single and multiple features, respectively, are being recorded at that point in the process. Features such as dissolved oxygen, pressure, flow, volume, total suspended solids, conductivity, and turbidity are automatically sensed every minute.

The Mines Park Water Reclamation Test Site (Mines Park) located in Golden, CO treats approximately 7,000 gallons of municipal wastewater per day using a combination of physical, chemical, and biological processes. Continuously operating for the past ten years, over 30 variables are monitored and recorded every minute, see Fig. 3.8. It receives its wastewater from a student apartment complex on the campus of Colorado School of Mines. Data from this facility have been used for case studies to develop and study multivariate process monitoring methods [42,94–96]. Both of the following examples use data that were generated by this facility.

3.5.1 Retrospective fault isolation

In this example, we illustrate a fault that occurred on January 29, 2018, in which the septic tank influent clogged, causing the sewage level to drop, which in turn exposed a pump, causing the system to overheat and shutdown. In principle,

https://t.me/learningnets

installation of a level switch could have solved this problem, but at the time, there was not such a sensor installed. Variables that should be isolated as shifted are `sewage_level`, bioreactor level, and bioreactor dissolved oxygen, denoted `bio_level`, and `bio_do`, respectively. In this particular system, two bioreactors are operating in parallel, so there are two variables associated with each of `bio_level` and `bio_do`, as shown in Table 3.3.

TABLE 3.3 Response, predictor, and cyclic variables used in the case study.

Response variables	Description
`bio_1_do`, `bio_2_do`	Bioreactor 1 or 2 dissolved oxygen concentration
`bio_1_level`, `bio_2_level`	Bioreactor 1 or 2 level
`bio_1_temp`, `bio_2_temp`	Bioreactor 1 or 2 temperature
`sewage_flow`	Sewage flow rate
`sewage_level`	Sewage level
`ras_ph`	Return activated sludge pH concentration
`ras_tss`	Return activated sludge total suspended solids
`ras_temp`	Return activated sludge temperature
Predictor variables	**Description**
`after_fault`	Obs. after fault corrected on Jan. 25 (ras_tss only)
`bio_1_blower_flow`,	Bioreactor 1 or 2 blower flow rate
`bio_2_blower_flow`	
`ambient_temp`	Ambient temperature
`bio_1_phase_1`, `bio_2_phase_1`	Bioreactor 1 or 2 in phase 1 (mix-fill)
`bio_1_phase_2`, `bio_2_phase_2`	Bioreactor 1 or 2 in phase 2 (react-fill)
`mbr_flux_mode`	Membrane bioreactors 1 and 2 in peak mode
`mbr_1_air_scour_valve`	Membrane bioreactor 1 valve is open
`mbr_1_mode_1`, `mbr_2_mode_1`	Membrane bioreactor 1 or 2 in mode 1 (permeate)
`mbr_1_mode_2`, `mbr_2_mode_2`	Membrane bioreactor 1 or 2 in mode 2 (backflush)
`mbr_1_mode_4`, `mbr_2_mode_4`	Membrane bioreactor 1 or 2 in mode 4 (relaxation)
Cyclic variables	**Description**
`cos_hourly`, `sin_hourly`	Harmonic components for hourly trend
`cos_2hour`, `sin_2hour`	Harmonic components for two-hour trend
`cos_daily`, `sin_daily`	Harmonic components for daily trend

We apply the retrospective change point detection method of [42]. The first step is to determine the training and monitoring windows. In this problem, the training window is five days of one-minute data during January 23–27, 2018 and contains 7,195 observations. The monitoring window begins on January 28, 2018 and lasts for two and a half days (3,750 observations). The next step is to separate the response variables from the predictor and cyclic variables. Response variables are those that need to be monitored and can be indicative of a fault in the process, such as the `sewage_flow` or `bio_level`. Predictor variables

https://t.me/learningnets

are either (1) controlled by the operator, such as the phase of the bioreactor, or (2) can explain variability within the system, but are not controllable, such as `ambient_temp`. A lagged version of each predictor variable is also included because there can be a delay in the effect of the predictor variable on the response variable. We adjust for a known shift in `ras_tss` due to the correction of a prior fault in the membrane bioreactors by including a binary variable to indicate whether an observation occurs before or after the correction. We also include cyclic variables that account for daily trends, two-hour trends, and hourly trends by adding three sine/cosine pairs with corresponding periods. Table 3.3 summarizes the response, predictor, and cyclic variables used in the case study. Then, we detrend the response variables using adaptive lasso [35] with initial parameter estimates provided by ridge regression based only on the data in the training window. Fig. 3.9 gives a heat map of the regression coefficients for each response variable. The color of the boxes indicates the estimated strength and sign of the linear relationship between the predictor or cyclic variables and the response variables; a white box indicates that the variable is excluded from the model. For example, `bio_1_do` has a strong positive linear relationship with the lag of `bio_1_blower_flow`; `ras_tss` has a strong daily trend; and the phase of the bioreactors has a significant impact on the mean of `sewage_flow`.

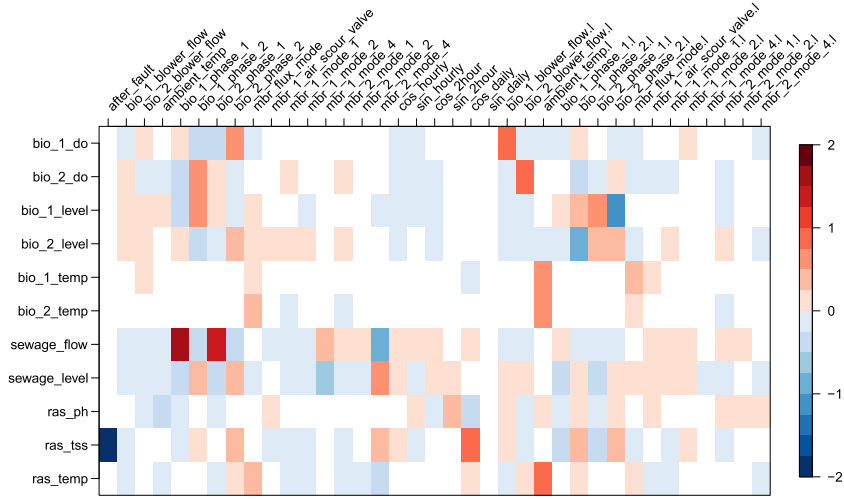


FIGURE 3.9 Heat map for regression coefficients of predictor and cyclic variables (columns) used to model the 11 response variables (rows). A white box corresponds to a predictor or cyclic variable that is excluded from the model, and predictor names ending with ".l" are the lagged versions of the predictor variables.

The estimated regression model is then used to detrend the observations in the monitoring period, and the residuals can be assessed for change points. Fig. 3.10 gives an example of the detrending process for `ras_tss` (left) and `bio_1_do` (right). In Figs. 3.10A and 3.10B, the models are fit using the data

https://t.me/learningnets

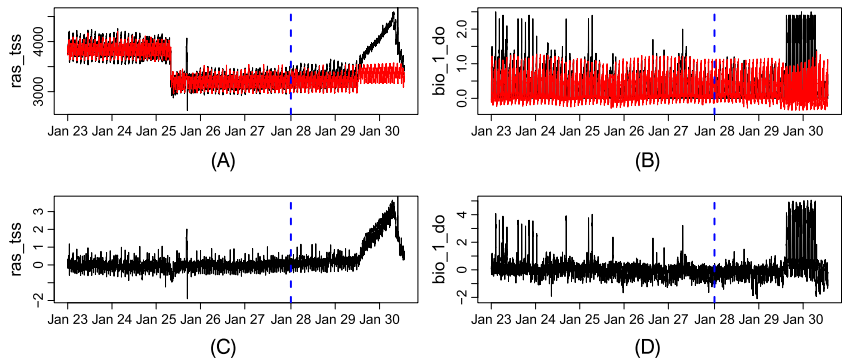


FIGURE 3.10 Original features (top) and residuals (bottom) for ras_tss (left) and bio_1_do (right) with fitted values in red (light gray in print version). The blue vertical dashed line identifies the end of the IC period and the beginning of the monitoring period. Fitted values to the right of the blue vertical dashed line are based on coefficient estimates from the IC period. (A) Original feature for ras_tss ($R^2 = 0.71$); (B) Original feature for bio_1_do ($R^2 = 0.95$); (C) Residuals for ras_tss; (D) Residuals for bio_1_do.

from the training window to the left of the blue vertical dashed line, which account for variability in the data over time not indicative of a fault. In Figs. 3.10C and 3.10D, the residuals still exhibit changes not explained by the fitted model, such as the gradual increase in `ras_tss` and increased variability in `bio_1_do` on January 29. Note that the large downward shift in Fig. 3.10A is due to a separate fault that occurred in the membrane bioreactor, but this is easily removed with an indicator variable.

The training data are also used to estimate the effective sample size to use in the aEBIC criteria to choose the complexity of the fused lasso model. To do this, the PACF plot of the residuals of each variable in the training window is constructed to assess the fit of an AR(1) model, and we see that this model works well for most variables, so we use the PACF plots to estimate the lag-one autocorrelation coefficient for each variable. These estimated autocorrelation coefficients range from 0.35 to 0.99, and they average to 0.83, which indicates quite strong positive temporal autocorrelation; thus, the effective sample size of the data in the monitoring window is estimated to be 339, which is substantially smaller than the observed 3,750 observations.

Finally, the fused lasso model is fit to the residuals in the monitoring period, and the penalty parameter γ is chosen using the aEBIC. Results are shown in Fig. 3.11 where a time series of the standardized residuals is plotted for six key response variables. These six plots depict all but one of the 53 shifts detected by the method. The green shaded (light gray shaded in print version) region is the time period during which the fault is suspected to have occurred; blue down arrows (dark gray in print version) indicate detected decreases; red up arrows (mid gray in print version) indicate detected increases; and the size of the arrows indicate small, medium, or large shifts, respectively. We note that if the effective

https://t.me/learningnets

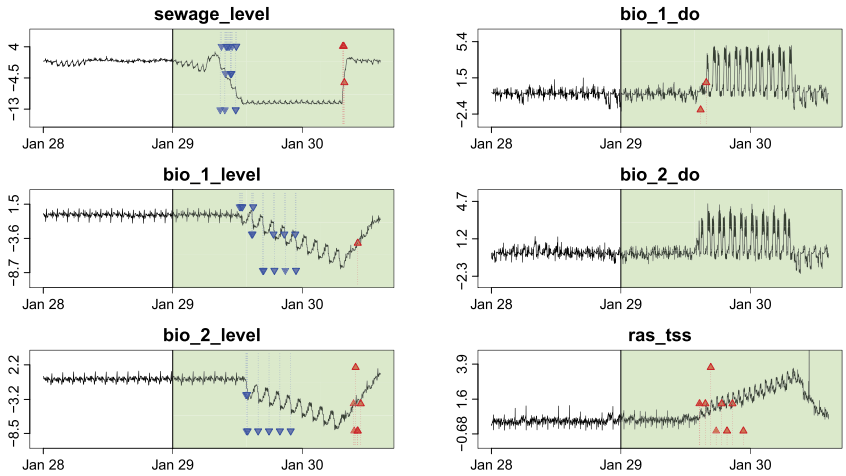


FIGURE 3.11 Time series of detrended variables with detected faults identified with red (mid gray in print version) and blue (dark gray in print version) arrows that are proportional to the size of the shift. The green shaded (light gray shaded in print version) region identifies the suspected time frame of the fault.

sample size adjustment had not been made, the fused lasso approach would have detected 429 shifts, the vast majority of which are spurious. Here, we see that the method detects a series of downward shifts in `sewage_level` beginning around 8:00 a.m. on January 29. Four hours later, numerous consecutive downward shifts are detected in `bio_1_level`, followed by a set of downward shifts in `bio_2_level` an hour later. An increase in `bio_1_do` is detected at 2:45 p.m., but there are no analogous shifts identified in `bio_2_do`. This is expected because the method is designed to detect a shift in the mean, but it appears that the variance has changed. Excluding `bio_2_do`, these results are consistent with the timeline of the fault given by operators. Additionally, the method detects a small upward drift fault in `ras_tss` beginning around 2:30 p.m., which was not initially identified by operators through visual inspection. Around mid-day on January 30, all three of the level variables appear to return to their prefault values, and these are indicated by shifts detected in the upward direction.

3.5.2 Real-time fault isolation

Early in the operation of the SBR-MBR system, two faults occurred within days of each other. First, there was a rapid decline in the level of the membrane bioreactor (MBR) tanks. This left the system susceptible to a second change in the influent water quality following a significant precipitation event that washed deicing salts from the roadways into the sewer system. The presence of deicing salts led to an increase in permeate conductivity and a decrease in return activated sludge (RAS) pH concentration, severely damaging the bio-

https://t.me/learningnets

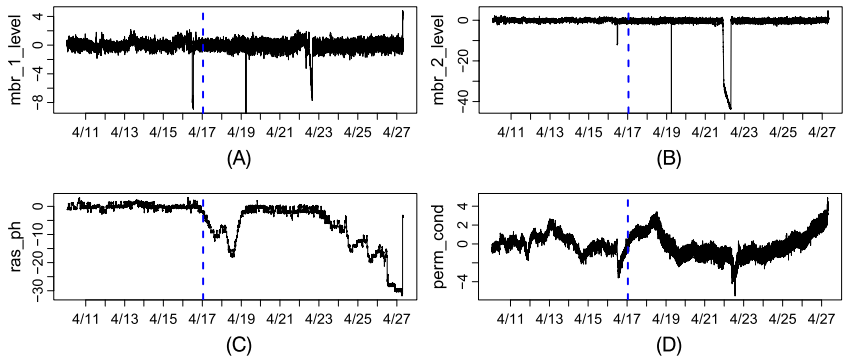


FIGURE 3.12 Residuals for the four variables that are suspected to have shifted: MBR 1 and 2 level, RAS pH, and permeate conductivity. The blue vertical dashed line indicates the end of the training window and the time at which real-time monitoring begins. (A) Residuals for mbr_1_level; (B) Residuals for mbr_2_level; (C) Residuals for ras_ph; (D) Residuals for perm_cond.

logical community of the system. In this case study, we apply a real-time FD&I method to demonstrate how advanced process monitoring methods may be used to identify faults in a system even before they are evident to system operators.

A week of one-minute observations are used as a training data set between April 10–16, 2010, which consists of 10,045 observations. We begin monitoring on April 17 and monitor the system for 11 days (14,859 observations). As in the case study presented in Sect. 3.5.1, we detrend the observations using a set of predictor variables (and their lags) and cyclic variables. A description of the data set, additional details on the detrending process, and the variables used for detrending can be found in [42].

We expect to see a shift in `mbr_1_level`, `mbr_2_level`, `ras_ph`, and `perm_cond`. The residuals of the four shifted variables are given in Fig. 3.12. The residuals for `mbr_1_level`, `mbr_2_level`, and `ras_ph` have a roughly constant mean during the training window; however, there is some nonstationarity still present in `perm_cond`. A downward shift in the residuals of `mbr_1_level` and `mbr_2_level` occurs around April 22 and is more severe for the MBR 2 tank. A decrease in the residuals of `ras_ph` first occurs on April 17 and recovers by April 19, but it begins to decline again around April 23. Around the same time, an increase in the residuals of `perm_cond` begins.

For FD, we apply a dynamic extension of principal component analysis (PCA) by including a lagged version of each detrended variable for monitoring [97–100]. For each observation, we retain the principal components that capture 85% of the variability and calculate Hotelling's T^2 and Q, which is also known as SPE. If either statistic exceeds a nonparametric threshold using a false alarm rate of $\alpha = 0.01$, an observation is flagged as out of control (OC). Additional details on the method can be found in [94] and [95], who also apply variations of PCA to this data set for FD. When an observation is flagged as OC, we ap-

https://t.me/learningnets

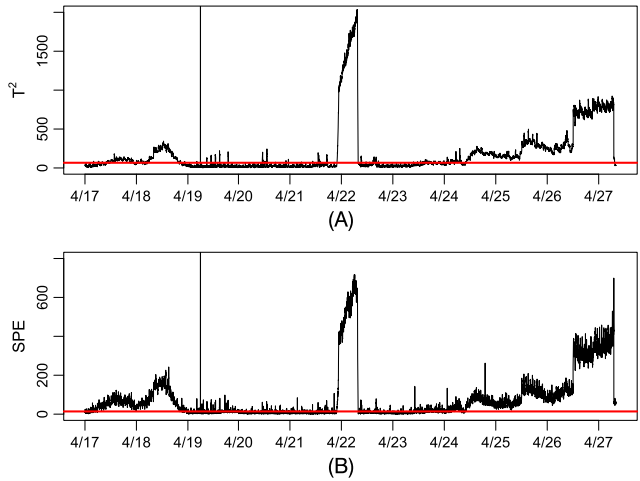


FIGURE 3.13 Monitoring statistics T^2 (top) and Q or SPE (bottom) plotted for the monitoring period. The red horizontal line (mid gray in print version) indicates the non-parametric threshold used to classify an observation as OC. (A) T^2 monitoring statistic; (B) SPE monitoring statistics.

ply the FI method proposed by [55] and described in Sect. 3.2.3.2 to identify the nonzero components of the mean, which correspond to the shifted variables because the detrended data from the training window have been centered.

Fig. 3.13 gives the monitoring statistics, T^2 and SPE, throughout the monitoring period. The nonparametric threshold is determined using a percentile of the kernel density estimate (KDE) of the monitoring statistic from the training period, which is given in red (mid gray in print). The two statistics tend to follow the same pattern, meaning, when one is large, the other is also large, and the monitoring statistics tend to be large when there is a shift in one or more of the variables in Fig. 3.12. There are three time periods during which the statistics consistently exceed their corresponding thresholds: (1) during the first downward shift in `ras_ph` between April 17–19; (2) during the fault in the MBR levels on April 22; and (3) beginning around April 24 through the end of the monitoring period as `ras_ph` and `perm_cond` become more and more OC.

Now that the FD method has detected a fault in the system, the FI method to identify the shifted variables is applied. Fig. 3.14 plots the classification of each variable over the monitoring period as shifted or unshifted, where a black box indicates that the variable is identified as shifted at a given time point and is white otherwise. The variables that we strongly suspect have shifted are highlighted in yellow (light gray in print version). Termed a *checkerboard plot*, these figures are helpful in tracking changes in shifted variables over time [15,73] and show how faults can propagate from one variable to another over time, particularly in those systems with feedback control implemented [66].

https://t.me/learningnets

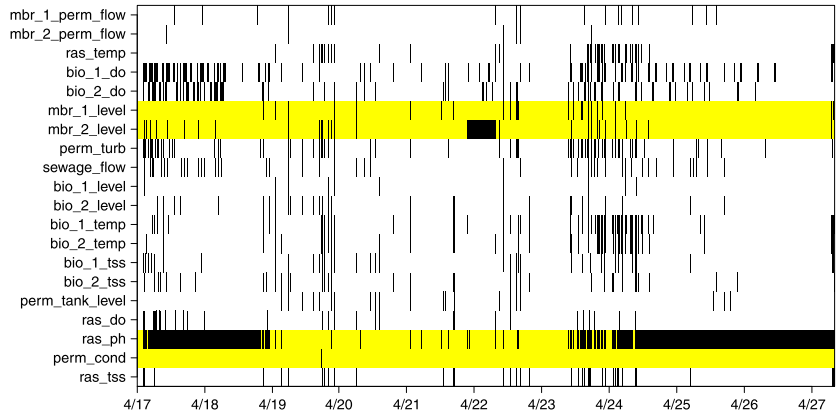


FIGURE 3.14 Checkerboard plot of fault isolation results, where a black box indicates that a variable is identified as shifted at a given time point but is white otherwise. Variables that are suspected to be truly shifted are highlighted in yellow (light gray in print version).

Between April 17–19, `ras_ph` is consistently identified as shifted. The shift in `mbr_2_level` is also clearly identified, but the shift in `mbr_1_level` is not; this is likely due to the fact that the shift is much smaller in magnitude and may have been masked by some extreme values in the training period. Another shift is identified in `ras_ph` at the end of April 23 and is consistently flagged as shifted until the end of the monitoring period. However, `perm_cond` is not identified as shifted, which is likely due to the nonstationarity present in the residuals. Additional shifts identified in variables such as `bio_1_temp`, `bio_2_temp`, and `ras_temp` are likely also caused by nonstationarity or noise in the data and can be disregarded without consistent evidence that the variable is shifted.

Operators were not alerted to the fault until 10:00 a.m. on April 24, and once the underlying problem was identified, it took two months for the system to fully recover and achieve normal operating conditions. Using the FD&I methods presented in this case study would have detected the fault sooner and would have also pointed operators to some of the key variables responsible for the fault, allowing them to more quickly address the problem. In systems with many complexly related variables, FD is often not sufficient because it can be difficult for operators to identify the shifted variables, especially in the case of a drift fault that becomes more severe over time. Therefore, FI is a vital step in diagnosing a fault before the entire system is impacted.

3.6 Further reading

Some excellent review papers that include references for further reading have been published. Capizzi [101] summarizes some of the Phase II VS-based control charts and identifies some open areas of research, such as developing distribution-free methods, methods for autocorrelated data, and VS methods for

https://t.me/learningnets

Phase I analysis. Peres and Fogliatto [102] provide a broad recent review of 30 VS methods integrated into multivariate statistical process monitoring. The limitations of existing methods that are addressed by each of the 30 papers is listed along with the goal of each one. They divide the methods into those that (i) preprocess variables by performing VS prior to process monitoring; (ii) postprocess variables by performing VS after process monitoring; or (iii) iterate between VS and process monitoring. They close with their own list of future research topics, some of which overlap with [101]. Additional unique problems include more work on batch processes and *fault diagnosis* methods, which Reis and Gins [66] also cite as an important area of future work. Indeed, Reis and Gins [66] provide a very nice high-level overview of the developments and future needs in industrial process monitoring that include both VS-based monitoring and isolation and classification methods. They observe that the speed with which a fault is detected is low relative to the time necessary to diagnosis a fault, so they advocate for researchers to focus on *fault prognosis*, which seeks to anticipate faults and intervene in the system to perform maintenance or repairs before faults occur. Finally, Qin [103] provides a more technical review with a focus on both PCA and PLS and the use of reconstruction plots for fault isolation in the context of controlled systems.

References

[1] M. Turkoz, S. Kim, Y.-S. Jeong, K.N. Al-Khalifa, A.M. Hamouda, Distribution-free adaptive step-down procedure for fault identification: non-parametric fault identification approach, Quality and Reliability Engineering International 32 (8) (2016) 2701–2716.

[2] K. Wang, W. Jiang, High-dimensional process monitoring and fault isolation via variable selection, Journal of Quality Technology 41 (2009) 247–258.

[3] C. Zou, W. Jiang, F. Tsung, A LASSO-based diagnostic framework for multivariate statistical process control, Technometrics 53 (3) (2011) 297–309.

[4] C. Zhao, W. Wang, Efficient faulty variable selection and parsimonious reconstruction modelling for fault isolation, Journal of Process Control 38 (2016) 31–41.

[5] N. Shinozaki, T. Iida, A variable selection method for detecting abnormality based on the T^2 test, Communications in Statistics. Theory and Methods 46 (2017) 8603–8617.

[6] W. Jiang, K. Wang, F. Tsung, A variable-selection-based multivariate EWMA chart for process monitoring and diagnosis, Journal of Quality Technology 44 (2012) 209–230.

[7] K. Paynabar, C. Zou, P. Qiu, A change-point approach for phase-I analysis in multivariate profile monitoring and diagnosis, Technometrics 58 (2) (2016) 191–204.

[8] G. Capizzi, G. Masarotto, Phase I distribution-free analysis of multivariate data, Technometrics 59 (2017) 484–495.

[9] M. Hubert, T. Reynkens, E. Schmitt, T. Verdonck, Sparse PCA for high-dimensional data with outliers, Technometrics 58 (4) (2016) 424–434.

[10] D. Dong, T.J. McAvoy, Batch tracking via nonlinear principal component analysis, AIChE Journal 42 (1996) 2199–2208.

[11] F. Baggiani, S. Marsili-Libelli, Real-time fault detection and isolation in biological wastewater treatment plants, Water Science and Technology: a Journal of the International Association on Water Pollution Research 60 (2009) 2949–2961.

[12] D.S. Lee, P.A. Vanrolleghem, Adaptive consensus principal component analysis for on-line batch process monitoring, Environmental Monitoring and Assessment 92 (2004) 119–135.

https://t.me/learningnets

[13] C.F. Alcala, S.J. Qin, Reconstruction-based contribution for process monitoring, Automatica 45 (7) (2009) 1593–1600.
[14] C.F. Alcala, S.J. Qin, Reconstruction-based contribution for process monitoring with kernel principal component analysis, Industrial & Engineering Chemistry Research 49 (17) (2010) 7849–7857.
[15] Z. Yan, Y. Yao, Variable selection method for fault isolation using least absolute shrinkage and selection operator (LASSO), Chemometrics and Intelligent Laboratory Systems 146 (2015) 136–146.
[16] C. Zou, P. Qiu, Multivariate statistical process control using LASSO, Journal of the American Statistical Association 104 (2009) 1586–1596.
[17] G. Capizzi, G. Masarotto, A least angle regression control chart for multidimensional data, Technometrics 53 (3) (2011) 285–296.
[18] Y.-H. Chu, S.J. Qin, C. Han, Fault detection and operation mode identification based on pattern classification with variable selection, Industrial & Engineering Chemistry Research 43 (7) (2004) 1701–1710.
[19] K. Ghosh, M. Ramteke, R. Srinivasan, Optimal variable selection for effective statistical process monitoring, Computers & Chemical Engineering 60 (2014) 260–276.
[20] F. de Assis Boldt, T.W. Rauber, F.M. Varejão, Cascade feature selection and ELM for automatic fault diagnosis of the Tennessee Eastman process, Neurocomputing 239 (2017) 238–248.
[21] I. González, I. Sánchez, Variable selection for multivariate statistical process control, Journal of Quality Technology 42 (3) (2010) 242–259.
[22] K. Wang, F. Tsung, An adaptive dimension reduction scheme for monitoring feedback-controlled processes, Quality and Reliability Engineering International 25 (2009) 283–298.
[23] K. Nishimura, S. Matsuura, H. Suzuki, Multivariate EWMA control chart based on a variable selection using AIC for multivariate statistical process monitoring, Statistics & Probability Letters 104 (2015) 7–13.
[24] Y.-H. Chu, Y.-H. Lee, C. Han, Improved quality estimation and knowledge extraction in a batch process by bootstrapping-based generalized variable selection, Industrial & Engineering Chemistry Research 43 (11) (2004) 2680–2690.
[25] S. Bersimis, A. Sgora, S. Psarakis, Methods for interpreting the out-of-control signal of multivariate control charts: a comparison study, Quality and Reliability Engineering International 33 (2017) 2295–2326.
[26] R.L. Mason, N.D. Tracy, J.C. Young, Decomposition of T^2 for multivariate control chart interpretation, Journal of Quality Technology 27 (1995) 99–108.
[27] R.L. Mason, N.D. Tracy, J.C. Young, A practical approach for interpreting multivariate T^2 control chart signals, Journal of Quality Technology 29 (4) (1997) 396–406.
[28] B.J. Murphy, Selecting out of control variables with the T^2 multivariate quality control procedure, Journal of the Royal Statistical Society. Series D. The Statistician 36 (5) (1987) 571–581.
[29] F. Aparisi, G.A. no, J. Sanz, Techniques to interpret T^2 control chart signals, IIE Transactions 38 (8) (2006) 647–657.
[30] L.-H. Chen, T.-Y. Wang, Artificial neural networks to classify mean shifts from multivariate T^2 chart signals, Computers & Industrial Engineering 47 (2) (2004) 195–205.
[31] S.T.A. Niaki, B. Abbasi, Fault diagnosis in multivariate control charts using artificial neural networks, Quality and Reliability Engineering International 21 (8) (2005) 825–840.
[32] S. Psarakis, The use of neural networks in statistical process control charts, Quality and Reliability Engineering International 27 (5) (2011) 641–650.
[33] M.G. de la Parra, P. Rodriguez-Loaiza, Application of the multivariate T^2 control chart and the Mason–Tracy–Young decomposition procedure to the study of the consistency of impurity profiles of drug substances, Quality Engineering 16 (2003) 127–142.
[34] R. Tibshirani, Regression shrinkage and selection via the lasso, Journal of the Royal Statistical Society, Series B, Methodological 58 (1996) 267–288.

https://t.me/learningnets

[35] H. Zou, The adaptive lasso and its oracle properties, Journal of the American Statistical Association 101 (2006) 1418–1429.
[36] H. Wang, C. Leng, Unified lasso estimation by least squares approximation, Journal of the American Statistical Association 102 (479) (2007) 1039–1048.
[37] P. Zhao, B. Yu, On model selection consistency of lasso, Journal of Machine Learning Research 7 (2006) 2541–2563.
[38] T. Hastie, R. Tibshirani, M. Wainright, Statistical Learning with Sparsity: The Lasso and Generalizations, CRC Press, 2015.
[39] H. Oja, Multivariate Nonparametric Methods with R: An Approach Based on Spatial Signs and Ranks, Springer, 2010.
[40] P. Qiu, Introduction to Statistical Process Control, CRC Press, 2014. [Online]. Available: https://www.crcpress.com/Introduction-to-Statistical-Process-Control/Qiu/p/book/9781439847992.
[41] J. Chen, Z. Chen, Extended Bayesian information criteria for model selection with large model spaces, Biometrika 95 (2008) 759–771.
[42] M.C. Klanderman, K.B. Newhart, T.Y. Cath, A.S. Hering, Fault isolation for a complex decentralized wastewater treatment facility, Journal of the Royal Statistical Society, Series C (2020), submitted for publication.
[43] P. Qiu, D. Xiang, Univariate dynamic screening system: an approach for identifying individuals with irregular longitudinal behavior, Technometrics 56 (2) (2014) 248–260.
[44] P. Qiu, D. Xiang, Surveillance of cardiovascular diseases using a multivariate dynamic screening system, Statistics in Medicine 34 (14) (2015) 2204–2221.
[45] J. Li, P. Qiu, Nonparametric dynamic screening system for monitoring correlated longitudinal data, IIE Transactions 48 (8) (2016) 772–786.
[46] J. Li, P. Qiu, Construction of an efficient multivariate dynamic screening system: construction of an efficient multivariate dynamic screening system, Quality and Reliability Engineering International 33 (8) (2017) 1969–1981.
[47] R. Tibshirani, M. Saunders, S. Rosset, J. Zhu, K. Knight, Sparsity and smoothness via the fused lasso, Journal of the Royal Statistical Society, Series B, Statistical Methodology 67 (2005) 91–108.
[48] S. Zhang, Z. Yan, P. Wu, Z. Zhang, Fault isolation based on Bayesian fused lasso, in: 2017 Chinese Automation Congress (CAC), IEEE, Jinan, 2017, pp. 2778–2783.
[49] C.R. Rojas, B. Wahlberg, On change point detection using the fused lasso method. [Online]. Available: http://arxiv.org/abs/1401.5408, 2014.
[50] R.J. Tibshirani, J. Taylor, The solution path of the generalized lasso, The Annals of Statistics 39 (2011) 1335–1371.
[51] T.B. Arnold, R.J. Tibshirani, Genlasso: path algorithm for generalized lasso problems, R package version 1.4. [Online]. Available: https://CRAN.R-project.org/package=genlasso, 2019.
[52] M.B. Priestley, Spectral Analysis and Time Series, Academic Press, 1981.
[53] I.T. Jolliffe, N.T. Trendafilov, M. Uddin, A modified principal component technique based on the lasso, Journal of Computational and Graphical Statistics 12 (3) (2003) 531–547.
[54] H. Zou, T. Hastie, R. Tibshirani, Sparse principal component analysis, Journal of Computational and Graphical Statistics 15 (2006) 265–286.
[55] S. Ebrahimi, C. Ranjan, K. Paynabar, Large multistream data analytics for monitoring and diagnostics in manufacturing systems, arXiv:1812.10430 [stat.ML], 2018. [Online]. Available: http://arxiv.org/abs/1812.10430.
[56] Q. Jiang, X. Yan, B. Huang, Performance-driven distributed PCA process monitoring based on fault-relevant variable selection and Bayesian inference, IEEE Transactions on Industrial Electronics 63 (2016) 377–386.
[57] D. Yan, S. Zhang, U. Jung, A variable-selection control chart via penalized likelihood and Gaussian mixture model for multimodal and high-dimensional processes, Quality and Reliability Engineering International 35 (2019) 1263–1275.

https://t.me/learningnets

[58] T.-H. Kuang, Z. Yan, Y. Yao, Multivariate fault isolation via variable selection in discriminant analysis, Journal of Process Control 35 (2015) 30–40.
[59] B. Pourbabaee, N. Meskin, K. Khorasani, Sensor fault detection, isolation, and identification using multiple-model-based hybrid Kalman filter for gas turbine engines, IEEE Transactions on Control Systems Technology 24 (2016) 1184–1200.
[60] K. Villez, B. Srinivasan, R. Rengaswamy, S. Narasimhan, V. Venkatasubramanian, Kalman-based strategies for fault detection and identification (FDI): extensions and critical evaluation for a buffer tank system, Computers & Chemical Engineering 35 (5) (2011) 806–816.
[61] W. Li, X. Pu, F. Tsung, D. Xiang, A robust self-starting spatial rank multivariate EWMA chart based on forward variable selection, Computers & Industrial Engineering 103 (2017) 116–130.
[62] W. Liang, D. Xiang, X. Pu, A robust multivariate EWMA control chart for detecting sparse mean shifts, Journal of Quality Technology 48 (2016) 265–283.
[63] J. Li, K. Liu, X. Xian, Causation-based process monitoring and diagnosis for multivariate categorical processes, IISE Transactions 49 (2017) 332–343.
[64] C. Zhao, F. Gao, A sparse dissimilarity analysis algorithm for incipient fault isolation with no priori fault information, Control Engineering Practice 65 (Aug. 2017) 70–82.
[65] M. Zarzo, A. Ferrer, Batch process diagnosis: PLS with variable selection versus block-wise PCR, Chemometrics and Intelligent Laboratory Systems 73 (2004) 15–27.
[66] M.S. Reis, G. Gins, Industrial process monitoring in the big data/industry 4.0 era: from detection, to diagnosis, to prognosis, Processes 5 (2017) 1–16.
[67] K. Wang, A.B. Yeh, B. Li, Simultaneous monitoring of process mean vector and covariance matrix via penalized likelihood estimation, Computational Statistics & Data Analysis 78 (2014) 206–217.
[68] Z. Yan, T.-H. Kuang, Y. Yao, Multivariate fault isolation of batch processes via variable selection in partial least squares discriminant analysis, ISA Transactions 70 (Sep. 2017) 389–399.
[69] Y. Yao, Y. Diao, N. Lu, J. Lu, F. Gao, Two-dimensional dynamic principal component analysis with autodetermined support region, Industrial & Engineering Chemistry Research 48 (2009) 837–843.
[70] S. Kim, M.K. Jeong, E.A. Elsayed, A penalized likelihood-based quality monitoring via L_2-norm regularization for high-dimensional processes, Journal of Quality Technology (2019) 1–16 (online).
[71] M. Turkoz, S. Kim, Y.-S. Jeong, M.K.M. Jeong, E.A. Elsayed, K.N. Al-Khalifa, A.M. Hamouda, Bayesian framework for fault variable identification, Journal of Quality Technology 51 (2019) 375–391.
[72] G.M. Abdella, K.N. Al-Khalifa, S. Kim, M.K. Jeong, E.A. Elsayed, A.M. Hamouda, Variable selection-based multivariate cumulative sum control chart: variable selection based MCUSUM, Quality and Reliability Engineering International 33 (2017) 565–578.
[73] Z. Yan, Y. Yao, T.-B. Huang, Y.-S. Wong, Reconstruction-based multivariate process fault isolation using Bayesian lasso, Industrial & Engineering Chemistry Research 57 (2018) 9779–9787.
[74] Z. Ge, F. Gao, Z. Song, Two-dimensional Bayesian monitoring method for nonlinear multimode processes, Chemical Engineering Science 66 (2011) 5173–5183.
[75] Z. Ge, M. Zhang, Z. Song, Nonlinear process monitoring based on linear subspace and Bayesian inference, Journal of Process Control 20 (2010) 676–688.
[76] Q. Jiang, B. Huang, Distributed monitoring for large-scale processes based on multivariate statistical analysis and Bayesian method, Journal of Process Control 46 (2016) 75–83.
[77] J.J. Downs, E.F. Vogel, A plant-wide industrial process control problem, Computers & Chemical Engineering 17 (1993) 245–255.
[78] N. Ricker, Optimal steady-state operation of the Tennessee Eastman challenge process, Computers & Chemical Engineering 19 (1995) 949–959.
[79] X. Gao, J. Hou, An improved SVM integrated GS-PCA fault diagnosis approach of Tennessee Eastman process, Neurocomputing 174 (2016) 906–911.

https://t.me/learningnets

[80] H. Zhang, Y. Qi, L. Wang, X. Gao, X. Wang, Fault detection and diagnosis of chemical process using enhanced KECA, Chemometrics and Intelligent Laboratory Systems 161 (2017) 61–69.
[81] A. Ragab, M. El-Koujok, B. Poulin, M. Amazouz, S. Yacout, Fault diagnosis in industrial chemical processes using interpretable patterns based on logical analysis of data, Expert Systems with Applications 95 (2018) 368–383.
[82] H. Zhao, S. Sun, B. Jin, Sequential fault diagnosis based on LSTM neural network, IEEE Access 6 (2018) 12929–12939.
[83] L.H. Chiang, R.J. Pell, M.B. Seasholtz, Multivariate analysis of process data using robust statistical analysis and variable selection, IFAC Proceedings Volumes 37 (2004) 269–274.
[84] Y. Shang, X. Zi, F. Tsung, Z. He, LASSO-based diagnosis scheme for multistage processes with binary data, Computers & Industrial Engineering 72 (2014) 198–205.
[85] J. Kim, M.K. Jeong, E.A. Elsayed, K.N. Al-Khalifa, A.M.S. Hamouda, An adaptive step-down procedure for fault variable identification, International Journal of Production Research 54 (11) (2016) 3187–3200.
[86] W. Li, D. Xiang, F. Tsung, X. Pu, A diagnostic procedure for high-dimensional data streams via missed discovery rate control, Technometrics (2019) 1–27.
[87] P.H. Westfall, S.S. Young, Resampling-Based Multiple Testing, Wiley, New York, 1993.
[88] Y. Benjamini, R. Heller, False discovery rates for spatial signals, Journal of the American Statistical Association 102 (480) (2007) 1272–1281.
[89] W. Sun, B.J. Reich, T.T. Cai, M. Guindani, A. Schwartzman, False discovery control in large-scale spatial multiple testing, Journal of the Royal Statistical Society, Series B, Statistical Methodology 77 (1) (2015) 59–83.
[90] C. Xing, J.C. Cohen, E. Boerwinkle, A weighted false discovery rate control procedure reveals alleles at FOXA2 that influence fasting glucose levels, American Journal of Human Genetics 86 (3) (2010) 440–446.
[91] H. Yan, K. Paynabar, J. Shi, Real-time monitoring of high-dimensional functional data streams via spatio-temporal smooth sparse decomposition, Technometrics 60 (2) (2018) 181–197.
[92] C.K. Yoo, K. Villez, S.W. Van Hulle, P.A. Vanrolleghem, Enhanced process monitoring for wastewater treatment systems, EnvironMetrics 19 (2008) 602–617.
[93] L. Corominas, K. Villez, D. Aguado, L. Rieger, C. Rosén, P.A. Vanrolleghem, Performance evaluation of fault detection methods for wastewater treatment processes, Biotechnology and Bioengineering 108 (2011) 333–344.
[94] G.J. Odom, K.B. Newhart, T.Y. Cath, A.S. Hering, Multistate multivariate statistical process control, Applied Stochastic Models in Business and Industry 34 (2018) 880–892.
[95] K. Kazor, R.W. Holloway, T.Y. Cath, A.S. Hering, Comparison of linear and nonlinear dimension reduction techniques for automated process monitoring of a decentralized wastewater treatment facility, Stochastic Environmental Research and Risk Assessment 30 (2016) 1527–1544.
[96] P. Krupskii, F. Harrou, A.S. Hering, Y. Sun, Copula-based monitoring schemes for non-Gaussian multivariate processes, Journal of Quality Technology (2020), now online.
[97] S.W. Choi, I.-B. Lee, Nonlinear dynamic process monitoring based on dynamic kernel PCA, Chemical Engineering Science 59 (24) (2004) 5897–5908.
[98] U. Kruger, Y. Zhou, G.W. Irwin, Improved principal component monitoring of large-scale processes, Journal of Process Control 14 (8) (2004) 879–888.
[99] J. Mina, C. Verde, Fault detection for large scale systems using dynamic principal components analysis with adaptation, IFAC Proceedings Volumes (2) (2007) 185–194.
[100] W. Ku, R.H. Storer, C. Georgakis, Disturbance detection and isolation by dynamic principal component analysis, Chemometrics and Intelligent Laboratory Systems 30 (1) (1995) 179–196.
[101] G. Capizzi, Recent advances in process monitoring: nonparametric and variable-selection methods for Phase I and Phase II, Quality Engineering 27 (2015) 44–67.

https://t.me/learningnets

[102] F.A.P. Peres, F.S. Fogliatto, Variable selection methods in multivariate statistical process control: a systematic literature review, Computers & Industrial Engineering 115 (2018) 603–619.
[103] S.J. Qin, Survey on data-driven industrial process monitoring and diagnosis, Annual Reviews in Control 36 (2) (2012) 220–234.

https://t.me/learningnets

Chapter 4

Nonlinear latent variable regression methods

4.1 Introduction

Anomaly detection in modern processes is indispensable for ensuring optimal operating conditions and enhancing safety and avoid system crashes. To monitor many key variables simultaneously in engineering and environmental processes, multivariate monitoring techniques exploit the high correlation and redundancy and generate a reduced set of uncorrelated variables. Although data-driven methods have shown some successes in-process monitoring, their efficiency is mainly related to the quality of extracted features. In other words, the accuracy of the model plays an important role in fault detection.

Generally speaking, developing a flexible model faces huge challenges which include collinearity among the measured variables, nonlinearity, and noise in the measured data. Latent variable regression (LVR) techniques have demonstrated good capacity for handling multivariate correlated data by exploiting a high degree of redundancy in data and generating a reduced number of uncorrelated variables. Such techniques lead to well-conditioned models [1]. Linear LVR methods have been extensively applied in statistical process monitoring, such as principal component analysis (PCA) [2,3], partial least squares (PLS) [2,4,5], and regularized canonical correlation analysis (RCCA) [6–9], and could handle multidimensionality and cross-correlations. The key idea in LVR methods is to apply appropriate multivariate dimension-reduction techniques according to the features of a process, and use control charts to monitor more informative variables in a lower dimension. However, this might not be adequate for environmental and engineering processes when nonlinearity prevails.

Indeed, because most practical data from various industrial applications are statistically challenging, due to their high-dimensional character, relationships among variables are nonlinear, and thus a variety of fault types can occur. Despite the successful application of LVR methods, their use is limited to capture linear information in the data. Thus, they result in poor performance, misleading insights, and loss of relevant information when used for highly nonlinear processes modeling and monitoring [10]. Indeed, selecting the optimal number of LVs when applying the linear LVR-based schemes to nonlinear processes is not an easy task. This is mainly because of nonlinear effects that distribute in a nonuniform manner among the LVs (i.e., LVs considered as less rele-

https://doi.org/10.1016/B978-0-12-819264-5.00010-3
Copyright © 2021 Elsevier Inc. All rights reserved.
https://t.me/learningnets

vant) may actually have a significant impact. Accordingly, flexible methods are required for suitable feature extraction to handle nonlinear processes. To capture nonlinearities in data, numerous nonlinear LVR strategies exist in the literature. Various LVR models for nonlinearity are available, such as Kernel PCA (KPCA), locally linear embedding (LLE) [11–13], and models based on polynomial functions, smoothing splines, artificial neural networks, and kernel learning [14–17].

Despite the successful application of LVR models (PLS and PCA), they are limited to describe only linear correlations in data and thus can result in a significant loss of substantial information when handling process nonlinearity [10]. To alleviate the limitations of linear LVR methods for nonlinear input–output processes, several nonlinear PLS (NLPLS) methods have been designed for improving the extraction of features in the nonlinear data. Various nonlinear extensions of the NLPLS have been designed in the literature, such as quadratic, neural network, fuzzy, and kernel PLS [18–22]. For instance, in [23,24], an improved NLPLS method has been introduced by applying both smoothers and spline-based additive nonlinear regression methods to the extracted LVs. In [18,25], QPLS is developed by modeling the inner relation using a quadratic function. Nevertheless, quadratic functions are not flexible enough for modeling complex processes. Similarly, several nonlinear extensions have been designed for modeling the inner relationship in PLS, such as neural network PLS [19,20] and fuzzy PLS [22,21]. In [26], Malthouse et al. proposed a nonlinear PLS method using an artificial neural network to model the inner relational. Different nonlinear methods are used to effectively describe the inner relation in PLS, such as the feed-forward neural network [19,20], the Takagi–Sugeno–Kang (TSK) relations [21], and the quadratic fuzzy function [22]. Another nonlinear PLS model, called kernel PLS, is designed by projecting the input data to a feature space using kernel functions where a linear PLS is performed [16].

For input-space models (such as PCA), nonlinear PCA (NLPCA) is an extension of PCA that allows extracting nonlinear relationships among the process variables. This can be obtained by mapping the measured variables onto curves and surfaces rather than lines and planes, as in the case of linear PCA. Also, different forms of neural network models have been proposed to capture the nonlinearity among the principal components [27,28,14], where the NLPCA model parameters are determined by minimizing the mean-squared error between the model prediction and the measured data. Another way of dealing with nonlinearities in process data is by using kernel PCA (KPCA) [20,21,13]. The key concept of the KPCA is first to project the input space into a high-dimensional feature space (transformed space) using a nonlinear mapping, and then extract principal components (PCs) in the feature space [12,29]. Essentially, KPCA can efficiently calculate PCs in feature spaces, using integral operators and nonlinear kernel functions [11,30]. The main advantage of the KPCA method over nonlinear PCA methods is that nonlinear optimization is totally avoided.

https://t.me/learningnets

Monitoring multivariate nonlinear processes is obviously more challenging than monitoring linear processes, and anomaly may get smeared in the inspected process. Since most of the gathered data from environmental and industrial processes are inherently nonlinear, nonlinear LVR methods with the needed flexibility to represent any nonlinear relationships are required. Recently, nonlinear LVR-based anomaly detection techniques have gained considerable attention by researchers and engineers for the monitoring of complex processes. This is because of the high capability of nonlinear LVR models to extract both linear and nonlinear correlations among process variables. Generally, for process monitoring purposes, nonlinear LVR, such as nonlinear PLS and PCA methods, are first constructed based on nominal measurements from the monitored process running under normal conditions. This makes it possible to design a reference model reflecting the nominal behavior of the inspected process that can be used to detect potential anomalies. After that, anomalies are flagged if the measurements diverge from the nominal operating region in the latent space or in the residual space. Indeed, different fault detection schemes can be used to check generated residuals including SPE, T^2, univariate monitoring techniques, or binary clustering algorithms.

Here, we review the basic problem formulation and algorithm of the nonlinear PLS method. For a better understanding of NLPLS, two forms of nonlinear PLS algorithms (polynomial and ANFIS) are presented. For the input-space model, we present a KPCA method that can be used to model and monitor nonlinear processes. This chapter provides a detailed case study comparing the advantages of the KPCA method over the PCA method for monitoring nonlinear processes. Data from an actual WWTP are used for comparison. In addition, simulated plug flow reactor data are used to show the performance of NLPLS for detecting various fault types. This chapter concludes with a presentation of a set of firmly established conclusions and a list of topics of interest for future research.

4.2 Limitations of linear LVR methods for process monitoring

For large-scale systems, such as wastewater treatment and petrochemical plants, the design of model-based monitoring approaches requires significant effort. Data-based techniques, such as LVR methods, provide a better, easier alternative. Techniques based on dimensionality reduction, such as PCA and PLS, have received considerable attention. Generally speaking, the successful application of a monitoring method relies on the quality of the data and the used extracted features from the measured data. The well-known LVR methods have been successfully used to capture the information of the system. Basically, LVR models project the original data space into a latent space, thus reducing the dimension of the original data space while considering the correlation among attributes. This helps predict the output variable in the reduced latent space and results in an improved prediction ability of the model. The main drawback of the linear

https://t.me/learningnets

LVR methods is that it assumes a linear relationship between input and output. Specifically, every LV is a linear combination of original variables, uncorrelated to all other LVs. Therefore, most of the LVR model parameter estimation uses the linear regression technique to estimate the regression coefficients. Thus, the linear LVR model has a limited ability to capture the linear relationship between the correlated inputs and outputs. In other words, linear LVR approaches are favorable, as long as no process nonlinearity can be expected. However, with the higher requirements for achieving the desired expectations, the process evolutions in modern industrial systems are significant, and the correlation between the process variables is nonlinear. In practice, processes that feature nonlinearity introduce more challenges for anomaly detection. Thus, the use of linear LVR methods to monitor real-world data with nonlinearity can lead to the loss of important information, resulting in poor model prediction and misleading analysis. The major limitation of LVR-based approaches is their ability to extract only linear features from the data; they also fail to describe the actual behavior. To alleviate this drawback, numerous nonlinear LVR methods have been developed to allow nonlinear data to be handled. Nonlinear relationships between variables are better modeled and described by nonlinear (curved) latent variables. Modeling both linear and nonlinear relationships can b achieved, using nonlinear LVR methods by projecting the original variables onto curves or surfaces instead of lines or planes. Fig. 4.1A–B illustrates the concept of using linear and nonlinear PCAs. In Fig. 4.1A, one can see that the data can be approximated, using two PCs; Fig. 4.1B shows that data can be approximated using one nonlinear component. It can be concluded from Fig. 4.1 that the generalization of the LV from lines to curves offers a better description of nonlinear relationships between process variables.

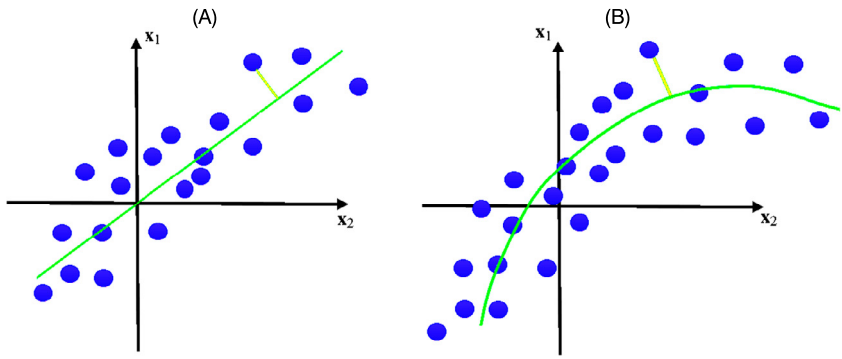


FIGURE 4.1 Linear and nonlinear PCA.

In the case of the input–output model, the latent space of the input and out of the LVR model is used to capture the nonlinearity in the data. Fig. 4.2 offers an illustrative example showing that the inner model linking score vectors of the LVR model is represented by a curve, instead of a straight line. It can be

https://t.me/learningnets

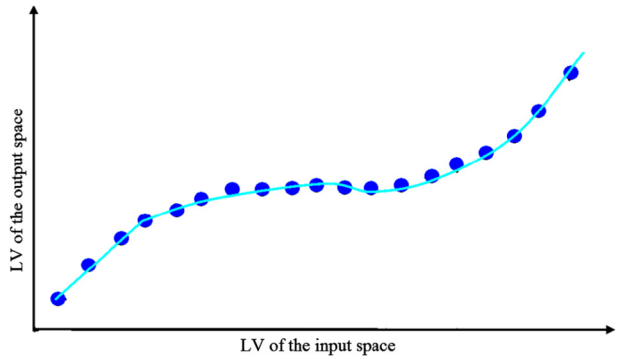


FIGURE 4.2 Mapping between PLS scores vectors.

observed that the linear model will fail if used to capture the process nonlinearity. Thus, it requires a nonlinear mapping between the score vectors of the LVR model. There are many nonlinear mapping functions that are used to map between the input and output latent space in the literature. The inner space (i.e., latent space) input and output in LVR is mapped as a nonlinear function. In this chapter, two different nonlinear methodologies are presented. First, we discuss the nonlinear mapping done through a polynomial function; we then present the adaptive-network based fuzzy inference system (ANFIS).

4.3 Developing nonlinear LVR methods for process monitoring

This section introduces first the general concept of nonlinear PLS models, and then presents two nonlinear PLS models, the polynomial PLS and adaptive neuro-fuzzy inference system (ANFIS)-PLS models, which both use the original iterative linear PLS framework. Lastly, we discuss fault detection schemes based on nonlinear PLS models.

4.3.1 Nonlinear partial least squares

In modern industrial processes, engineers are frequently required to estimate, from available data, some important variables that can be expensive to measure and not easy to collect. Soft sensor techniques have been widely applied for estimating these key variables based on easily measured variables. For input–output multivariate data, PLS is a well-known soft sensor technology that has been successfully used in industry, in order to extract relationships between two sets of variables, inputs and outputs. Specifically, linear PLS is usually used to predict key variables that cannot be easily acquired (called outputs) from other available measurements (called inputs). For instance, PLS models can be used to predict the compositions in a distillation column, using temperature and pressure measurements at different trays of the column. It can be used to predict

https://t.me/learningnets

power production in wind turbines from measurements of various meteorological conditions (wind direction, wind speed, etc.). In PLS, the input and output matrices, $\mathbf{X} \in \mathbb{R}^{n \times m}$ and $\mathbf{Y} \in \mathbb{R}^{n \times p}$, are expressed as follows [31,32]:

$$\begin{cases} \mathbf{X} = \sum_{i=1}^{l} \mathbf{t}\mathbf{p}_i^T = \mathbf{T}\mathbf{P}^T + \mathbf{G}, \\ \mathbf{Y} = \sum_{i=1}^{l} \mathbf{u}\mathbf{q}_i^T = \mathbf{U}\mathbf{Q}^T + \mathbf{F}, \end{cases} \tag{4.1}$$

where matrices $\mathbf{T} \in \mathbb{R}^{n \times l}$ and $\mathbf{U} \in \mathbb{R}^{n \times q}$ contain the retained principal components (latent variables) of the input and output data, respectively, matrices $\mathbf{P} \in \mathbb{R}^{m \times l}$ and $\mathbf{Q} \in \mathbb{R}^{p \times q}$ contain the input and output loading vectors, respectively, and matrices $\mathbf{G}$ and $\mathbf{F}$ are the input and output residual matrices, respectively. The model in Eq. (4.1) is usually called the "outer model". The representation of the input and output matrices (shown in the outer model (4.1)) is determined by maximizing the covariance between the input and output latent variables, i.e., $\mathbf{T}$ and $\mathbf{U}$. Then, the model relating the input and output principal components (called inner model) can be obtained as

$$\mathbf{U} = \mathbf{T}\mathbf{B} + \mathbf{H}, \tag{4.2}$$

where $\mathbf{B}$ is a matrix containing the model parameters relating the input and output principal components, and $\mathbf{H}$ is a residual matrix.

However, the natural data from engineering and environmental processes cannot well meet the linearity assumption and often exhibits significant nonlinearity. Accordingly, nonlinear PLS models are required. Two families of nonlinear PLS modeling techniques can be distinguished. The first family consists of techniques where the input and output data matrices are projected onto nonlinear surfaces [33], while the relationship between $\mathbf{T}$ and $\mathbf{U}$ (shown in Eq. (4.2)) is assumed to be linear. Methods in this first nonlinear PLS family have received very limited attention in prediction and process monitoring. This is mainly because these methods do not retain the properties of the linear methodology, and the mapping of the input and output data matrices is costly and challenging [33]. In the second family of techniques, on the other hand, the input and output matrices are projected linearly, but the relationship between any input principal component t and any output principal component u is modeled using a nonlinear function, i.e.,

$$\mathbf{u} = f(\mathbf{t}) + \mathbf{h}, \tag{4.3}$$

where $f(\cdot)$ is a continuous nonlinear function relating $\mathbf{u}$ to $\mathbf{t}$, and $\mathbf{h}$ is the residual.

Here, the inner relationship between the latent space of the input and output are mapped using a nonlinear function. This mapping helps capture the nonlinear relationship of the data set, allowing the decomposition of a multivariate regression problem into a few univariate regression problems. The linear PLS outer projection is used as a dimension-reduction tool to remove collinearity,

https://t.me/learningnets

and the nonlinear mapping (e.g., polynomial function or ANFIS) is used in the inner model of PLS. The input latent variable $\mathbf{t}_i$ and the output $\mathbf{u}_i$ are used to estimate the inner polynomial or ANFIS model. The parameters of the $f(\mathbf{t}_j)$ are to be chosen for minimizing $\mathbf{h}_j$ without overfitting.

In summary, by keeping the outer relation, NLPLS maintains the linear PLS concept that original variables are projected along with the directions that maximize the covariance. The conceptual presentation of a nonlinear PLS is given in Fig. 4.3 and sketched in Table 4.1.

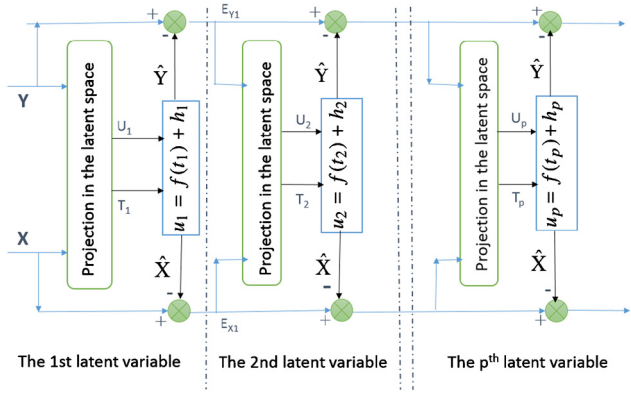


FIGURE 4.3 Nonlinear PLS model.

TABLE 4.1 Main steps of nonlinear PLS modeling procedure.

1. Process the input and output data, i.e., $\mathbf{X}$ and $\mathbf{Y}$, to have zero mean and unit variance. Set $J = 1$ and assume $\mathbf{E}_{X_j} = \mathbf{X}$ and $\mathbf{E}_{Y_j} = \mathbf{Y}$
2. For each J, set $\mathbf{u}$ equal to a column of $\mathbf{E}_{Y_{J-1}}$
3. Perform the outer transform of PLS model using NIPALS algorithm:
   - $\mathbf{w} = \dfrac{\mathbf{u}^T \mathbf{E}_{X_j}}{\mathbf{u}^T \mathbf{u}}$
   - Normalize $\mathbf{u}$ to have unit length
   - Evaluate the scores, $\mathbf{t} = \dfrac{\mathbf{E}_{X_j} \mathbf{w}}{\mathbf{w}^T \mathbf{w}}$
   - Evaluate the new $\mathbf{u}$ vector, $\mathbf{u} = \dfrac{\mathbf{E}_{Y_j} \mathbf{q}}{\mathbf{q}^T \mathbf{q}}$
4. Find a nonlinear relationship $f(\cdot)$, which predicts the output LV $\mathbf{u}$ using input LV $\mathbf{t}$,

$$\mathbf{u} = f(\mathbf{t}) + \mathbf{h}. \tag{4.4}$$

   Then we get the estimated value $\hat{\mathbf{u}}$, $\hat{\mathbf{u}} = f(\mathbf{t})$
5. Determine the input and output space latent variables
6. Calculate the residual matrices $\mathbf{E}_{X_j} = \mathbf{E}_{X_{j-1}} - \mathbf{t}\mathbf{p}^T$ and $\mathbf{E}_{Y_j} = \mathbf{E}_{Y_{j-1}} - \hat{\mathbf{u}}\mathbf{q}^T$
7. Update $J = J + 1$, then return to step 2 until all the latent variables of PLS are evaluated

https://t.me/learningnets

In this section, we present only nonlinear PLS models, polynomial PLS and ANFIS-PLS, that both use the original iterative linear PLS framework.

4.3.1.1 Polynomial PLS modeling algorithm

As discussed above, it is possible to represent the inner model by any continuous nonlinear function linking u to t. Numerous models have been investigated in the literature to represent this nonlinear function, and include polynomial functions, smoothing splines, artificial neural networks, and kernel learning [17,34,16]. This can be performed by changing only the detail of the nonlinear regression in step 4 of the algorithm. Here, we present the basic idea behind the polynomial PLS modeling, which provides simple and first nonlinear PLS models to capture the nonlinearity in the data by fitting the optimal polynomial function in the latent space of the PLS algorithm. The outer loop of the PLS model estimation is untouched. The main advantage of this method is that it will result in the multivariate problem reduction into a univariate regression problem.

The core concept in a polynomial PLS is to replace the linear inner relation by a higher-order polynomial function in order to learn nonlinearity in data and achieve more flexibility. A general form of the inner model can be expressed as

$$u = f(t, \gamma) + h, \tag{4.5}$$

where $f(\cdot)$ us a polynomial function with a selected degree and γ contains the coefficients of the polynomial to be estimated. The goal of this model is to capture nonlinear features and data with more complex curvature characteristics by using a polynomial function in the inner model of a linear PLS. For instance, the polynomial PLS method uses a simple quadratic function to describe the inner model (called Quadratic PLS) presented in [18] as

$$u = \gamma_0 + \gamma_1 t + \gamma_2 t^2 + h. \tag{4.6}$$

However, using a quadratic function may not be sufficient to learn the strong nonlinearity between LVs. Improved prediction capability could be anticipated by using polynomial functions of a higher-order [17]. For instance, in [17], a spline PLS model is proposed by using a cubic spline function for modeling the PLS inner model. Implementing polynomial PLS with high order will offer more flexibility to learn the nonlinearity between the PCs of the input and output variables; however, this makes the implementation of the algorithm challenging and cumbersome. Table 4.2 summarizes the main steps of the polynomial PLS algorithm.

Overall, the polynomial PLS, or PLS with a polynomial inner model, is basically using a polynomial or spline function in the inner model to enhance the prediction capability of the linear PLS [35]. However, the forms of the polynomial and spline are limited and computationally costly. In [36], fractional polynomials are used to model the inner relation in PLS. The main reason for

https://t.me/learningnets

TABLE 4.2 Main steps of polynomial PLS modeling algorithm.

1. Autoscale the input–output data (faultless data), $\mathbf{X}$ and $\mathbf{y}$
2. Let $\mathbf{E}_{\mathbf{X_j}} = \mathbf{X}$, $\mathbf{E}_{\mathbf{y_j}} = \mathbf{y}$, and $j = 1$
3. Calculate the loading vector and latent variables for nonlinear PLS using NIPALS procedure [19]
4. Determine the polynomial function $f(\cdot)$ that predicts the LV response $\mathbf{u_j}$ using the input latent variable t_j,

$$u_j = \gamma_0 + \gamma_1 t_j + \gamma_2 t_j^2 + \gamma_3 t_j^3 + \cdots + \epsilon_j. \tag{4.7}$$

   The estimated model is computed so that the regression error is minimized and overfit is avoided
5. Calculate the residual matrices $\mathbf{E}_{X_j} = \mathbf{X} - \hat{\mathbf{X}}$ and $\mathbf{E}_{y_j} = \mathbf{y} - \hat{\mathbf{y}}$
6. Increment $j = j + 1$ and go to step 3 until all LVs are computed. Cross-validation is used to find the optimized number of LVs, where the output prediction mean squared error (MSE) is minimized

using fractional polynomials is their ability to produce a better description of data with fewer terms than conventional polynomials [37]. Thus, they are relatively flexible and parsimonious, which makes them more suitable to use in NLPLS than the fixed order polynomials. To alleviate these limitations, other extensions of nonlinear PLS, for example, those based on neural networks (NN), have been developed in order to approximate a nonlinear function of the inner model with more flexibility and fewer assumptions [38]. By using NN to learn and construct the inner model of PLS, the residual can be reduced and prediction improved [39,40,19]. Such network PLS methods, however, are not parsimonious and have the tendency to overfit the data [41]; they are also unstable [42]. Recently, Liu et al. developed a nonlinear PLS model by using a Gaussian process regression, a powerful nonlinear model to construct a nonlinear model between each pair of latent variables in the PLS [43]. In their model, augmented data matrices are included to achieve improved prediction performance. In [44], an adaptive neuro-fuzzy inference system (ANFIS) model is used to model the inner relation between the input and output PCs in order to construct a reliable nonlinear PLS model.

4.3.2 ANFIS-PLS modeling framework

This section discusses the basic concept of nonlinear PLS, using the ANFIS method, a robust method to tune the parameters of nonlinear inner relations of the PLS model. Fuzzy model-based fault detection approaches have sparked a flurry of research studies over the past decade, in several fields of engineering [45,46]. In [47], a nonlinear fuzzy PLS using the Takagi–Sugeno–Kang (TSK) fuzzy model has been introduced and showed better prediction capability, compared with the conventional PLS. It has also been shown that the fuzzy PLS model overcomes the quadratic PLS and the traditional linear PLS when

https://t.me/learningnets

applied to nonlinear biological processes [46]. By merging the desirable properties of fuzzy logic and neural networks, the ANFIS showed greater capacity to handle process nonlinearity [45]. It should be pointed out that the ANFIS parameters are optimized online by the self-learning ability of NN. ANFIS is a hybrid system that combines the subjective knowledge representation using fuzzy inference and the learning ability of artificial neural networks. Thus, it provides a greater ability to adapt its membership function of a fuzzy model to achieve the desired performance [48]. The fuzzy "if–then" rules with assigned membership functions have been used to relate the inner relationship of the PLS model. Using the idea of adaptive neural network-based fuzzy inference systems to develop a relationship of the inner model, it optimizes the parameters in the premise and consequent part of the fuzzy inference system in an adaptive framework [48–50].

For reaping the advantages of both PLS and ANFIS, the outer model is kept as in a linear PLS and the ANFIS is applied to model the inner relation. The coupled ANFIS-PLS model combines the robust capacity of PLS to learn the relationship between input–output variables and the adaptive learning of ANFIS to deal with nonlinearity in internal relations. Details on ANFIS can be found in [51–53]. The schematic illustration of the ANFIS-PLS is given in Fig. 4.4. ANFIS is implemented to model the inner relation between each pair of the input and output LVs,

$$\mathbf{U} = f(\mathbf{T}, \Theta) + \mathbf{e}, \tag{4.8}$$

where Θ represents the parameters of the model, and $\mathbf{E}$ the residual vector. Indeed, after getting the LVs (i.e., $\mathbf{T}_i$ and $\mathbf{U}_i$) from the outer model, they will be used to train and estimate the ANFIS parameters.

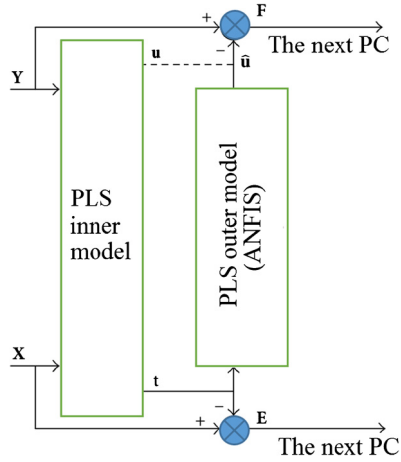


FIGURE 4.4 Conceptual illustration of ANFIS-PLS model.

https://t.me/learningnets

Importantly, ANFIS first uses a subtractive clustering procedure to find the initial rules and then applies five layers of the neural network model to tune the rule parameters [54]. The first layer in the ANFIS architecture focuses on the fuzzy formation by finding the membership functions for each input value. The second layer (called rule layer) generates fuzzy rules. The main role of the third layer is to normalize membership functions. The fourth layer is responsible for the conclusive part of fuzzy rules, and the last layer provides the final output. In the model training phase, the least-squares and back-propagation gradient descent algorithms are used to evaluate the parameters of the fuzzy inference system [48]. The network weights are trained by minimizing the residuals, $\mathbf{e}_i$, given as

$$J_i = \|\mathbf{e}_i\|^2 = \|\mathbf{U}_i - \mathbf{f}_i(\mathbf{T}_i, \Theta)\|^2. \tag{4.9}$$

Essentially, by keeping the outer relation unchanged, ANFIS-PLS benefits from the PLS characteristics by mapping the process variables in the directions that maximize the covariance. Hence, this allows transforming the multivariate regression problem to several univariate regressions, which are handled individually using ANFIS.

4.3.2.1 Nonlinear PLS-based monitoring

Primarily, we have to obtain the model first, then perform fault detection procedures accordingly. The basic point of fault detection using NLPLS approaches is similar to that of using linear PLS. Importantly, the estimation of the residual space, crucial in fault detection, depends on modeling with an appropriate process. Generally, the latent space is monitored by using the Hotelling's T^2, whereas the residual space is inspected via the squared prediction error (SPE) or Q charts [55] (Fig. 4.5). The T^2 statistic is computed as

$$T^2 = \sum_{i=1}^{m} \left(\frac{t_i}{\lambda_i}\right)^2, \tag{4.10}$$

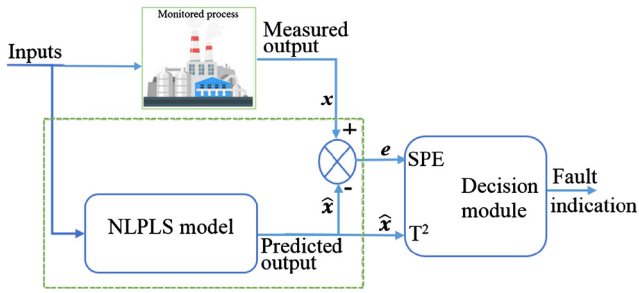


FIGURE 4.5 NLPLS-based fault detection methodology.

https://t.me/learningnets

where m is the number of factors retained in the NLPLS model, and λ_i is the ith eigenvalue of the covariance matrix Σ of the input data matrix $\mathbf{X}$. The T^2 flags out the presence of a possible anomaly to indicate that the performance of the process under monitoring is not running as designed if $T^2 > T_\alpha^2$, where T_α^2 is a detection threshold defined in [55].

In ANFIS-PLS, the outer model is similar to polynomial PLS, but instead of using polynomial functions, a more flexible nonlinear model based on ANFIS is used to learn the nonlinearity between input–output LVs.

The SPE statistic quantifies the information in the data that is not considered by the model. It is defined as

$$\text{SPE} = \|\mathbf{e}\|^2 = \|\mathbf{x} - \hat{\mathbf{x}}\|^2, \tag{4.11}$$

where $\mathbf{e}$ is the residual, and $\hat{\mathbf{x}}$ is the prediction corresponding to the input data $\mathbf{x}$ using the designed NLPLS model. During the absence of fault, the value of the Q statistic will be below the detection threshold, Q_α at significance level α given in [55]. However, if the Q statistic exceeds the detection threshold, a fault is flagged out.

These fault-detection charts are developed with the hypothesis that the underlying variables are uncorrelated and Gaussian. It is worth noting that the NLPLS model is efficient to monitor nonlinear processes. But using it with detection thresholds based on the Gaussian distribution can degrade its effectiveness. Numerous extensions of NLPLS-based monitoring have been devised, recently, in order to extend its detection capabilities. In [44], nonlinear PLS monitoring based on ANFIS is proposed, and the kernel density estimation (KDE) is applied to estimate the detection thresholds. This makes it possible to not use the theoretical assumptions of T^2 and SPE detection thresholds and handling non-Gaussian data. In [56], an assumption-free ANFIS-PLS method is developed for fault detection in multivariate input–output processes. To escape assumptions of ANFIS-PLS's T^2 and Q, a novel fault indicator with a k-nearest neighbor (kNN) and exponential smoothing scheme is developed. The exponentially smoothed kNN scheme is applied to residuals for the ANFIS-PLS model for detecting anomalies. Since kNN is able to separate anomalies from relevant features without making structural hypotheses on data yields; it is an elegant and flexible fault indicator, compared with traditional monitoring techniques (T^2 and Q). The key reason for choosing kNN, instead of conventional monitoring techniques, is that kNN overcomes the Gaussianity and absence of correlation assumptions that are the backbone of methods such as T^2 and Q [57]. In [58], an approach merging polynomial PLS and Hellinger distance (HD)-based schemes has been suggested to improve the sensitivity of NLPLS-based methods to small faults in nonlinear processes. Here, HD was used because this metric is useful for assessing the deviation between two distributions. Indeed, the capacity for detecting small changes could be improved by applying CUSUM or EWMA to the statistics T^2 and Q, after appropriately selecting the parameters of CUSUM and EWMA.

https://t.me/learningnets

4.3.3 Kernel PCA

Here, we introduce one of the well-known nonlinear input-space models, KPCA, which we use in this work. First, the basic idea behind KPCA and how it is used to model multivariate nonlinear processes is introduced. Lastly, the commonly used conventional monitoring techniques (i.e., SPE and T^2) with the KPCA model for anomaly detection are presented, along with a recently KPCA-based one-class SVM for process monitoring.

4.3.4 Kernel principal components analysis (KPCA) model

As discussed above, most environmental and engineering processes possess nonlinear features, and linear PCAs are not effective in extracting important information in the nonlinear processes.

To overcome this problem, nonlinear PCA methods, such as KPCA, provide the possibility to learn process nonlinearity in the data [13,30,29]. Generally speaking, the KPCA method stands out from the other existing nonlinear PCA methods in the literature due to its flexibility and simplicity [13]. KPCA can be viewed as a reformulation of the linear PCA in a high-dimensional space via kernel function to reveal nonlinear correlations among variables. It has been primarily used in anomaly detection for avoiding the complexity generated by nonlinear optimization in nonlinear PCA algorithms with neural networks [30]. The core concept of KPCA consists of projecting the data in the input space to a high-dimensional feature space (the kernel space) where data becomes more linear. This transformation is performed using a kernel function. Then, the linear PCA algorithm can be performed in the feature space, and the costly nonlinear optimization is avoided. The conceptual schematic of the idea behind KPCA is sketched in Fig. 4.6. Despite the fact that the KPCA learns linear features in the kernel space, these features are related to nonlinear features in the original space (before transformation) as displayed in Fig. 4.6. The original data can be reconstructed by projecting the features in kernel space onto a low-dimensional space, spanned by the eigenvectors that caught most of the variability. KPCA

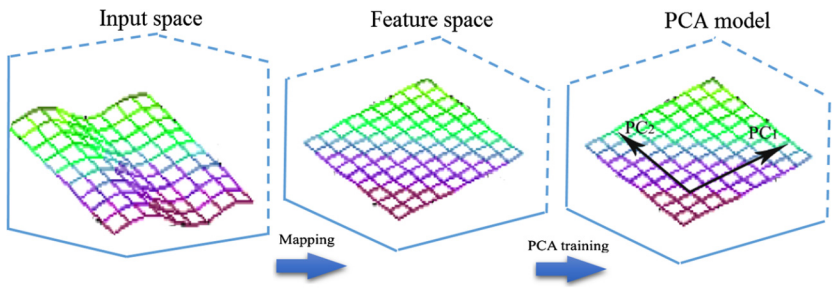


FIGURE 4.6 Basic concept of KPCA mapping.

https://t.me/learningnets

can effectively compute PCs in high-dimensional feature spaces based on integral operators and kernel functions [11].

In summary, KPCA provides an interesting formalism for learning process nonlinearity in comparison with other nonlinear methods based on neural networks. Specifically, KPCA has the ability to learn nonlinear features from the data without involving a nonlinear optimization. Also, the computation procedures from linear PCA can be directly inherited in kernel PCA [13].

The major benefit of KPCA, in comparison with its linear counterpart, is shown in Fig. 4.7. It can be seen that the observations cannot be linearly separated in the original space (Fig. 4.7A). However, the data becomes separable after the extraction of the nonlinear features, two groups of data are clearly distinguished (Fig. 4.7B).

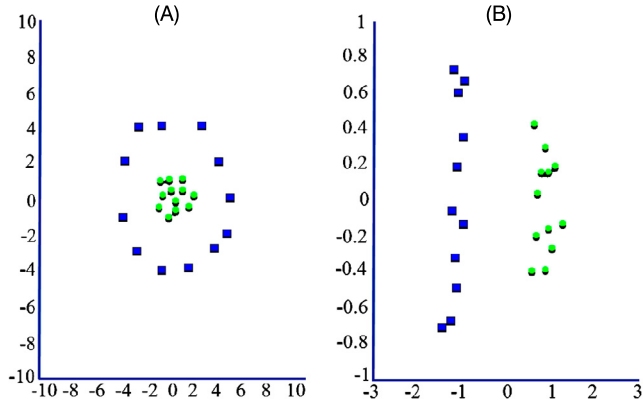


FIGURE 4.7 Conceptional representation of (A) PCA and (B) nonlinear PCA.

It should be noted that KPCA can be performed without straightforward knowledge of the mapping function Φ or the feature space F. As an alternative, computation is done on the inner product of pairs of points that are saved in a kernel matrix. This procedure is called "the kernel trick", which is a core part of the KPCA approach.

As indicated before, KPCA provides the possibility of transforming nonlinear features into linear separable features from the input space to feature space $\mathfrak{F}$ via kernel mapping and then apply the conventional PCA in the feature space. Let us consider the training dataset $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \in \mathbb{R}^m$, where m is the dimension of the variable and n is the number of available measurements. The original data is mapped from the input space to features space via a nonlinear-mapping function, $\Phi(\cdot)$, defined as

$$\mathbb{R}^m \xrightarrow{\Phi(\cdot)} \mathbb{F}^h, \tag{4.12}$$

where h is the dimension of the feature space. Thereby, $\Phi(\mathbf{x}_i)$ is the image of the observation $\mathbf{x}_i \in \mathbb{R}^m$ in the feature space.

https://t.me/learningnets

In a similar manner to the conventional PCA, the sample covariance matrix (with its elements are the inner product of all pairs of points $\Phi(\mathbf{x}_i)$ and $\Phi(\mathbf{x}_j)$) in feature space, can be calculated as

$$\mathbf{\Sigma}_{\mathbb{F}} = \frac{1}{n}\sum_{i=1}^{n}\left[\Phi(\mathbf{x}_i) - \mathbf{m}_{\Phi}\right]\left[\Phi(\mathbf{x}_i) - \mathbf{m}_{\Phi}\right]^T, \tag{4.13}$$

where $\mathbf{m}_{\Phi} = \sum_{i=1}^{n}\Phi(\mathbf{x}_i)/n$ is the sample mean in $\mathfrak{F}$. Let us consider $\overline{\Phi}(\mathbf{x}_i) = \Phi(\mathbf{x}_i) - \mathbf{m}_{\Phi}$ as the centered nonlinear mapping.

The principal components are found by solving the eigenvalue problem in $\mathfrak{F}$ as

$$\lambda\mathbf{v} = \mathbf{\Sigma}_{\mathbb{F}}\mathbf{v} = \frac{1}{n}\sum_{i=1}^{n}\left[\overline{\Phi}(\mathbf{x}_i)^T\mathbf{v}\right]\overline{\Phi}(\mathbf{x}_i), \tag{4.14}$$

where λ is an eigenvalue, $\lambda > 0$, and $\mathbf{v}$ is the corresponding eigenvector of the covariance matrix $\mathbf{\Sigma}_{\mathbb{F}}$. The first PC in $\mathfrak{F}$, which shows the largest variability in the data, is given along the direction of $\mathbf{v}$ with the greatest eigenvalue, and the PC indicates the direction of $\mathbf{v}$ with the lowest eigenvalue.

For nonzero λ, $\mathbf{v}$ could be expressed as a linear combination of α_i, $i = 1, \ldots, n$, thus

$$\mathbf{v} = \sum_{i=1}^{n}\alpha_i\overline{\Phi}(\mathbf{x}_i). \tag{4.15}$$

Multiplying by $\Phi(\mathbf{x}_j)$ on both sides of Eq. (4.14) gives

$$\lambda(\Phi(\mathbf{x}_j)\mathbf{v}) = \Phi(\mathbf{x}_j)\mathbf{\Sigma}_{\mathbb{F}}\mathbf{v}. \tag{4.16}$$

Now, expanding Eq. (4.14), we get

$$\lambda\sum_{i=1}^{n}\alpha_i\overline{\Phi}(\mathbf{x}_i)\overline{\Phi}(\mathbf{x}_j) = \frac{1}{n}\sum_{i=1}^{n}\alpha_i\left(\overline{\Phi}(\mathbf{x}_i)\sum_{i=1}^{n}\alpha_i\overline{\Phi}(\mathbf{x}_j)\right)\left(\overline{\Phi}(\mathbf{x}_i)\overline{\Phi}(\mathbf{x}_j)\right). \tag{4.17}$$

It should be noted that the eigenvalue problem in (4.14) can be solved using only scalar products of the mapped vector in $\mathcal{F}$, and that it can be handled based on a kernel matrix. Therefore, the use of a kernel function can simplify the computational challenges in the higher-dimensional space $\mathcal{F}$ to computing scalar products of vectors in $\mathcal{F}$, which highlights the great power of kernel tricks [59].

The kernel function allows determining in an implicit manner the nonlinear mapping in $\mathfrak{F}$. The kernel matrix $\mathbf{K} \in \mathbb{R}^{n\times n}$ is expressed as

$$\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i)^T\Phi(\mathbf{x}_j). \tag{4.18}$$

Then, from Eq. (4.17), we will get

$$n\lambda\mathbf{K}\boldsymbol{\alpha} = \mathbf{K}^2\boldsymbol{\alpha}, \tag{4.19}$$

https://t.me/learningnets

where $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_n]$. Mean-centering in the feature space is essential before applying KPCA, which is done as follows:

$$\overline{\mathbf{K}}(\mathbf{x}_i, \mathbf{x}_j) = \overline{\Phi}(\mathbf{x}_i)^T \overline{\Phi}(\mathbf{x}_j), \tag{4.20}$$

$$\overline{\mathbf{K}} = \mathbf{K} - \mathbf{K}\mathbf{1}_n - \mathbf{1}_n\mathbf{K} + \mathbf{1}_n\mathbf{K}\mathbf{1}_n, \tag{4.21}$$

where $\mathbf{1}_n$ is the following matrix:

$$\mathbf{1}_n = \frac{1}{n}\begin{pmatrix} 1 & \ldots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \ldots & 1 \end{pmatrix} \in \mathbb{R}^{n \times n}. \tag{4.22}$$

To solve the eigenvalue decomposition problem in Eq. (4.19), KPCA now solves the reformed eigenvalue problem of the centered kernel as [29]

$$n\lambda\boldsymbol{\alpha} = \overline{\mathbf{K}}\boldsymbol{\alpha} \tag{4.23}$$

for nonzero eigenvalues ($\lambda \neq 0$). This is equivalent to applying the conventional PCA in feature space, F. Thereby, performing PCA in F allows obtaining eigenvectors $\boldsymbol{\alpha_1}, \ldots, \boldsymbol{\alpha_n}$, with $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$ as the corresponding eigenvalues.

At last, we can represent the output as

$$y_j = \sum_{i=1}^{n} \boldsymbol{\alpha}_{ij}^k \overline{\mathbf{K}}(x_j, x), \quad j = 1, \ldots, d. \tag{4.24}$$

Generally speaking, the performance of the KPCA-based method mainly depends on the selected kernel functions. Numerous kernels are designed in the literature [60], but, to the best of our knowledge, there is no automated way for kernel selection. In the existing techniques, the selection of a kernel function is usually done empirically or experimentally from an ensemble of candidates. Indeed, inappropriate performance could be achieved in the case of a poor kernel choice [61,62]. Here are some of the most widely used kernel functions in the machine learning literature:

Linear kernel,

$$\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) = \langle \mathbf{x}_i, \mathbf{x}_j \rangle. \tag{4.25}$$

Polynomial kernels,

$$\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) = \langle \mathbf{x}_i, \mathbf{x}_j \rangle^d, \quad d \in \mathbb{Z}^+. \tag{4.26}$$

Cosine kernels,

$$\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) = \frac{\langle \mathbf{x}_i, \mathbf{x}_j \rangle}{\|\mathbf{x}_i\| \, \|\mathbf{x}_j\|}. \tag{4.27}$$

https://t.me/learningnets

Sigmoid kernels,

$$\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\beta_0 \langle \mathbf{x}_i, \mathbf{x}_j \rangle + \beta_1). \tag{4.28}$$

Radial basis functions (RBF),

$$\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) = e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\delta^2}} = e^{-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2}, \quad \gamma \in \mathbb{R}^+, \tag{4.29}$$

where $2\delta^2 = \frac{1}{\gamma}$ is the width of the Gaussian kernel, while d, β_0, and β_1 are specified a priori before using the above kernel functions. It should be noted that similar results are obtained by the above kernels if the values of parameters are appropriately selected. The RBF, which is frequently used in fault-detection, offers more flexibility in selecting the associated parameter; it is also assumption-free with respect to relationships among the process variables.

4.3.5 KPCA-based fault detection procedures

KPCA monitoring is performed in two broad phases: model development and anomaly detection. The decision statistics (e.g., Hotelling's T^2 and Q) and their detection thresholds are computed in the offline modeling phase. In the anomaly detection phase, the monitoring statistics are calculated based on the new testing measurements and compared with the previously established decision thresholds to verify the status of the process. An anomaly is flagged if a test statistic is above its detection threshold.

For anomaly-detection purposes, the T^2 and SPE (squared prediction error) monitoring schemes can be computed by testing online measurements. To monitor the variation in the KPCA model, T^2 scheme, which uses normalized squared scores, is employed. Its statistic is computed as

$$T_f^2 = \left[t_1, \ldots, t_p\right] \Lambda^{-1} \left[t_1, \ldots, t_p\right]^T, \tag{4.30}$$

where t_k $(k = 1, 2, \ldots, p)$ are the retained p PCs, and Λ^{-1} is the inverse of the matrix of eigenvalues corresponding to the retained PCs.

The detection threshold for T^2 is computed from the F distribution as

$$T_\alpha^2 = \frac{p(m-1)}{(m-p)} F_{p,m-p,\alpha}, \tag{4.31}$$

where $F_{p,m-p,\alpha}$ is an F-distribution with p and $m - p$ degrees of freedom and having a significance level of α.

The variations in the smallest $m - p$ PCs can be monitored using the Q statistic. Indeed, the Q statistic is relevant to indicate the variability not described by the maintained PCs in the KPCA model:

$$Q_f = \left\| \Phi(\mathbf{x}) - \Phi_p(\mathbf{x}) \right\|^2 = \sum_{j=1}^{m} t_j^2 - \sum_{j=1}^{p} t_j^2. \tag{4.32}$$

https://t.me/learningnets

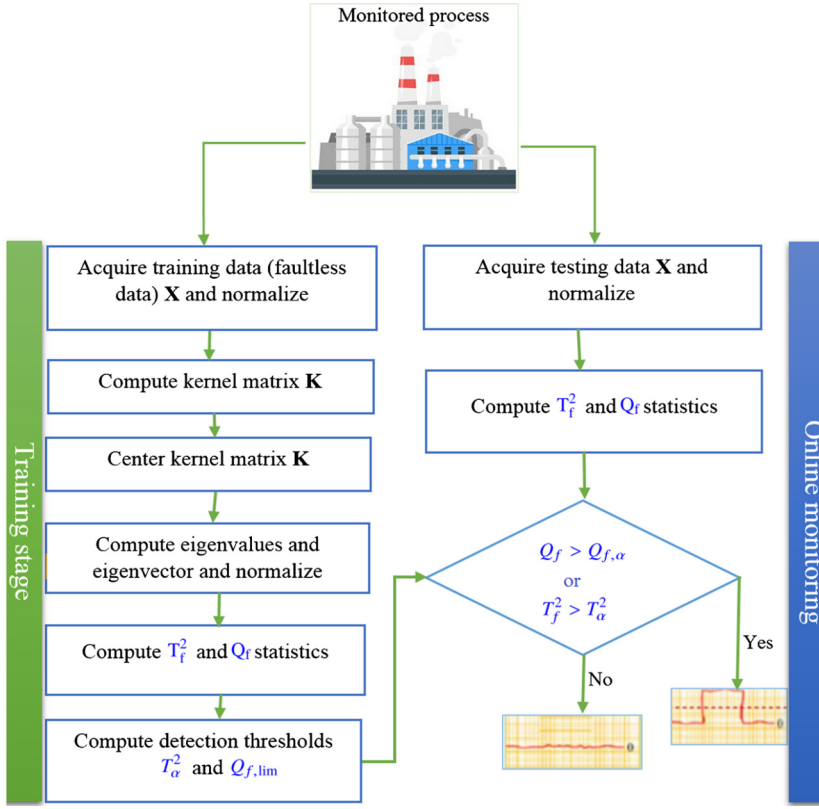


FIGURE 4.8 KPCA monitoring strategy.

The Q_f detection threshold is computed as

$$Q_{f,\lim} = \theta_1 \left[c_\alpha \sqrt{2\theta_2 h_0^2}/\theta_1 + 1 + \theta_2 h_0 (h_0 - 1)/\theta_1^2 \right]^{1/h0}, \tag{4.33}$$

where $\theta_j = \sum_{p+1}^{m} (\lambda_{ii})^j$ $(j = 1, 2, 3)$, $h_0 = 1 - 2\theta_1\theta_3/3\theta_2^2$, λ_i are the eigenvalues, and c_α represents the $100(1-\alpha)$th normal percentile.

The basic framework of the KPCA fault-detection strategy is performed in two main stages, offline model building and online fault detection (Fig. 4.8) (for further details, see [63,64]):

- In offline learning, the initial step is to autoscale the training dataset (faultless) by subtracting the mean and dividing by the standard deviation for each variable (i.e., the column of the input data matrix). As discussed above, KPCA extracts linear and nonlinear structures by mapping data into high-dimensional spaces using kernel tricks. Then, we select one kernel function $\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j)$, and compute the scaled kernel matrix of the data, $\overline{\mathbf{K}}$. We subse-

https://t.me/learningnets

quently determine the eigenvalues and eigenvectors by solving the eigenproblem $\overline{\mathbf{K}}\boldsymbol{\alpha} = n\lambda\boldsymbol{\alpha}$ and select the number of PCs to retain. We can represent the outputs as $\mathbf{y}_j = \sum_{i=1}^{d} \alpha_{ij}\overline{\mathbf{k}}(\mathbf{x}_i, \mathbf{x})$, for $j = 1, \ldots, m$. Lastly, we compute the detection thresholds for PCA's T^2 and Q charts.

- In the online monitoring stage, at first new arrival datasets are scaled with the mean and variance of the training dataset (faultless). Then, the KPCA's monitoring statistics T^2 and Q are computed for anomaly-detection purposes. Finally, the monitoring statistics are compared with the detection thresholds computed in the training stage. If the monitoring statistics are above the detection thresholds, an anomaly is flagged out; otherwise, the process is run under normal conditions.

It should be noted that the detection thresholds of the T^2 and Q statistics are computed based on the hypothesis that process variables are Gaussian-distributed as in the conventional PCA. However, this hypothesis is not often valid for datasets from several real industrial processes. Accordingly, adopting these detection thresholds for process-monitoring based on the multivariate Gaussian assumption is not suitable, and may give wrong results [65,66]. An alternative solution to escape the Gaussian assumption in computing detection thresholds of KPCA's T^2 and Q charts is to compute these thresholds nonparametrically using the kernel density estimation (KDE) [64]. In this approach, at first, the KDE is used to estimate the distributions underlying the KPCA's T^2 and Q statistics based on faultless datasets. Then, the decision threshold is set to the corresponding $(1-\alpha)$th quantile. Results in [64] claim that KPCA with nonparametric thresholds outperforms empirical PCA's Q and T^2 statistics based on the Gaussian distribution for effective fault detection in nonlinear processes. Recently in [67], an assumption-free KPCA fault-detection method was introduced for monitoring nonlinear processes. This method uses a support-vector machine (OCSVM) to evaluate the extracted features from KPCA. Of course, the KPCA-OCSVM approach escapes theoretical KPCA assumptions and offers better monitoring performance in nonlinear processes than thresholds based on the Gaussian hypothesis. Over the years, several improvements of KPCA were developed for meeting numerous needs in practical use. In [68], a recursive kernel PCA, which updates the model continuously online, has been applied in adaptive monitoring of nonlinear processes. In [69], a dynamic KPCA was proposed to identify both spatial and temporal relationships in the data matrix augmented by time-lagged variables. In [70], a moving window KPCA for adaptive monitoring of nonlinear processes was developed. In [71], methods combining the advantages of multiscale decomposition using wavelets with KPCA and KPLS models were developed for nonlinear process monitoring. These methods allow the extraction of the relevant information at different scales and take the multiscale nature of data. They provided satisfactory detection performances when applied to the continuous annealing process and fused magnesium furnace. Authors in [72] proposed multiscale KPCA based on sliding median filter (SFM) for monitoring nonlinear processes with noise and outliers. Results showed

https://t.me/learningnets

that MSKPCA-SFM had a superior fault-detection capacity, compared to the KPCA. This is mostly due to the capacity of SFM in removing disturbances and noises and MSKPCA's ability to provide nonlinear dynamic modeling, unlike MSPCA. In other applications, KPCA is amalgamated with SVM, ICA, recursive-weighted PCA, and EWMA, to cite a few, for enhancing monitoring multivariate nonlinear processes [73–77].

4.4 Cases study: monitoring WWTP

The aim of this case study is to show the detection capabilities of an assumption-free approach based on the KPCA model and the OCSVM detector. Here, the amalgamated KPCA-OCSVM is applied to monitor influent measurements from an actual wastewater treatment plant located in Saudi Arabia. Also, we compared the KPCA-OCSVM methods with conventional PCA-based methods to detect anomalies in nonlinear multivariate time series data. The influent measurements contain seven years of the daily dataset of 21 variables that include different flow quantities and water quality values. Measurements were carried out on samples collected from the headwork of WWTP to keep conformity with local standards. The supervised WWTP was treated municipal wastewater. Even with extended efforts and duly inspection by the local technicians, over a hundred anomalous influent conditions were missed. These undetected anomalies have led to negative impacts on the process due to different reasons causes as claimed by the practitioners, which highlights the greater necessity of research on statistical process monitoring.

To illustrate the interactions between the collected variables, a hierarchical clustering heatmap with a density plot using quarterly averages is displayed in Fig. 4.9. In this figure, Z scores are summarized as probability density distribution using KDE, as illustrated in the density plot. Even with the use of quarterly averages for smoothing and visualization, asymmetric unimodal distribution having positive skewness and positive kurtosis is clearly visible. It can be concluded that this dataset is non-Gaussian distributed. This heatmap was reordered by the hierarchical clustering and the process variables are reordered in rows. Indeed, palettes having a similar composition in rows imply positive correlations, whereas the reverse for negative correlations. The total inflow is largely impacted by the inflow from the lift station. The inflow from lift station eight "InFlow_LS8", which is linked to a desalination plant, is negatively correlated to Cl, COD, BOD, and alkalinity. This shows the presence of various compositions of industrial discharges and municipal wastewater in this situation. It can be seen that the inflow from WWTP inside is related to hardness and TSS, which could be a result from the sludge-processing events and the regular cleaning of the membrane with chemicals dosed. On the other hand, the temperature is negatively correlated to all major water quality variables, but positively correlated to almost all municipal water quantity variables.

Overall, data generated from WWTP are statistically challenging because they are high dimensional, relationships among variables are nonlinear, vari-

https://t.me/learningnets

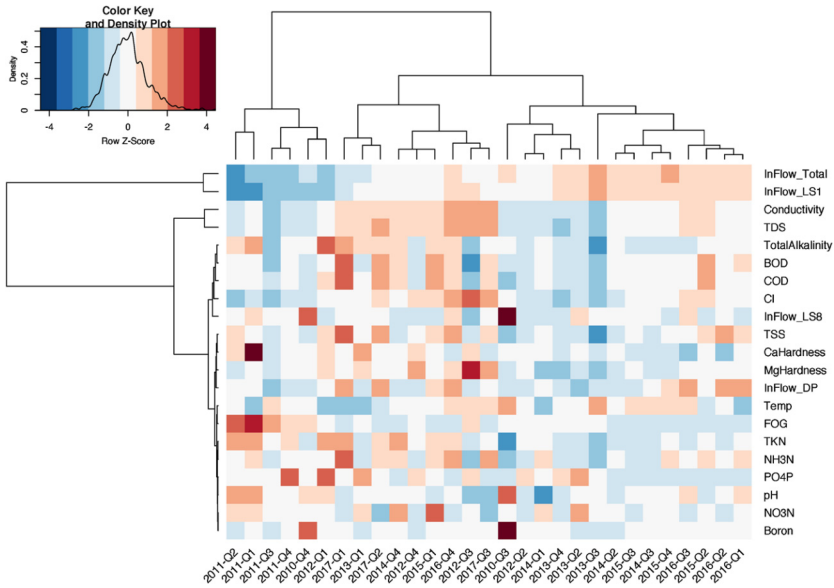


FIGURE 4.9 Heatmap of WWTP testing data with density plot.

ables are non-Gaussian and nonstationary, and a variety of fault types can occur. These data would challenge conventional modeling methods that are designed based on hypotheses such as linear relationships between variables, absence of autocorrelation, and Gaussian distributions. Thus, to handle the nonlinearity and non-Gaussian behavior of the WWTP process, nonlinear and flexible methods such as kernel techniques could be promising.

4.4.1 Anomaly detection using KPCA-OCSVM method

To evaluate the performance of the proposed monitoring schemes, true-positive rate (TPR, or recall), false-positive rate (FPR), area under the receiver operating characteristic curve (AUC), accuracy, precision, and F1-score are computed. Fig. 4.10 summarizes these frequently used metrics in the evaluation of detection quality. The performance of KPCA-OCSVM methods is summarized in Fig. 4.11. Previous research on PCA-based methods based on the same WWTP datasets, which are summarized in Fig. 4.12, is used as a benchmark for comparison [78]. In PCA-based methods provided in Fig. 4.12, the reference PCA model is constructed first, using seven PCs that capture over 80% of the variance in the data. Then, nine of PCA's anomaly-detection schemes are derived, including univariate residuals, squared prediction error (SPE), T^2, and k-nearest neighbor distances (Euclidean or Manhattan), for which parametric/nonparametric thresholds were set to detect anomalies.

https://t.me/learningnets

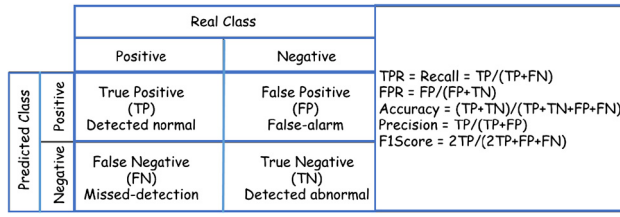


FIGURE 4.10 Commonly used metrics for evaluating detection performance.

Fig. 4.11 shows the detection performance of the KPCA-OCSVM, using different kernels (polynomial, cosine, and RBF) and increasing numbers of PCs. The columns in the heatmaps represent the evaluation metrics and the rows represent KPCA-OCSVM with umbers of PCs. In general, to design a reference KPCA model using faultless datasets describing the nominal behavior of the process, accurate learning and approximating are achieved when using relatively more PCs. However, excessive PCs also give overfitted solutions and lead to degradation in the detection performance for later OCSVM processing. Thus, TPR and FPR are usually increased with a larger number of PCs, while precision would drop. The better detection performance of AUC, as a weighted combination of TPR and FPR, and F1-score as the harmonic mean of TPR and precision, are achieved when tuning. Hence, the perfect detections are realized with polynomial KPCA when using around 10 PCs, indicating its greater capacity in learning relevant information in the training dataset. Moreover, the performance of KPCA with a cosine kernel based on 15 PCs was comparable to that of polynomial KPCA in this study. In contrast, RBF KPCAs barely capture the anomalies when using fewer PCs but sufficiently recognize anomalies when using large counts of PCs in this study, which needs heavy computations or risks potential overfitting. Thus, in this study, RBF KPCAs is not suited for real-time monitoring.

The results in Figs. 4.11 and 4.12 indicate that kernel methods capture the nonlinear relationship in the process variables better than using conventional PCA-based methods. Also, it is interesting to notice that whereas applying KNN lazy learning procedure in the PCA-KNN approach to address the nonlinearity problem outperformed other counterparts, extracted features from PCA are not adequately pertinent information in data. This degrades the detection capabilities of PCA's methods in this case mainly due to the process nonlinearity. In contrast, dealing with nonlinearity could be performed better with polynomial KPCA and summarized by RBF-based OCSVM than linear PCA, but not highly nonlinear, compared to outcomes that are obtained when using RBF in KPCA.

In this case study, it has been demonstrated that KPCA-OCSVM methods to nonlinear multivariate process monitoring of a WTTP compare favorably with existing linear PCA-KNN methods. The KPCA-OCSVM models could appropriately extract relevant information in data and reveal linear and nonlinear relationships among the process variables. Then, OCSVM is applied to

https://t.me/learningnets

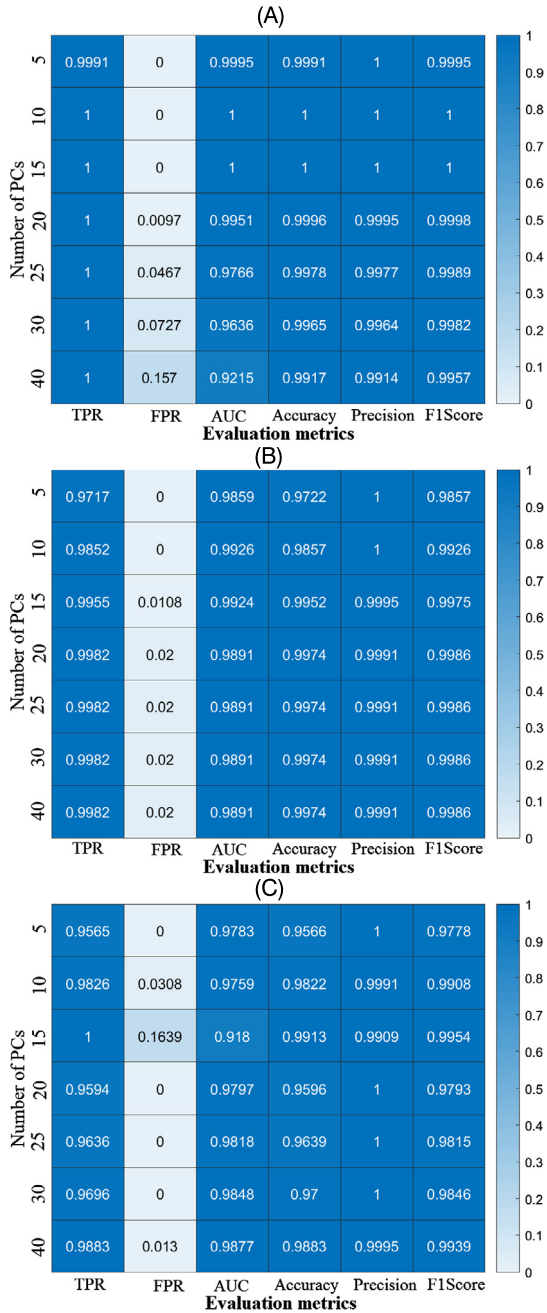


FIGURE 4.11 Heatmap depicting the detection performance of KPCA-OCSVM schemes by kernel: (A) Results of KPCA-OCSVM scheme with polynomial kernel; (B) Results of KPCA-OCSVM scheme with cosine kernel; (C) Results of KPCA-OCSVM scheme with RBF kernel.

https://t.me/learningnets

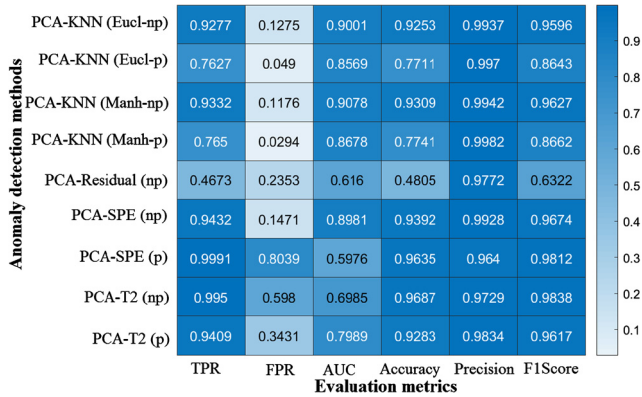


FIGURE 4.12 Heatmap showing the detection performance of the nine PCA-based monitoring algorithms (in rows) and evaluation metrics (in columns).

KPCA's extracted features to uncover anomalies. The obtained results on real data show the efficiency of the kernel techniques in monitoring WWTPs. Furthermore, the major advantage of using this approach is its flexibility to model a wide range of nonlinearities in the data by using kernel functions, and without involving any nonlinear optimization. As an assumption-free approach, the KPCA-based approach could be transferred, rebuilt, and adjusted for ICs from different WWTPs.

4.5 Simulated synthetic data

The aim of this example is to show the prediction performance of nonlinear PLS models, in comparison with linear PLS. Three PLS algorithms, including linear PLS, polynomial PLS, and ANFIS-PLS, were compared in terms of data fitting and prediction. Simulated nonlinear datasets with one response and ten input (predictor) variables were generated. The first two inputs were simulated using "randn" signals and the remaining eight inputs were computed as:

$$\mathbf{x}_3 = \mathbf{x}_1 + \mathbf{x}_2, \tag{4.34}$$

$$\mathbf{x}_4 = \mathbf{x}_3^2, \tag{4.35}$$

$$\mathbf{x}_5 = \sqrt{\mathbf{x}_3}, \tag{4.36}$$

$$\mathbf{x}_6 = \frac{1}{\sqrt{\mathbf{x}_3}}, \tag{4.37}$$

$$\mathbf{x}_7 = 1 - e^{-\mathbf{x}_3}, \tag{4.38}$$

$$\mathbf{x}_8 = 1.4\mathbf{x}_2 - 1.2\mathbf{x}_7, \quad \mathbf{x}_9 = 1.3\mathbf{x}_2 + 2.1\mathbf{x}_1, \tag{4.39}$$

$$\mathbf{x}_{10} = 1.3\mathbf{x}_6 - 2.3\mathbf{x}_9. \tag{4.40}$$

https://t.me/learningnets

The output was generated based on inputs as follows:

$$\mathbf{y} = \sum_{i=1}^{10} b_i \mathbf{x}_i, \tag{4.41}$$

where $b_i = \{0.07, 0.03, -0.05, 0.04, 0.02, -1.1, -0.04, -0.02, 0.01, -0.03\}$, for $i = 1, \ldots, 10$. A noise-free dataset with 256 samples was generated and then contaminated with zero-mean Gaussian noise.

The performance of the NL-PLS was compared with that of the conventional PLS model. The cross-validation method was used for selecting the optimum number of latent variables. The total data were split into two parts, a training dataset and a testing dataset. The training data was used to develop the model, the obtained model was tested with a testing dataset, and its mean squared error (MSE) was evaluated. The optimum number of the latent variable was selected by choosing the one which gives the minimum MSE. Five latent variables were selected for PLS, three for poly-PLS and ANFIS-PLS models based on the minimum MSE values. Table 4.3 summarizes the results of the comparison for different signal-to-noise ratio (SNR) and clearly indicates that poly-PLS and ANFIS-PLS modeling provides a significant improvement over the conventional PLS modeling techniques. This is because the nonlinear PLS algorithm uses the appropriate mapping function in the inner relation of the PLS algorithm, thus it avoids overfitting the model and improves the model prediction ability. Additionally, results showed the strongest predictive capability of ANFIS-PLS among all the models. In the following section, one more example is considered.

TABLE 4.3 Monte Carlo MSEs for the three models.

Model	SNR = 5	SNR = 10	SNR = 20
PLS	0.2189	0.1209	0.0641
Poly-PLS	0.2019	0.1019	0.049
ANFIS-PLS	0.1739	0.08781	0.047

4.5.1 Application of plug flow reactor

Here, the detection performance of NLPLS-based monitoring methods is assessed using simulated plug flow reactor (PFR) data. PFRs, or piston flow reactors, consist of a hollow pipe in which reactants travel [79]. Fig. 4.13 displays an example of a schematic representation of a PFR having tubular form wrapped with an acrylic mold that is encased in a tank. To keep a relatively stable reactant temperature, water at a controlled temperature is passed via the tank. PFRs are widely used in either gas or liquid-phase systems. Two first-order reactions are performed in the reactor, based on the reactant species, A:

$$A \rightarrow B \rightarrow C \tag{4.42}$$

https://t.me/learningnets

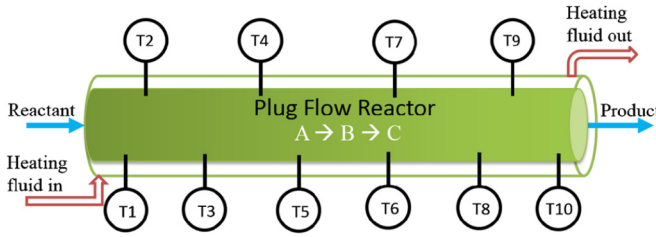


FIGURE 4.13 Schematic representation of a plug flow reactor.

where B is the intended product and C is an undesired product. As the reaction is endothermic, the reactor is heated by the fluid in the jacket of the reactor and its flow rate is controlled to obtain the needed outlet concentration of product B. The set of model PDE equations (4.43), (4.44), (4.45), and (4.46) is used to model this process:

$$\frac{\partial C_A}{\partial t} = -v_l \frac{\partial C_A}{\partial x} - k_{10} e^{-E_1/RT_r} C_A, \tag{4.43}$$

$$\frac{\partial C_B}{\partial t} = -v_l \frac{\partial C_B}{\partial x} + k_{10} e^{-E_1/RT_r} C_A - k_{20} e^{-E_2/RT_r} C_B, \tag{4.44}$$

$$\begin{aligned}\frac{\partial T_r}{\partial t} = &-v_l \frac{\partial T_r}{\partial x} + \frac{\Delta H_{r_1}}{\rho_m c_{pm}} k_{10} e^{-E_1/RT_r} C_A \\ &+ \frac{\Delta H_{r_2}}{\rho_m c_{pm}} k_{20} e^{-E_2/RT_r} C_B + \frac{U_w}{\rho_m c_{pm} V_r} (T_j - T_r),\end{aligned} \tag{4.45}$$

$$\frac{\partial T_j}{\partial t} = -u \frac{\partial T_j}{\partial x} + \frac{U_{w_j}}{\rho_{mj} c_{pmj} V_j} (T_r - T_j). \tag{4.46}$$

Next, the prediction ability of nonlinear PLS is used to predict product concentration.

4.5.1.1 *Data generation and modeling*

In this study, PFR process data is first simulated. Then, the performance of the nonlinear PLS-based monitoring schemes is assessed through their application to detect faults in simulated PFR data (see Table 4.4). To generate FPR data, the flow rate of the feed stream is perturbed from the nominal steady-state ranges based on pseudorandom binary signals (PRBS) in the frequency interval of $[0, 0.05w_N]$, where $w_N = \pi/T$ denotes the Nyquist frequency. To do so, we use the function "idinput" of the system identification toolbox of Matlab®. Here, there are 11 input variables, ten temperatures measured at various locations of the reactor, and the feed flow. The output variables are the concentration of the product (i.e., C_B). The PFR input–output perturbation data are presented

https://t.me/learningnets

TABLE 4.4 Plug flow reactor: model parameters.

Process variable	Value	Process variable	Value
v_l	1 ml/min	C_{pm}	0.231 kcal/(kg K)
L	1.0 m	R	1.987 kcal/(min K)
V_r	10.0 lt	ρ_m	0.09 kg/lt
E_1	20000 kcal/kmol	U_w	0.20 kcal/(min K)
E_1	50000 kcal/kmol	C_{pmj}	0.8 kcal/(min K)
k_{10}	5.0×10^{12} min^{-1}	V_j	366 K
k_{10}	5.0×10^{2} min^{-1}	ρ_{mj}	0.10 kg/lt
H_{r_1}	0.5480 kcal/kmol	C_{A0}	4 mol/lt
H_{r_2}	0.9860 kcal/kmol	C_{B0}	0 mol/lt
T_{r0}	320 K	T_{j0}	375 K

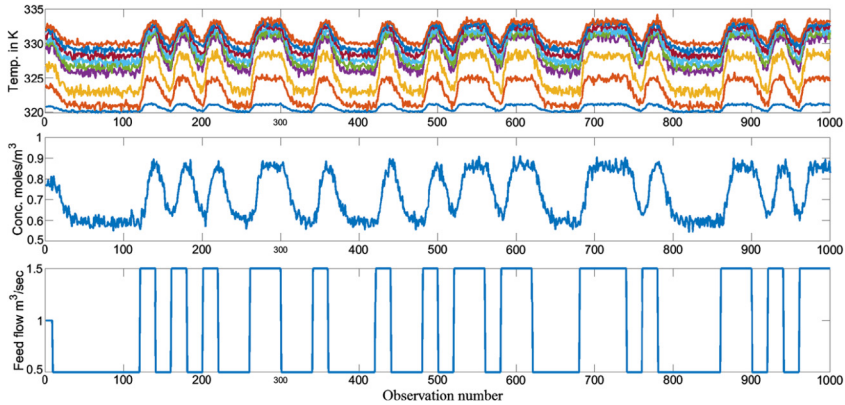


FIGURE 4.14 PFR input–output data.

in Fig. 4.14. These data, which are noise-free, are then contaminated with zero-mean Gaussian noise.

A total of 500 samples of fault-free data generated using the PFR model described above were used to construct the NLPLS model. To construct a reference NLPLS model, the cross-validation method was applied to find the optimal number of PCs and 4 resulting PCs were retained for the NLPLS model. Fig. 4.15 illustrates the plot of the observed data versus the predicted values from the NLPLS. Fig. 4.16 shows the scatter graph of the observed and predicted values obtained by the selected NLPLS model. It can be seen that the selected NLPLS fits the training data well. Furthermore, this model achieved an R^2 of 0.94, and small root mean squared error (RMSE) and mean absolute error (MAE) (i.e., RMSE = 0.22618 and MAE = 0.17). This means that the NLPLS model with 4 PCs possesses good predictive capability.

https://t.me/learningnets

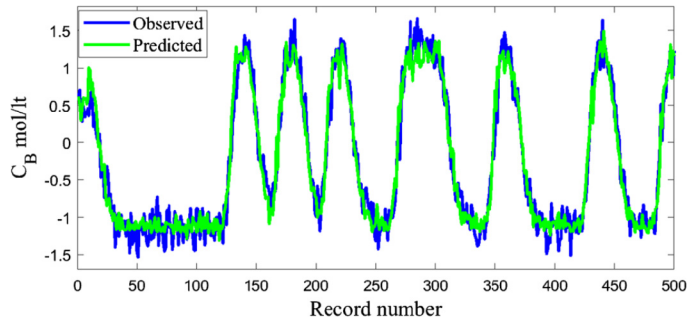


FIGURE 4.15 Plots of observed testing data and NLPLS predicted data.

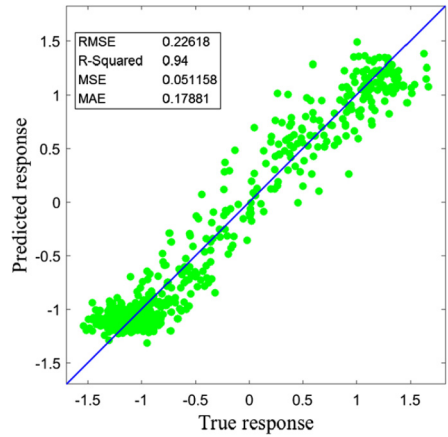


FIGURE 4.16 Scatter graph of test data and NLPLS estimated data.

4.5.1.2 Detection results

After identifying the reference NLPLS model, we used it to detect anomalies in the PFR process. Three monitoring techniques were considered here based on the identified NLPLS model: the conventional NLPLS-Q scheme and NLPLS-based Hellinger distance (HD) using unfiltered residuals and filtered residuals. In the NLPLS-HD scheme, the HD was used as an indicator of fault by computing the deviation between probability distributions of actual residuals and residuals obtained using fault-free data. Here, the 3-sigma rule was used to set up the detection threshold for NLPLS-HD. To improve the robustness of the NLPLS-HD measurement noise, we applied HD on the filtered residuals with a wavelet-based filter to reduce the noise in the data and improve the robustness to measurement noise. The residuals, which show a mismatch between the output of the reference NLPLS model and the measurements, are filtered via a wavelet-based filter and used as the input for the NLPLS-HD scheme.

https://t.me/learningnets

4.5.1.3 Case (A) – abrupt anomaly detection

The testing data set, which was simulated using the same PFR model, consisted of 500 data samples completely independent from the training data. Here, the feasibility of the three NLPLS-based schemes was verified to sense atypical abrupt changes (sensor bias). Three examples are presented to test the detection capability of NLPLS-based schemes to uncover abrupt faults in PFRs. To assess the abilities of the three fault detection methods, for data with SNR = 30, an additive fault was introduced in x_5 between samples 300 and 350, which consists of bias of amplitude equal to 10% of the total variation in x_5. In practice, this might be similar to a sudden sensor offset or miss-calibration. Table 4.5 provides a summary of detection performance by method. As the magnitude of this abrupt fault is relatively large and the level of SNR is also high (i.e., SNR = 30), the performance of the three methods is slightly comparable. We can see an improvement when applying the NLPLS-HD, compared to the conventional NLPLS-Q methods by achieving the highest AUC of 0.975.

TABLE 4.5 Detection performance by method (10% bias, SNR = 30).

Method	TPR	FPR	Accuracy	Precision	FIScore	AUC
NLPLS-Q	0.941	0.000	0.994	1.000	0.970	0.971
NLPLS-HD	0.980	0.031	0.970	0.781	0.870	0.975
NLPS-HD (filtered)	1.000	0.053	0.952	0.680	0.810	0.973

To assess the ability of the various fault detection methods under low SNR (i.e., SNR = 5). The detection performance of the three methods is given in Table 4.6. The conventional NLPLS-Q can recognize this fault but with several missed detection and moderate AUC (TPR = 0.706 and AUC = 0.853). This highlights the sensitivity of the Q statistic to noisy data. Here, the NLPLS-HD statistic with filtered residuals achieves a better performance over the other test statistics by achieving a higher AUC of 0.983 and F1-score of 0.872. An interesting observation is that the NLPLS-HD is more robust to noise compared to the NLPLS-Q. Furthermore, applying NLPLS-HD on filtered data using a wavelet filter, which is a powerful tool to reduce noise in data, significantly improves the detection performance under low SNR values, compared with other methods.

TABLE 4.6 Detection performance by method (10% bias, SNR = 5).

Method	TPR	FPR	Accuracy	Precision	FIScore	AUC
NLPLS-Q	0.706	0.000	0.970	1.000	0.828	0.853
NLPLS-HD	0.961	0.031	0.968	0.778	0.860	0.965
NLPS-HD (filtered)	1.000	0.033	0.970	0.773	0.872	0.983

https://t.me/learningnets

TABLE 4.7 Detection performance by method (intermittent fault, first example).

Method	TPR	FPR	Accuracy	Precision	FIScore	AUC
NLPLS-Q	1.000	0.069	0.947	0.811	0.896	0.966
NLPLS-HD	0.971	0.000	0.993	1.000	0.985	0.985
NLPS-HD (filtered)	0.981	0.032	0.971	0.902	0.940	0.974

4.5.1.4 Case (B) – intermittent anomaly detection

Here, we present the capacity of NLPLS-based Q and HD techniques in uncovering intermittent faults. A bias of amplitude 10% of the total variation in the temperature variable T_5 is added to the testing measurements for samples in [200, 250] and a bias of 15% is injected between [400, 450]. Table 4.7 provides a summary of detection performance for the three methods. Results showed that the NLPLS-HD had a better detection performance compared to the conventional NLPLS-HD. In the second example, biases of amplitude 3%, 4%, and 5% were injected respectively between samples intervals [200, 250], [300, 350], and [400, 450]. The NLPLS-Q is not able to detect these faults; it is less sensitive to small changes. Table 4.8 shows a summary of detection performance for the NLPLS-HD methods based on unfiltered residuals and filtered residuals using the wavelet denoising approach. As no detection is flagged by the NLPLS-Q approach, its results are not presented in the summary. Results in Table 4.8 highlight the capability of the NLPLS-HD in uncovering these intermittent faults. Furthermore, the NLPLS-HD based on filtered residuals exhibits better detection compared to the NLPLS-HD using raw residuals.

TABLE 4.8 Detection performance by method (intermittent faults, second example).

Method	TPR	FPR	Accuracy	Precision	FIScore	AUC
NLPLS-HD	0.708	0.030	0.880	0.924	0.801	0.839
NLPS-HD (filtered)	0.994	0.114	0.922	0.818	0.897	0.940

4.5.1.5 Case (B) – drift anomaly detection

Now, the capability of the NLPLS-based FD methods in monitoring a sensor drift fault is investigated. A gradual ramp fault with a slope of 0.01 (resembling an aging sensor fault) is injected into data of temperature variable T_5 at sample 300. Fig. 4.17A–C displays the monitoring results of the three NLPLS-based methods. Fig. 4.17 indicates that the NLPLS-based Q, HD, and HD using filtered residuals methods sense this drift fault respectively at sample 372, 328, and 309. The results clearly suggest that the NLPLS-HD scheme is capable to detect drift fault the earliest compared with the conventional NLPLS-HD scheme.

https://t.me/learningnets

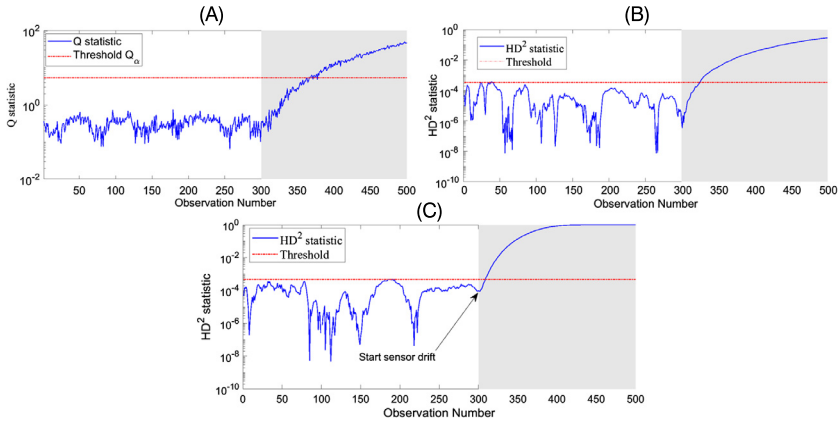


FIGURE 4.17 Results from (A) NLPLS-Q statistic, (B) NLPLS-HD, and (C) NLPLS-HD based on denoised residuals in the case of a drifting fault.

In summary, this case study demonstrates the capability of the NLPLS model to describe nonlinearly in the FPR process. Also, results show the superior detection efficiency of the NLPLS-HD compared to the conventional NLPLS-Q in identifying abrupt, intermittent, and progressive faults. This is mainly due to the sensitivity of the HD metric to sense the deviation between the distributions of residual of fault-free data and actual residuals generated by NLPLS. Applying NLPLS-HD to filtered residuals improves the detection capability, in particular when dealing with noisy measurements.

4.6 Discussion

To maintain the product quality, efficiency, and reliability of modern and complex processes, anomaly detection is becoming extremely important for process monitoring. Effective and early detection of process anomalies is vital for avoiding the progression of anomalies and reducing productivity loss; this can help avoid serious degradations or system crashes. Hence, anomaly detection is a hot topic attracting the interest of industrial practitioners and researchers.

Linear LVR methods, such as PLS and PCA, are popular tools for monitoring multivariate data with collinearity or redundancy among the variables. LVR techniques have been widely exploited in modeling and monitoring multivariate industrial data that exhibit correlated/collinear variables. They are, however, limited due to their ineffective handling of processes nonlinearity by extracting only linear information from the data. Of course, linear LVR methods are inappropriate for data analysis and monitoring from complex nonlinear processes. To remedy this issue, various nonlinear LVR methods have been designed to extract nonlinear features and learn any nonlinear relation among process variables. NLVR-based monitoring methods are generally performed in two main steps, offline model building and online anomaly detection. In the first step, an

https://t.me/learningnets

empirical model of the inspected process is designed from data representing the nominal behavior of the process, and decision thresholds are computed using SPE and T^2 charts. Then, the designed model with the decision thresholds is used to check new testing measurements for anomaly detection purposes. Nonlinear LVR-based monitoring methods, such as KPCA and nonlinear PLS, are effective tools due to their detection efficiency and ease of implementation.

In this chapter, we showed that KPCA-based monitoring approaches compare favorably with the conventional PCA approach. The performance of these methods was assessed in terms of FAR and MDR, based on data from WWTP located in Saudi Arabia. KPCA-based lSVM method could reliably uncover abnormal events that occurred in influent flow measurements. Different kernels were investigated; when applied to WWTP data, KPCA-lSVM using RBF kernel performed better than the considered linear and nonlinear kernels.

In many practical applications such as air quality monitoring, multiple times series data are collected from different spatial locations with necessary spatio-temporal dependence, which is rarely considered in monitoring and decision-making. One direction for improvement is to extend the developed LVR-based monitoring techniques to account for spatio-temporal evolution of data (i.e., including information from spatial lags in the monitoring), and to utilize these improvements in various applications. Specifically, for the time-dependent data matrix, $\mathbf{X}_{n\times m}$, the dynamic of the data are considered by incorporating lagged variables in $\mathbf{X}$. For instance, the data matrix $\mathbf{X}$ could comprise the m variables observed at instant t and instant $t-1$. Similarly, observations from different locations can also be included in $\mathbf{X}$. Careful consideration should be given to which lags and variables to include. In spatial statistics, it is common to select the nearest neighbors only; however, the optimal choice depends on the correlation among variables and the dependence existing in the underlying spatiotemporal process. We will investigate how to adapt the nearest neighbor idea to multivariate spatio-temporal processes in order to borrow valuable information while controlling the dimensionality of $\mathbf{X}$.

In various practical processes (including environmental, biological, and hydrological), data are functional in nature. For example, dust measurements for air quality monitoring can be viewed as a function of time. It has been shown that PCA for multivariate observations is not suitable for functional data [80–83]. For functional data, functional PCA (FPCA), similar to PCA, captures the most variations in the data based on the first few orthogonal functional principal components [84,82,85]. Although FPCA is a popular statistical method in functional data analysis, it has not been used for process monitoring, fault detection, and diagnosis. The challenge is that space-time data are highly correlated and very high dimensional; in such cases, existing multivariate methods often have limitations, and suitable methods (e.g., control chart methods based on functional PCA) to deal with functional data are needed. Also, it will be very beneficial to develop function latent variable methods to monitor functional data.

https://t.me/learningnets

References

[1] B.R. Kowalski, M.B. Seasholtz, Recent developments in multivariate calibration, Journal of Chemometrics 5 (1991) 129–145.

[2] I. Frank, J. Friedman, A statistical view of some chemometric regression tools, Technometrics 35 (2) (1993) 109–148.

[3] M. Stone, R.J. Brooks, Continuum regression: cross-validated sequentially constructed prediction embracing ordinary least squares, partial least squares and principal components regression, Journal of the Royal Statistical Society, Series B 52 (2) (1990) 237.

[4] S. Wold, Soft Modeling: The Basic Design and Some Extensions, Systems Under Indirect Observations, Elsevier, Amsterdam, 1982.

[5] E. Malthouse, A. Tamhane, R. Mah, Non-linear partial least squares, Computers & Chemical Engineering 21 (8) (1997) 875–890.

[6] H. Hotelling, Relations between two sets of variables, Biometrika 28 (1936) 321–377.

[7] F.R. Bach, M.I. Jordan, Kernel independent component analysis, Journal of Machine Learning Research 3 (1) (2003).

[8] S.S.D.R. Hardoon, J. Shawetaylor, Canonical correlation analysis: an overview with application to learning methods, Neural Computation 16 (12) (2004) 2639–2664.

[9] M. Borga, T. Landelius, H. Knutsson, A unified approach to PCA, PLS, MLR and CCA, Technical Report, Linköping University, 1997.

[10] A. Maulud, D. Wang, J. Romagnoli, A multi-scale orthogonal nonlinear strategy for multivariate statistical process monitoring, Journal of Process Control 16 (7) (2006) 671–683.

[11] H. Peng, R. Wang, L. Hai, Sensor fault detection and identification using kernel PCA and its fast data reconstruction, in: Control and Decision Conference, 2010, pp. 3857–3862.

[12] S. Mika, B. Schölkopf, A.J. Smola, K.-R. Müller, M. Scholz, G. Rätsch, Kernel PCA and de-noising in feature spaces, in: Advances in Neural Information Processing Systems, 1999, pp. 536–542.

[13] B. Schölkopf, A. Smola, K.-R. Müller, Nonlinear component analysis as a kernel eigenvalue problem, Neural Computation 10 (5) (1998) 1299–1319.

[14] D. Dong, T.J. McAvoy, Nonlinear principal component analysis—based on principal curves and neural networks, Computers & Chemical Engineering 20 (1) (1996) 65–78.

[15] A.H. Monahan, Nonlinear principal component analysis by neural networks: theory and application to the Lorenz system, Journal of Climate 13 (4) (2000) 821–835.

[16] R. Rosipal, L.J. Trejo, Kernel partial least squares regression in reproducing kernel Hilbert space, Journal of Machine Learning Research 2 (2001) 97–123.

[17] S. Wold, Nonlinear partial least squares modelling II. Spline inner relation, Chemometrics and Intelligent Laboratory Systems 14 (1–3) (1992) 71–84.

[18] S. Wold, N.K. Wold, B. Skagerberg, Nonliner PLS modeling, Chemometrics and Intelligent Laboratory Systems 7 (1989) 53–65.

[19] S.J. Qin, T.J. McAvoy, Nonlinear PLS modeling using neural networks, Computers & Chemical Engineering 16 (4) (1992) 379–391.

[20] D. Lee, M. Lee, S. Woo, Y. Kim, J. Park, Nonlinear dynamic partial least squares modeling of a full scale biological wastewater treatment plant, Process Biochemistry 41 (1992) 2050–2057.

[21] Y.H. Bang, C.K. Yoo, I.-B. Lee, Nonliner PLS modeling with fuzzy inference system, Chemometrics and Intelligent Laboratory Systems 64 (2) (2003) 137–155.

[22] I. Araby Abdel-Rahman, G.J. Lim, A nonliner partial least squares algorithm using quadratic fuzzy system, Journal of Chemometrics 23 (2009) 530–537.

[23] I.E. Frank, A nonlinear PLS model, Chemometrics and Intelligent Laboratory Systems 8 (2) (1990) 109–119.

[24] I. Frank, NNPPSS: neural networks based on PCR and PLS components nonlinearized by smoothers and splines, in: INCINC94 Chemometrics Conference, 1994.

[25] I. Frank, A nonliner PLS model, Chemometrics and Intelligent Laboratory Systems 8 (1990) 109–119.

https://t.me/learningnets

[26] E.C. Malthouse, A.C. Tamhane, R. Mah, Nonlinear partial least squares, Computers & Chemical Engineering 21 (8) (1997) 875–890.
[27] M.A. Kramer, Nonlinear principal component analysis using autoassociative neural networks, AIChE Journal 37 (2) (1991) 233–243.
[28] S. Tan, M.L. Mayrovouniotis, Reducing data dimensionality through optimizing neural network inputs, AIChE Journal 41 (6) (1995) 1471–1480.
[29] S.W. Choi, C. Lee, J.-M. Lee, J.H. Park, I.-B. Lee, Fault detection and identification of nonlinear processes based on kernel PCA, Chemometrics and Intelligent Laboratory Systems 75 (1) (2005) 55–67.
[30] J.-M. Lee, C. Yoo, S. Choi, P. Vanrolleghem, I.-B. Lee, Nonlinear process monitoring using kernel principal component analysis, Chemical Engineering Science 59 (1) (2004) 223–234.
[31] S.J. Qin, Survey on data-driven industrial process monitoring and diagnosis, Annual Reviews in Control 36 (2) (2012) 220–234.
[32] Y. Wang, Y. Wei, T. Liu, T. Sun, K.T. Grattan, TDLAS detection of propane/butane gas mixture by using reference gas absorption cells and partial least square approach, IEEE Sensors Journal 18 (20) (2018) 8587–8596.
[33] R. Rosipal, Nonlinear partial least squares an overview, in: Chemoinformatics and Advanced Machine Learning Perspectives: Complex Computational Methods and Collaborative Techniques, IGI Global, 2011, pp. 169–189.
[34] G. Baffi, E. Martin, A. Morris, Non-linear projection to latent structures revisited (the neural network PLS algorithm), Computers & Chemical Engineering 23 (9) (1999) 1293–1307.
[35] Z. Cheng, L. Zhang, Spectral reconstruction and quantitative analysis by B-spline transformations and penalized partial least squares approach, Chinese Journal of Analytical Chemistry 37 (12) (2009) 1820–1824.
[36] Y. Jiang, Contributions to partial least squares regression and supervised principal component analysis modeling, PhD dissertation, The University of New Mexico, 2010.
[37] P. Royston, D.G. Altman, Regression using fractional polynomials of continuous covariates: parsimonious parametric modelling, Journal of the Royal Statistical Society. Series C. Applied Statistics 43 (3) (1994) 429–453.
[38] S. Lee, W.S. Choi, A multi-industry bankruptcy prediction model using back-propagation neural network and multivariate discriminant analysis, Expert Systems with Applications 40 (8) (2013) 2941–2946.
[39] T. Holcomb, M. Morari, PLS/neural networks, Computers & Chemical Engineering 16 (4) (1992) 393–411.
[40] G. Cybenko, Approximation by superpositions of a sigmoidal function, MCSS. Mathematics of Control, Signals and Systems 2 (4) (1989) 303–314.
[41] M. Mörtsell, M. Gulliksson, An overview of some non-linear techniques in chemometrics, FSCN, Mitthögskolan, 2001.
[42] B. Li, E.B. Martin, A.J. Morris, Box–Tidwell transformation based partial least squares regression, Computers & Chemical Engineering 25 (9–10) (2001) 1219–1233.
[43] H. Liu, C. Yang, B. Carlsson, S.J. Qin, C. Yoo, Dynamic nonlinear partial least squares modeling using Gaussian process regression, Industrial & Engineering Chemistry Research 58 (36) (2019) 16676–16686.
[44] H. Liu, X. Li, C. Yoo, Nonlinear PLS monitoring based on ANFIS, in: 2017 29th Chinese Control and Decision Conference (CCDC), IEEE, 2017, pp. 319–324.
[45] J.-S. Jang, ANFIS: adaptive-network-based fuzzy inference system, IEEE Transactions on Systems, Man and Cybernetics 23 (3) (1993) 665–685.
[46] C.K. Yoo, D.S. Lee, P.A. Vanrolleghem, Application of multiway ICA for on-line process monitoring of a sequencing batch reactor, Water Research 38 (7) (2004) 1715–1732.
[47] Y.H. Bang, C.K. Yoo, I.-B. Lee, Nonlinear PLS modeling with fuzzy inference system, Chemometrics and Intelligent Laboratory Systems 64 (2) (2002) 137–155.
[48] J.S.R. Jang, ANFIS-adaptive-network based fuzzy inference system, IEEE Transactions on Systems, Man and Cybernetics 23 (3) (1993) 665–685.

https://t.me/learningnets

[49] J.S.R. Jang, Fuzzy modeling using generalized neural networks and Kalman filtering algorithm, in: Proc. Ninth Nat. Conf. Artificial Intell. (AAAI-91), 1991, pp. 761–767.
[50] J.S.R. Jang, Rule extraction using generalized neural networks, in: Proc. 4th IFSA World Congress, vol. 23(3), 1991, pp. 191–212.
[51] J.-S.R. Jang, C.-T. Sun, E. Mizutani, Neuro-fuzzy and soft computing–a computational approach to learning and machine intelligence, book review, IEEE Transactions on Automatic Control 42 (10) (1997) 1482–1484.
[52] H. Liu, M. Huang, C. Yoo, A fuzzy neural network-based soft sensor for modeling nutrient removal mechanism in a full-scale wastewater treatment system, Desalination and Water Treatment 51 (31–33) (2013) 6184–6193.
[53] H. Liu, M. Huang, J.T. Kim, C. Yoo, Adaptive neuro-fuzzy inference system based faulty sensor monitoring of indoor air quality in a subway station, Korean Journal of Chemical Engineering 30 (3) (2013) 528–539.
[54] M.K. Goyal, B. Bharti, J. Quilty, J. Adamowski, A. Pandey, Modeling of daily pan evaporation in subtropical climates using ANN, LS-SVR, fuzzy logic, and ANFIS, Expert Systems with Applications 41 (11) (2014) 5267–5276.
[55] S. Qin, Statistical process monitoring: basics and beyond, Journal of Chemometrics 17 (8/9) (2003) 480–502.
[56] M. Madakyaru, F. Harrou, Y. Sun, Monitoring distillation column systems using improved nonlinear partial least squares-based strategies, IEEE Sensors Journal 19 (23) (2019) 1–9.
[57] X. Wu, V. Kumar, J.R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G.J. McLachlan, A. Ng, B. Liu, S.Y. Philip, et al., Top 10 algorithms in data mining, Knowledge and Information Systems 14 (1) (2008) 1–37.
[58] F. Harrou, M. Madakyaru, Y. Sun, Improved nonlinear fault detection strategy based on the Hellinger distance metric: plug flow reactor monitoring, Energy and Buildings 143 (2017) 149–161.
[59] G.T. Jemwa, C. Aldrich, Classification of process dynamics with Monte Carlo singular spectrum analysis, Computers & Chemical Engineering 30 (5) (2006) 816–831.
[60] B. Schölkopf, A. Smola, K.-R. Müller, Kernel principal component analysis, in: International Conference on Artificial Neural Networks, Springer, 1997, pp. 583–588.
[61] J.-D. Shao, G. Rong, J.M. Lee, Learning a data-dependent kernel function for KPCA-based nonlinear process monitoring, Chemical Engineering Research and Design 87 (11) (2009) 1471–1480.
[62] M. Jia, H. Xu, X. Liu, N. Wang, The optimization of the kind and parameters of kernel function in KPCA for process monitoring, Computers & Chemical Engineering 46 (2012) 94–104.
[63] J. Ni, C. Zhang, L. Ren, S.X. Yang, Abrupt event monitoring for water environment system based on KPCA and SVM, IEEE Transactions on Instrumentation and Measurement 61 (4) (2011) 980–989.
[64] R.T. Samuel, Y. Cao, Nonlinear process fault detection and identification using kernel PCA and kernel density estimation, Systems Science & Control Engineering 4 (1) (2016) 165–174.
[65] Z. Ge, Z. Song, Multivariate Statistical Process Control: Process Monitoring Methods and Applications, Springer Science & Business Media, 2012.
[66] Z. Ge, Z. Song, F. Gao, Review of recent research on data-based process monitoring, Industrial & Engineering Chemistry Research 52 (10) (2013) 3543–3562.
[67] T. Cheng, A. Dairi, F. Harrou, Y. Sun, T. Leiknes, Monitoring influent conditions of wastewater treatment plants by nonlinear data-based techniques, IEEE Access 7 (1) (2019) 108827–108837.
[68] L. Xie, S. Wang, Recursive kernel PCA and its application in adaptive monitoring of nonlinear processes, Journal of Chemical Industry and Engineering, China 58 (7) (2007) 1776.
[69] S.W. Choi, I.-B. Lee, Nonlinear dynamic process monitoring based on dynamic kernel PCA, Chemical Engineering Science 59 (24) (2004) 5897–5908.
[70] X. Liu, U. Kruger, T. Littler, L. Xie, S. Wang, Moving window kernel PCA for adaptive monitoring of nonlinear processes, Chemometrics and Intelligent Laboratory Systems 96 (2) (2009) 132–143.

https://t.me/learningnets

[71] Y. Zhang, C. Ma, Fault diagnosis of nonlinear processes using multiscale KPCA and multiscale KPLS, Chemical Engineering Science 66 (1) (2011) 64–72.
[72] Y. Zhang, S. Li, Z. Hu, Improved multi-scale kernel principal component analysis and its application for fault detection, Chemical Engineering Research and Design 90 (9) (2012) 1271–1280.
[73] C. Chouaib, H. Mohamed-Faouzi, D. Messaoud, Adaptive kernel principal component analysis for nonlinear dynamic process monitoring, in: 2013 9th Asian Control Conference (ASCC), IEEE, 2013, pp. 1–6.
[74] Y. Zhang, Fault detection and diagnosis of nonlinear processes using improved kernel independent component analysis (KICA) and support vector machine (SVM), Industrial & Engineering Chemistry Research 47 (18) (2008) 6961–6971.
[75] J.-M. Lee, S.J. Qin, I.-B. Lee, Fault detection of non-linear processes using kernel independent component analysis, Canadian Journal of Chemical Engineering 85 (4) (2007) 526–536.
[76] Y. Zhang, Enhanced statistical analysis of nonlinear processes using KPCA, KICA and SVM, Chemical Engineering Science 64 (5) (2009) 801–811.
[77] C.K. Yoo, I.-B. Lee, Nonlinear multivariate filtering and bioprocess monitoring for supervising nonlinear biological processes, Process Biochemistry 41 (8) (2006) 1854–1863.
[78] T. Cheng, F. Harrou, Y. Sun, T. Leiknes, Monitoring influent measurements at water resource recovery facility using data-driven soft sensor approach, IEEE Sensors Journal 19 (1) (2018) 342–352.
[79] C. Wang, L. Chen, S. Xia, F. Sun, Maximum production rate optimization for sulphuric acid decomposition process in tubular plug-flow reactor, Energy 99 (2016) 152–158.
[80] C. Croux, A. Ruiz-Gazen, High breakdown estimators for principal components: the projection-pursuit approach revisited, Journal of Multivariate Analysis 95 (1) (2005) 206–226.
[81] F. Ferraty, P. Vieu, Nonparametric Functional Data Analysis: Theory and Practice, Springer Science & Business Media, 2006.
[82] J.O. Ramsay, Functional data analysis, in: Encyclopedia of Statistical Sciences, vol. 4, 2004.
[83] H.L. Shang, A survey of functional principal component analysis, AStA Advances in Statistical Analysis 98 (2) (2014) 121–142.
[84] J.R. Berrendero, A. Justel, M. Svarc, Principal components for multivariate functional data, Computational Statistics & Data Analysis 55 (9) (2011) 2619–2634.
[85] Y. Sun, M.G. Genton, D.W. Nychka, Exact fast computation of band depth for large functional datasets: how quickly can one million curves be ranked?, Stat - The ISI's Journal for the Rapid Dissemination of Statistics Research 1 (1) (2012) 68–74.

https://t.me/learningnets

Chapter 5

Multiscale latent variable regression-based process monitoring methods

5.1 Introduction

Accurate and flexible modeling of process variables based on implicit or explicit models is necessary for improving system performance monitoring and process control. In other words, the prediction accuracy of the designed model to mimic the nominal conditions of the inspected process or plant has a direct effect on the performance of fault detection and diagnosis. In various practical processes (including environmental, biological, and hydrological), data are collinear (or contain redundancy among the variables), tainted with noise, and nonstationary. However, data generated from multivariate processes are statistically challenging because they are highly dimensional, the relationships among variables are nonlinear, the variables themselves are non-Gaussian, and the processes change over time. Moreover, various types of fault can occur. Thus, the timely detection of abnormal conditions of complex systems using conventional monitoring methods can be challenging and ineffective.

Fault detection is of great significance for operators to detect abnormal conditions timely, and to take effective measures to avoid adverse consequences. In general, the two essential assumptions underlying the design of most conventional process monitoring methods are that the process data need to be both independent and identically normally distributed. However, data collected from modern industrial processes often are autocorrelated and/or nonnormally distributed. The violation of these major assumptions can seriously affect the monitoring performance of these conventional charts. Due to its core role in industrial processes, fault detection is extensively studied in academia and industry. Various studies reported in the literature have focused on monitoring charts for monitoring processes with time-dependent data [1–5]. There are two main techniques used to monitor processes with autocorrelated data: residuals-based and adapting model-based methods [4,6]. Residuals-based methods to monitor processes with autocorrelated data are the pioneer in the fault-detection field, with a lot of impressive results reported in the literature. The basic idea behind

https://doi.org/10.1016/B978-0-12-819365-5.00011-5
Copyright © 2021 Elsevier Inc. All rights reserved.
https://t.me/learningnets

these approaches is to describe the autocorrelation in the data, using a mathematical model, and to then apply traditional control charts, using computed uncorrelated residuals obtained from the constructed model. The performance of residual-based monitoring charts, however, is very sensitive to the prediction quality of the model [4,7]. Other techniques monitor the autocorrelated data by adapting the model to each sampling time [4,6]. Nevertheless, due to the high complexity of the modern industry, designing an accurate model is not always an easy task [8]. In reality, various chemical, environmental, and biological processes provide data that are not normally distributed. In such cases, the normality assumption underlying the conventional statistical monitoring chart is invalid. Many researchers have studied the impact of departure from normality on process monitoring charts [8–10]. For example, the author of [9] have investigated the impact of a violation of normality assumption on the Shewhart chart, by using various known distributions, namely uniform, right triangular, gamma, and bimodal distributions. Also, Burrows [11] studied the impact of skewed distribution on the Shewhart chart. Several works have designed process monitoring charts for non-Gaussian distribution when the form of the underlying distribution is known. The authors of [12,13] have developed an EWMA monitoring scheme for multivariate Poisson-distributed data. Also, univariate monitoring charts were designed to monitor nonnormally chi-square distributed processes [14]. However, we usually use the normal theory results in the absence of information regarding the form of distribution of the data [8]. Then, ignoring the nonnormality in data can result in significant degradation of the detection quality of the designed monitoring approach [15].

Most real processes, however, are multivariate in nature. The conventional multivariate fault-detection methods also rely on most of the assumptions made in univariate techniques, such as the normality, independence of the data or residuals. The collinearity in the data can lead to a large uncertainty of the model parameters and degrades the model prediction. The most commonly used strategy to reduce the effect of the collinearity on the model efficacy is to use latent variable regression (LVR) models. The basic idea behind the LVR models is that most of the information on the data is captured by transforming the variables into a smaller set of variables that are used for model development. The LVR model is developed using a smaller set of variables that linearly combine the original variables. This leads to improvements in prediction as well as well-conditioned models for estimation [16]. Some of the well-known LVR model techniques include principal component regression (PCR) [17,18] and partial least squares (PLS) [17,19,20].

Another challenge in designing process monitoring schemes is to overcome the high noise embedded in the measured data. Most of the data measured from processes are corrupted with noise which is of different forms, including random and gross errors. These errors are largely due to disturbances, fluctuations, sensor degradation, and human errors. These errors may swamp the important features of the data that are essential to detect anomalies during the estima-

https://t.me/learningnets

tion of the model parameters. Thus, measurement noise must be filtered to get an improved model prediction. Unfortunately, most practical data contain relevant features and noise that have contributions in time and frequency. This kind of data is multiscale in nature [21]. For example, slow change in the data covers a wide range in the time domain and a small range in the frequency domain. Similarly, a large change in the measured data covers a small range in the time domain and a wide range in the frequency domain. Conventional filtering techniques, such as a mean filter or exponentially weighted moving average filter, fail to separate the feature-noise due to the data being multiscale in nature. Convectional low pass filters classify the noise as a high-frequency feature and therefore filter data higher than a defined threshold. Therefore, to take care of the multiscale nature of the data, it requires the use of multiscale model estimation methods. Wavelet-based techniques are powerful tools to separate pertinent features from noise.

Various researchers have applied multiscale methods to improve the accuracy of prediction of the estimated model [22–28]. For example, Palavajjhala et al. proposed the wavelet prefilters for modeling purposes using the multiscale representation of data [23]. Bhakshi [22] have demonstrated the benefits of data representing in multiscale for empirical modeling and shown that it can enhance the noise removal capability of the PCA model and its capacity for supervising multivariate processes. Other researchers showed that collinearity can be reduced by the multiscale representation of data and also that it is able to shrink the variations of FIR model parameters [25,26]. In system identification steps, wavelets have been used as a modulating function for control-relevant system identification [28]. Unfortunately, parameter estimation of LVR models using multiscale data has not been exploited much in the literature, which is the main focus of this chapter.

Motivated by the above-mentioned challenges and hindrances limiting the mentioned conventional univariate and multivariate fault detection approach, the aim of this chapter is to present a set of wavelet-based multiscale monitoring and fault detection methods to address these challenges. In this chapter, we offer a brief review of the wavelets, present their benefits, and introduce the multiscale representation of data using wavelets. We then show the impact of ignoring the basic assumptions underlying the data structure on the detection performance of a univariate monitoring chart, and subsequently introduce the core idea of wavelet-based univariate monitoring charts, i.e., multiscale EWMA, CUSUM, and GLR fault detection methods, that can improve the performance of these techniques, especially when their assumptions are violated. An illustration of the concept of wavelet-based multivariate extension of LVR approaches is provided. In our last section, we highlight the major results obtained when applying multiscale monitoring approaches to monitor distillation columns. Lastly, we discuss the challenges that remain when using multiscale approaches.

https://t.me/learningnets

5.2 Theoretical background of wavelet-based data representation

Conventional and well-known monitoring schemes, such as EWMA, CUSUM, and GLR, would not be effective in monitoring nonstationary, non-Gaussian, and correlated processes. Some have been improved to handle the possible correlation among process measurements by describing the correlation via a statistical model, such as the autoregressive integrated moving average (ARIMA), which is commonly used and also quite flexible time series model. However, most of these models are designed by assuming stationarity of the data and they are effective to detect anomalies only of a certain range. For this reason, we choose to use wavelet-based methods commonly used to alleviate the stationarity assumption and the limitations encountered in many conventional monitoring schemes.

Let us first focus on the core idea and benefits of wavelets. Generally speaking, wavelet functions look at data on different scales or frequency components and are able to analyze every frequency component, via a level-matched resolution. For instance, if we look at the data with a large window, we can notice essential features. In the same manner, if we look into the data with a small window, we would notice the small features. Indeed, most of data-based empirical modeling approaches involve representing the data for each of the variables, using basis functions that represent all the measured variables individually. A large part of the methods are currently used to represent data at a single scale or fixed in time and frequency. This type of representation is useful only for data-containing contributions with a uniform localization everywhere in time and frequency. In practical situations, it is rare to find measured data with a uniform localization or a single scale. Measured data involve stochastic and deterministic events. Most of practical data that contain stochastic components of the measured data are a combination of scale and time-dependent parameters, whereas a deterministic component of the data is at multiple locations of the frequency and time. Also, measured variables sometimes may have different sampling rates or may have some missing segments.

Indeed, each occurred event is linked with some frequency band. Wavelet methods are an efficient tool enabling the decomposition of the original data into multiple frequency bands and thus permitting the simultaneous analysis in time and frequency domains. Besides the limitation described above, the conventional monitoring methods have the other disadvantage of being single-scale (time scale) and may not be able to efficiently handle multiscale data that are frequently present in modern industrial processes. Because of the multiscale nature of the data from most engineering and environmental processes, it is necessary to include these features when designing monitoring methods. This means that some events occur at different localizations in time and frequency, such as processes for which the power spectrum changes with time and/or frequency, and for variables collected at different sampling rates. Most of the data-based empirical modeling approaches involve representing the data

https://t.me/learningnets

for each of the variables using a basis function that, for example, represents each measured variable. Wavelet techniques have been shown to be highly competitive when data are multiscale, correlated, and do not follow a Gaussian distribution. Essentially, wavelet-based monitoring provides more desirable features than conventional methods for several reasons; it generates uncorrelated wavelet coefficients at different scales and decomposes signals into several frequency bands that enable the analysis of data in time and frequency domains simultaneously.

5.2.1 Wavelet transform

A time-series data can be represented using wavelets as coefficients that correspond to a specific time and frequency [29]. Wavelet decomposition has been extensively used in data compression, signal and image processing, speech discrimination, model prediction, and process monitoring. Wavelet decomposition is a sophisticated method that allows obtaining localized components, unlike other well-known decomposition methods such as the Fourier transform. In Fourier transform, data is decomposed as a combination of sines and cosines, where the different frequencies are calculated from the global period of the signal, thus maintain specificity in frequency alone. Contrary to the Fourier transform, a localized frequency analysis is obtained by the wavelet decomposition. Thus, it comes with information about the frequency components that exist in a signal and their time of occurrence [30]. Additionally, the wavelet filter can quickly capture high-frequency features and acquire slowly the low-frequency ones; this makes it very efficient to expose patterns with different magnitudes and durations in a signal, while preserving the timing very precise.

Another aspect is the fact that the wavelet transform has the ability to distinguish the stochastic components from the deterministic components taken from the data. The deterministic components are captured by a relatively small number of large coefficients, whereas stochastic changes are distributed among all coefficients. The major advantage of the wavelet transform is that it represents the data in varying windows. It should be noted that to overcome the problem of discontinuity in the signal, very short basis functions are used, and to get detailed frequency analysis, long basis functions can be used. This way, one can achieve long low-frequency and high-frequency basis functions. In the following section, we shed more light on the multiscale representation based on wavelets and its advantages.

5.2.2 Multiscale representation of data using wavelets

Historically, multiresolution time-series decomposition was introduced as the quintessential mathematical tool for image decoding and compression [31]. In [31], Mallat employed orthogonal wavelets for image compression, because

https://t.me/learningnets

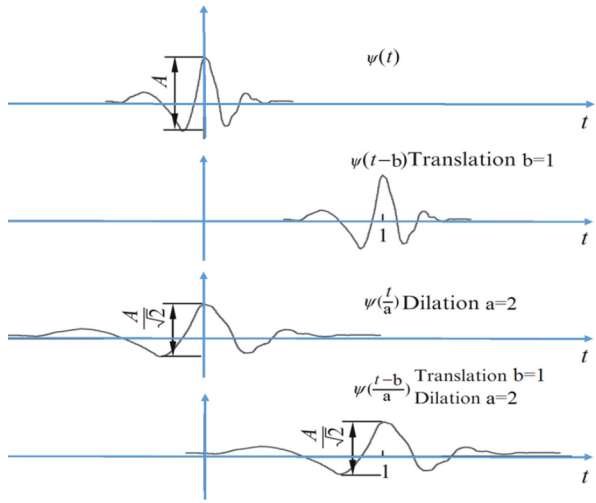


FIGURE 5.1 Illustration of the dilation and translation mechanism in wavelets.

of their capacity to easily adapt to patterns of images and to reconstruct them with reduced space. Nowadays, wavelets are used in numerous applications, including data compression, image analysis, time–frequency localization, and process monitoring [32–35]. Wavelets represent a set of mathematical functions that can offer a localized analysis in both time and frequency [34]:

$$\psi_{a,b}(t) = \frac{1}{\sqrt{a}}\psi\left(\frac{t-b}{a}\right), \tag{5.1}$$

where a and b are respectively the dilation and translation parameters [32]. The function $\psi(t)$ represents the mother wavelet, which becomes $\psi\left(\frac{t-b}{a}\right)$ when stretched or shrunk. Fig. 5.1 shows how the mother wavelet is stretched and shrunk to capture features in time and frequency. In practice, the dyadically discretized version of the dilation and the translation parameters are commonly used: $a = 2^m$, $b = 2^m k$, $(m, k) \in \mathbb{Z}^2$. Thus, the family of wavelets can be expressed as $\psi_{mn}(t) = 2^{\frac{-m}{2}}\psi(2^{-m}t - m)$. Here, m and k respectively denote the dilation and translation parameters. Numerous families of basis functions are generated, based on their convolution with different filters, such as the Haar scaling function and the Daubechies filters [32,36,37]. Generally speaking, the form of the wavelet function is related to the wavelet family; for instance, the Haar wavelet is discrete symmetrically square-shaped, and Daublets, symmlets, and Coiflets are almost symmetric (Fig. 5.2). It should be noted that the dyadic discretization imposes downsampling, which reduces the number of parameters dyadically at each decomposition. Indeed, with this discretization, decomposition of samples at nondyadic locations is performed with a certain time delay.

https://t.me/learningnets

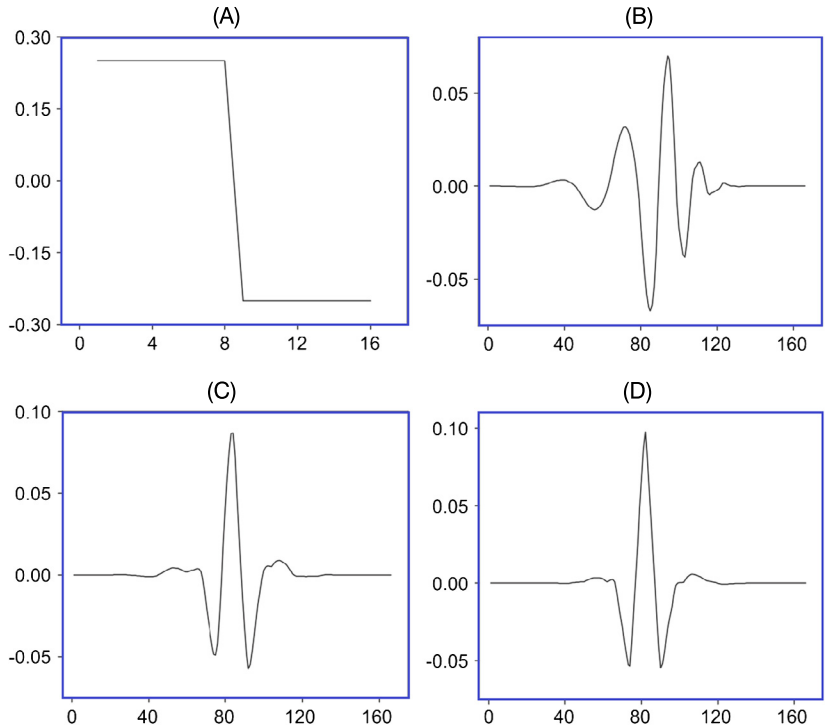


FIGURE 5.2 Different form of wavelets: (A) Haar wavelet; (B) Daublet of order 12 wavelet; (C) Symmlet of order 12 wavelet; (D) Coiflet of order 12 wavelet.

Discrete Wavelet Transform (DWT) is an efficient procedure for separating deterministic from stochastic components of the input signal [34,38]. Thus, it is widely used for denoising measurements and changing detection [39,40]. The essence of the DWT is to represent time series data as a combination of approximation and detail coefficients [38]. Using DWT, the time series can be decomposed as [28,41,42]:

$$x(t) = \overbrace{\sum_{k=1}^{n2^{-J}} \mathbf{a}_{Jk}\phi_{Jk}(t)}^{A_J(t)} + \sum_{j=1}^{J} \overbrace{\sum_{k=1}^{n2^{-j}} \mathbf{d}_{jk}\psi_{jk}(t)}^{D_j(t)}, \tag{5.2}$$

where $A_J(t)$ and $D_j(t)$ are respectively the approximation and detail coefficients, and J is the decomposition level.

The detail coefficients are generated by projecting the original signal $x(t)$, using a set wavelet basis functions defined as $\psi_{j,k}(t) = \sqrt{2^{-j}}\psi(2^{-j}t - k)$, $j = 1, \ldots, J$, $k \in \mathbb{Z}$, where k is the shift parameter; ψ is the mother wavelet used. In other words, the detailed signal $D_j(t)$ at scale j can be obtained by ap-

https://t.me/learningnets

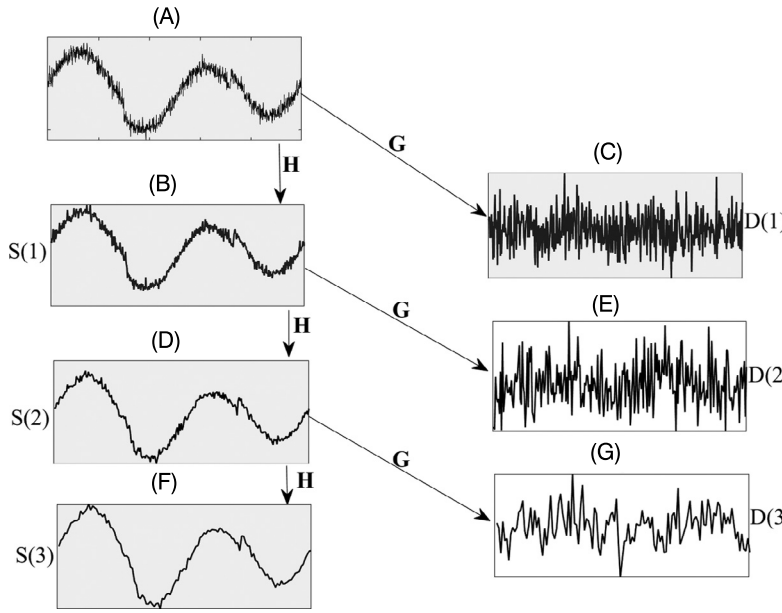


FIGURE 5.3 Illustration of DWT of a heavy sine (approximations S_i and details d_i) using Haar wavelet; here, $L = 3$.

plying a high-pass filter (g) on the original and scaled signals. In a similar way, the approximation coefficients are generated by projecting the signal on a set of orthonormal scaling functions given by $\phi_{j,k}(t) = \sqrt{2^{-j}}\phi(2^{-j}t - k)$, $j = 1, \ldots, J, k \in \mathbb{Z}$. Similarly, the scale signals are computed by applying a low-pass filter (h) on the original and scaled signals.

To illustrate the DWT decomposition process, an example of multiscale representation-based DWT applied to the square-integrable signal is presented in Fig. 5.3. This figure shows a signal, S, at three scales, using DWT; S_1 and d_1 are the respective approximation and detail coefficients at level 1; S_1 is represented by the sum of the approximation and detail coefficients at level 2, S_2 and d_3, respectively; and so on. The signals in Figs. 5.3B, 5.3D, and 5.3F are at increasingly coarser levels, in comparison with the original signal presented in Fig. 5.3A. A low-pass filter (H) of length, r, is used to obtain scaled signals, which is equivalent to projecting the original signal on a set of orthonormal scaling functions, $\phi_{jk}(t)$. Figs. 5.3C, E, and G represent the detail signals that capture the details between the signal at a finer scale and the scaled signal. Projecting the signal onto the wavelet basis function, $\psi_{jk}(t)$, is equivalent to using a high-pass filter of length r, $\mathbf{g_f} = [g_1, g_2, \ldots, g_r]$ for filtering the scaled signal at the finer scale. Of course, Fig. 5.3 shows the approximation signal, S_3 (Fig. 5.3F), representing the low frequencies (large scales), and the coefficients of details D_1, D_2, D_3, and D_4 with large variations, especially scales

https://t.me/learningnets

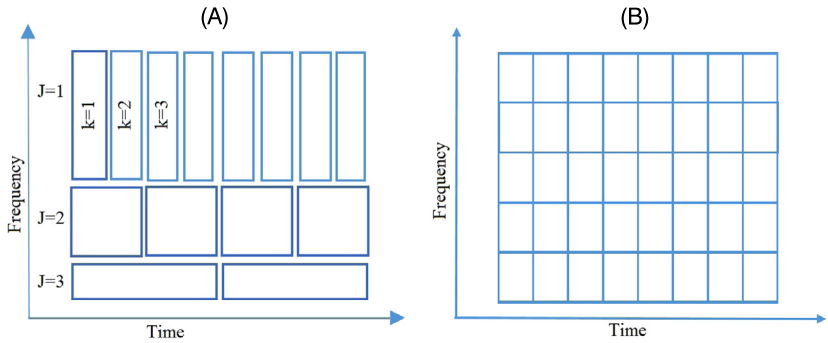


FIGURE 5.4 (A) Spanning of time–frequency plane using wavelets with variable aspect ratio; (B) Fourier transform with fixed aspect ratio.

that represent high frequencies. This decomposition process can be performed at multiples scales, as many as $J = \log_2(n)$. Indeed, the number of coefficients decreases by 2, from one level to the next, due to the down-sampling in DWT after every filtering step, during which the filtered output is subsampled by two. In other words, the length of the scaled signal, at level (j), is half the length of the scaled signal at the previous scale $(j-1)$.

Fig. 5.4A provides a better understanding of the effect of the dilation parameter (j) and translation parameter (k), in the time–frequency plane. It can be seen that, for a fixed value of j and k, the energy is concentrated at the appropriate box. Fig. 5.4A shows that as k changes, the wavelet is changing in the time scale whereas energy in the frequency remains unchanged. Similarly, changing j results in a dilation of the functions ϕ and ψ, and also it doubles the span in time [28,41].

By analogy, in Fourier transform, the input signal is decomposed using a sinusoidal base. Each sine function corresponds to a given frequency weighted by coefficients, so-called Fourier coefficients. Thus, Fourier transform-based methods are not appropriate for data exhibiting features that vary with the frequency [43,44]. Indeed, the major advantage of using the wavelet transform, in comparison with Fourier transform-based methods, is that the size of the analysis window (mother wavelet) is variable (Fig. 5.4A). Each dilation or compression of the mother wavelet gives rise to a scale. In addition, the wavelet transform preserves the temporal information against the Fourier transform, thus generating a 2D time scale representation (Fig. 5.4A–B).

In [45], fast wavelet transform algorithms with complexity of $O(n)$ for a discrete signal of dyadic length are presented. It has been shown that wavelets and scaling functions coefficients at a specific scale (j), $\mathbf{a}_j$ and $\mathbf{d}_j$, are obtained just by the multiplication of the scaling coefficient vector at the finer scale, $\mathbf{a}_{j-1}$, by the matrices, $\mathbf{H}_j$ and $\mathbf{G}_j$, respectively, i.e.,

$$\mathbf{a}_j = \mathbf{H}_j\, \mathbf{a}_{j-1}, \quad \text{and} \quad \mathbf{d}_j = \mathbf{G}_j\, \mathbf{a}_{j-1} \tag{5.3}$$

https://t.me/learningnets

where

$$\mathbf{H}_j = \begin{bmatrix} h_1 & . & h_r & . & . \\ 0 & h_1 & . & h_r & 0 \\ 0 & 0 & . & . & . \\ 0 & 0 & h_1 & . & h_r \end{bmatrix}_{n2^j \times n2^j} \quad \text{and}$$

$$\mathbf{G}_j = \begin{bmatrix} g_1 & . & g_r & . & . \\ 0 & g_1 & . & g_r & 0 \\ 0 & 0 & . & . & . \\ 0 & 0 & g_1 & . & g_r \end{bmatrix}_{n2^j \times n2^j} . \tag{5.4}$$

Overall, wavelet-based multiscale techniques have gained popularity for handling both stationary and nonstationary data, as they can analyze time and frequency localized features simultaneously, with a high resolution. In addition, these methods are adaptable to transient signals. Accordingly, wavelet-based methods are designed and preferred for various purposes such as denoising, data compression, and process monitoring.

5.2.3 Advantages of multiscale representation

Autocorrelation and non-Gaussianity of process variables present important challenges when designing and implementing monitoring techniques. Here, we show the great potential of wavelet-based multiscale representations to decrease the autocorrelation degree of autocorrelated process and make the distribution of wavelet coefficients close to Gaussian, even if input data have non-Gaussian distributions.

5.2.3.1 Decorrelating autocorrelated measurements

One main characteristic of the multiscale representation is its capacity for decorrelating autocorrelated measurements at a multiscale level, making it an elegant and flexible tool for presenting and supervising environmental and engineering processes over conventional monitoring methods [46]. Generally speaking, a white noise process, which is uncorrelated with its values at any lag, has an autocorrelation function (ACF) of zero at all lags except the value of unity at lag zero. In contrast, time-dependent processes including autoregressive (AR) or autoregressive moving average (ARMA) process have nonzero values at lags other than zero that show a correlation among different lagged measurements [47]. It should be noted that the monitoring quality of conventional fault-detection methods can be deteriorated in the presence of autocorrelated measurement noise [1,48,49]. In the literature, two main strategies can be distinguished to monitor autocorrelated data. In the first strategy, the decision thresholds are adjusted to consider the time dependence in the data by estimating the true variance of the process [50]. In the second strategy, a model is first constructed and

https://t.me/learningnets

then used to generate residuals that are checked by the conventional monitoring schemes for anomaly detection [2]. For more details about methods for monitoring autocorrelated data, see [51,52].

The advent of the multiscale presentation can alleviate these challenges and overcome fault detection limitations that occur when using many of the conventional monitoring methods. Here, we show the capacity of wavelet decomposition to significantly decrease autocorrelation of input data that are normally-like distributed. An AR model with order 1 is used to generate autocorrelated data. The time-domain of a signal generated from an AR(1) model with an autoregression coefficient of $a = 0.7$ is illustrated in Fig. 5.5A. Fig. 5.5C, which shows the ACF of the time domain signal, indicates that the AR(1) signal is autocorrelated as expected. Fig. 5.5 gives an example of the capacity of wavelet decomposition of decorrelating the autocorrelated AR(1) data at multiple levels. Specifically, wavelet coefficients corresponding to time-dependent processes become almost uncorrelated at multiple scales (Fig. 5.5). Detailed representations of the wavelet signals are presented in Figs. 5.5D, G, and J. The distribution of the time domain and all detailed signals are close to Gaussian, as shown in the second column in Fig. 5.5. From Figs. 5.5F, I, and L, it is clear that the detail signals are relatively uncorrelated; this can be attributed to the application of the high-pass filters in wavelet decomposition.

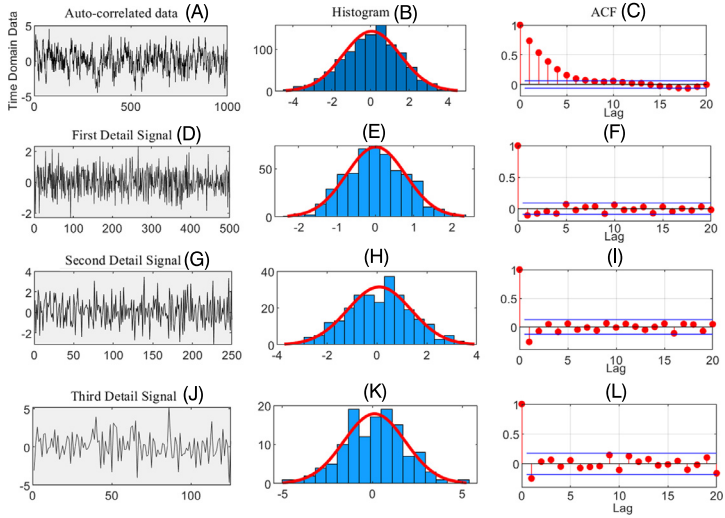


FIGURE 5.5 An illustrative example of the capacity of DWT in decorrelating time-dependent measurements generated from AR(1) model at multiple scales.

From a fault detection point of view, the presence of autocorrelated data degrades the effectiveness of various fault detection methods. Therefore, having the noise decorrelated at multiple scales should provide an advantage for any fault detection method, if it is applied using the detail signals at multiple scales.

https://t.me/learningnets

This advantage can be used to develop multiscale univariate and multivariate fault detection methods with better performance, and to show these improvements through their utilization to monitor various chemical and environmental systems.

5.2.3.2 *Data are closer to normality at multiple scales*

As discussed above, another factor affecting the effectiveness of fault detection is the presence of non-Gaussian errors in the data. In practice, we are confronted with data that are non-Gaussian distributed (e.g., gamma, χ^2, Poisson). Several studies have investigated the effect of violating normality on the performance of the conventional monitoring techniques and showed that poor performances are obtained in terms of the false alarm rates as well as missed detection rates [9,10,15]. Four main methods to handle non-Gaussian data have been reported in the literature. Those involve: (1) transforming the original data to get approximately Gaussian distributed data using transformation techniques, such as Box–Cox [53], (2) adjusting the decision thresholds based on statistical features of data (kurtosis and skewness), (3) designing a specific monitoring technique to the distribution of the data [54], and (4) using nonparametric or distribution-free methods [55,56].

Another key reason for choosing a multiscale presentation is its ability to make the distribution of data close to Gaussian at multiple scales [57,58]. Figs. 5.6 and 5.7 show that the signals of uniform and chi-square data are closer

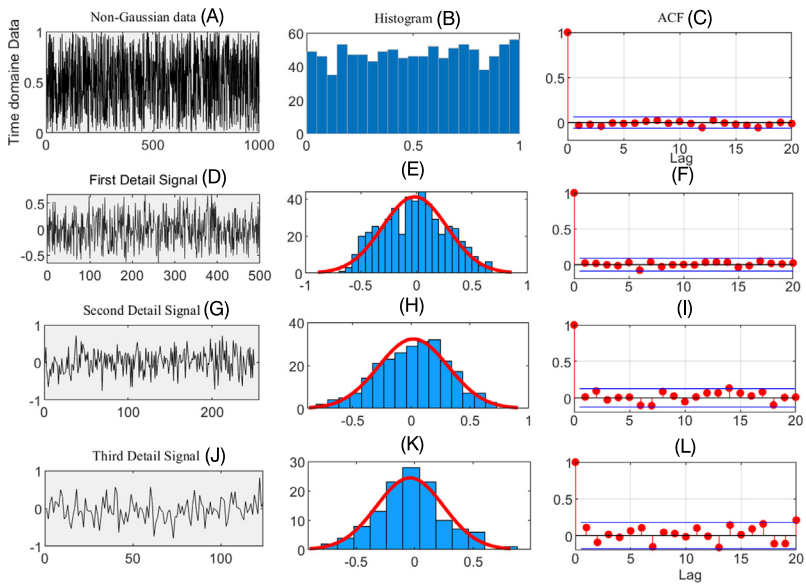


FIGURE 5.6 ACF and distributions of wavelet coefficients of uniformly distributed data.

https://t.me/learningnets

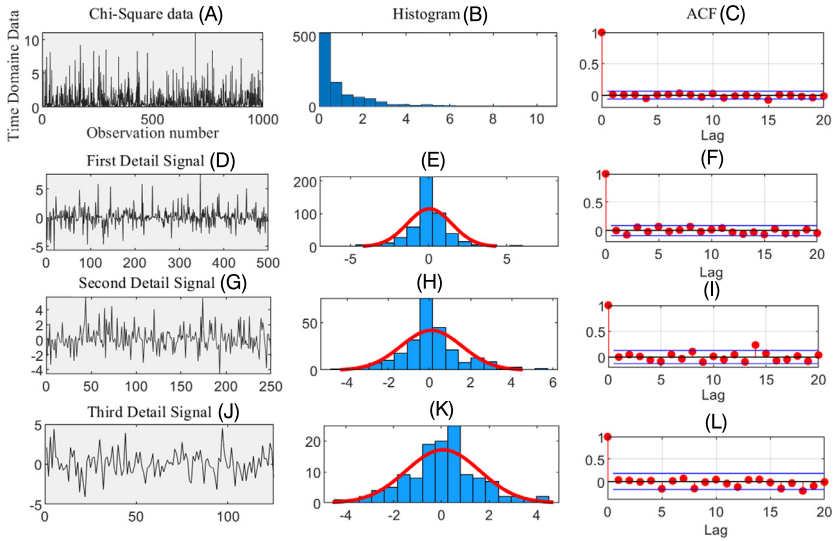


FIGURE 5.7 ACF and distributions of wavelet coefficients of χ^2 distributed data.

to normality at multiple scales. As discussed above, the performances of various fault detection methods can dramatically decrease when the data diverge from normality. To remedy that, multiscale representation is a potential solution to help improve the performance of these methods in the presence of non-Gaussian data. Theoretical demonstrations on the characteristics of multiscale representation can be found in [59–61].

5.3 Multiscale filtering using wavelets

Eliminating or reducing noise from collected data by keeping the maximum of underlying process information is an essential task and is usually termed as data filtering or denoising. This section offers an introduction to multiscale filtering, which is designed based on a multiscale representation.

5.3.1 Single scale filter method

In general, single scale linear filters are built by calculating the weighted sum of past data in a window size of finite or infinite length. These conventional denoising methods, such as the mean and exponential filters, are known as finite impulse response (FIR) and infinite impulse response (IIR) filters. The commonly used linear filter is expressed by

$$\hat{y}_k = \sum_{i=0}^{N-1} w_i y_{k-i}, \tag{5.5}$$

https://t.me/learningnets

where $\sum_i w_i = 1$, and N is the filter length. Linear filters include moving average, CUSUM, Chebyshev, and EWMA filters. For instance, the mean filter is implemented by using equal weights, i.e., $w_i = \frac{1}{N}$, whereas the EWMA filter proceeds by averaging all the past observations. The EWMA filter is performed in a recursive way as

$$\hat{y}_k = \alpha y_k + (1 - \alpha)\hat{y}_{k-1}, \tag{5.6}$$

where y_k and $\hat{y}_k$ are respectively the observation and the filter output at time step k and $\alpha \in [0, 1]$ is the smoothing parameter that defines the depth of the memory of EWMA. From the construction of the EWMA filter, its performance would depend on the value of the parameter α. Values of α close to one would result in less smoothed data and smaller values of α would result in more smoothed data [62].

It should be noted that in a single scale filtering, the measured data represented by basis functions have a uniform temporal localization with respect to the sampling time. In the case of single scale data analysis, the basis functions have uniform time-frequency localization which results in a trade-off between temporally localized changes and capacity to temporally remove global noise [24]. In other words, the major limitation of single scale filtering is their fixed time–frequency localization, i.e., they explore data at a single frequency alone. Moreover, these filters are not suited when the data are time-dependent and non-Gaussian-distributed. Accordingly, accurate feature extraction and noise removal are not possible, in an effective manner, using single-scale filtering strategies [24].

5.3.2 Multiscale filtering methods

Recently, denoising techniques based on wavelet representation received considerable attention from researchers and engineers. The potential of these nonlinear denoising methods is still being investigated. Indeed, in multiscale denoising methods using wavelets, the deterministic components of the data (e.g., spikes, trends, and deviation in mean/variance) are represented by a small number of coefficients of a wavelet with a higher magnitude, whereas the random components of the signal are captured by the remaining coefficients [22,46,63,64]. This desirable characteristic of wavelets is used to denoise and detect features of the signals. Thus, the extraction of the stationary Gaussian noise can be performed by the following three steps [46]:

- Transforming the raw data to the time–frequency domain by decomposing the input data, using a chosen set of orthonormal wavelet functions.
- Ignoring coefficients that are smaller than the fixed threshold value.
- Converting back the selected coefficients into the original time domain.

In [64], it has been demonstrated that for the denoised signal of a noisy data with length n there is an error within $O(\log n)$ between the noiseless data and the denoised data with a priori information about the smoothness of the underlying

https://t.me/learningnets

data. Therefore, the selection of an appropriate threshold represents a core step for efficient denoising. Various thresholds have been developed in the literature including universal threshold, soft and hard thresholding, and adaptive thresholding rules [64,65]. There are many methods available in the literature for this purpose [65]. Generally speaking, in hard thresholding, the values of the coefficients exceeding the threshold are kept, whereas in soft thresholding they are set to zero. It should be noted that hard thresholding can lead to a larger variance in the reconstructed signal, with infrequent artifacts. However, they are suitable for representing discontinuities and peaks. On the other hand, soft thresholding, which results in a larger bias, provides an adequate visual quality of denoising. Bruce and Gao [66] introduced a threshold to get a compromise between variance and bias, using two threshold values and requiring more computation. Practically, the universal threshold is mainly based on the input measurements through the σ_j estimate. Indeed, for large samples, this threshold is efficient to reduce noise; however, it can remove a portion of the underlying deterministic signal. To bypass this limitation, Donoho and Johnstone [65] proposed keeping coefficients of the first j_0 coarse scales irrespective of whether they exceed the decision threshold or not. However, the selection of j_0 impacts the MSE, and the selection should be based on the smoothness of the underlying data [58,67–69].

The Visushrink method can be used to get better quality of the denoised signal [65],

$$t_j = \sigma_j \sqrt{2 \log n}, \tag{5.7}$$

where σ_j represents the standard deviation the errors at scale j and n is the length of the signal. Usually, the value of σ_j is estimated based on the wavelet coefficients at that level based on the following formula:

$$\sigma_j = \frac{1}{0.6745} \text{median} \left\{ \left| d_{jk} \right| \right\}. \tag{5.8}$$

Numerous methods were introduced to find the value of the threshold and can be found in [70].

5.3.3 Advantages of multiscale denoising

The single scale denoising techniques are highly ranked due to their use and are simple in computation methods. Once proper tuning of the single scale filtering method has been achieved, it can then be used to identify the small shift in the mean. Thus, they are widely applied in statistical monitoring systems. The main drawback of single-scale filter parameters is their deficiency to adapt to the nature of the signal. If the signal contains features at a multiscale level, single-scale filters are strained for a trade-off of the extent of error removal with the quality of the maintained local features. Furthermore, if the signal errors are nonstationary or time-varying, single-scale rectification methods then fail

https://t.me/learningnets

to decorrelate the error signal. In addition to the above disadvantage of single-scale filters, linear filters are not robust for non-Gaussian such as outliers in the signal.

5.4 Wavelet-based multiscale univariate monitoring techniques

After filtering noise from the data, various fault detection charts can be used for detecting abnormal features in the data. Fault detection charts can be categorized into two groups that use single and multiscale methods. Multiscale-based fault detection methods have been developed for both univariate and multivariate process variables. Here, we discuss how DWT can be used for denoising and for fault-detection in univariate process data. There are a few different ways of using DWT for univariate fault detection and diagnosis [58]. Overall, the core idea is to apply DWT for signal decomposition and then check individual details and the approximated signal. Other techniques focus on monitoring the coefficients (see, e.g., [71]), while some others inspect the reconstructed approximation and details [72]. Here, we provide a general framework integrating univariate monitoring charts, such as Shewhart, EWMA, and GLRT, that include desirable features of the wavelet decomposition.

Merging the desirable features of a multiscale representation with conventional univariate monitoring schemes, such as Shewhart, CUSUM, and EWMA, lead to improved detection performance. The general framework of multiscale univariate monitoring schemes is illustrated in Fig. 5.8. The core principle of this methodology is to first decompose the data using a multiscale representation and to apply the selected monitoring scheme to detailed signals. Specifically, we use a multiscale representation to decorrelate the autocorrelated features of data. Then, conventional monitoring schemes are applied to verify the obtained coefficients at each level. Here, decision thresholds are determined at multiple scales based on data representing a normal operating mode. These thresholds are used

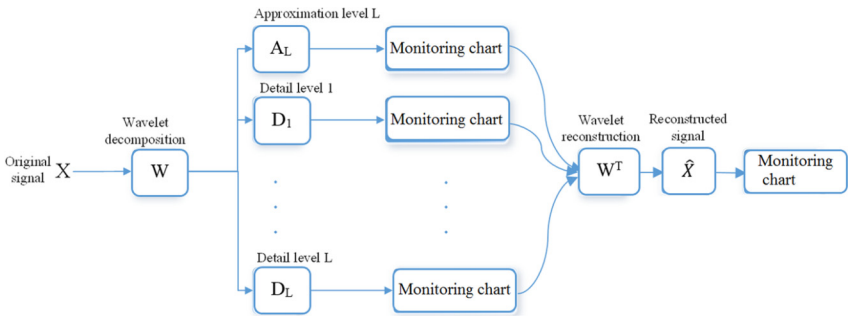


FIGURE 5.8 A general framework of univariate multiscale monitoring.

https://t.me/learningnets

to monitor detailed signals of new data, and the fault is flagged when the thresholds are exceeded. In the reconstruction phase, only coefficients exceeding the thresholds are used, whereas those below the threshold are set to zero. Lastly, the reconstructed signal is monitored using the monitoring scheme. The multiscale monitoring approach is schematically presented in Fig. 5.8. In this chapter, we use the widely used multiscale monitoring framework [73]. Some methods reported in the literature focus on monitoring the details and approximations, using statistical monitoring schemes, such as Shewhart and EWMA [58,74]. One method only monitors the reconstructed signal from the filtered wavelet coefficients, at all levels, along with the scaling coefficients [58,75].

To simplify, we only present here the main steps to implement a multiscale EWMA scheme. In a similar way, other univariate charts (i.e., GLR, CUSUM, and Shewhart) can be performed. The diagrammatic illustration of multiscale EWMA is given in Fig. 5.9, and the main steps to perform multiscale EWMA are sketched next.

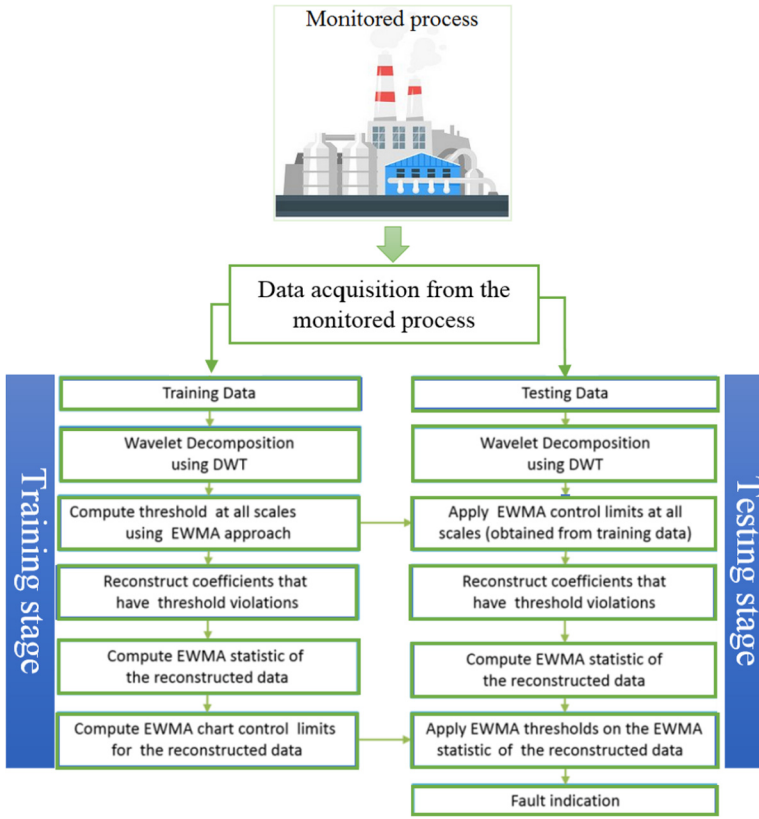


FIGURE 5.9 Main steps for implementing the multiscale EWMA algorithm.

https://t.me/learningnets

1. **Training phase**
   - **Step 1** Gather fault-free measurements representative of nominal operating conditions, which are required for computing the decision limits
   - **Step 2** Scale both the training and testing data to zero mean and unit variance
   - **Step 3** Transform the input data to multiple scales set by decomposing the scaled data into wavelet coefficients
   - **Step 4** Calculate the EWMA statistic and the control limits
   - **Step 5** Maintain only scales violating the decision thresholds and use them to reconstruct the data
   - **Step 6** Calculate the control limits for the reconstructed data
2. **Testing phase**
   - **Step 1** Transform the testing data into multiple scales by applying DWT
   - **Step 2** Calculate the EWMA statistic and apply the detection limits for each scale computed in the training phase
   - **Step 3** Select only coefficients that overpass the detection threshold
   - **Step 4** Reconstruct data in the time dome using the selected coefficients
   - **Step 5** Flag an anomaly when the EWMA statistic overpasses the detection limits

Of course, in multiscale monitoring of a univariate process, the input single-scale data is decomposed into coefficients at every scale, then the details are monitored using univariate charts, as discussed above. After reconstructing the data in the time domain, using only the coefficients violating the detection threshold, the convectional charts are applied to monitor the reconstructed data.

5.4.1 An illustrative example

To illustrate the benefits of using multiscale monitoring charts, we compare the detection performance of the multiscale Shewhart scheme with its conventional counterpart under different conditions (i.e., for autocorrelated, non-Gaussian, and noisy data). The criteria used for comparison are the missed detection rate and the false alarm rate.

5.4.1.1 Impact of autocorrelated data on the conventional Shewhart chart

The aim of this example is to demonstrate the capacity of a wavelet-based monitoring scheme to monitor autocorrelated process data. To this end, here we compare the performance of a single scale and multiscale Shewhart schemes in detecting process mean faults in the autocorrelated data. To show how this impacts the statistical performances on the two charts, first the AR(1) model is used to simulate 500 fault-free data samples, which are used to compute control limits. The control limits computed based on the training data are then used for detecting potential faults in the unseen testing dataset. Second, the testing

https://t.me/learningnets

dataset, which is generated via the same AR model, consists of 500 data samples. An additive bias fault is added to the testing data from samples 200 to 300. This fault is represented by a constant bias of amplitude equal to 3. We repeated this simulation 5000 times for different values of the AR-parameter taken between 0.3 and 1. The averages of missed detection rate and false alarm rate of the two charts as functions of the first-order autoregressive AR(1) parameter, a, are given in Fig. 5.10.

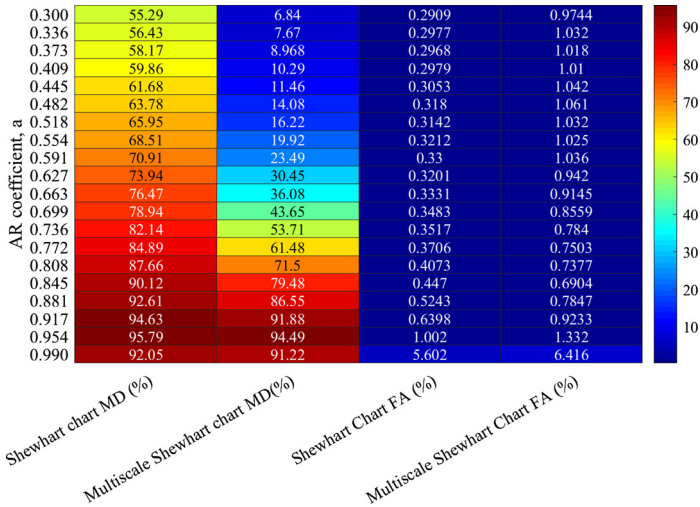


FIGURE 5.10 Detection performance of Shewhart and multiscale Shewhart schemes in the case of autocorrelated data.

In this comparative study, in all figures, the Shewhart chart the missed detection rate is represented by a light-blue solid line with diamonds; the multiscale Shewhart chart missed detection rate is illustrated by a bold dashed line with diamonds; the Shewhart chart false alarms rate is shown by a light-green solid line with circles, and the multiscale Shewhart chart false alarm rate is in bold-green solid line with circles. Fig. 5.10 indicates that the performance of both schemes depends highly on the degree of autocorrelation in the data. We observe that the increase in the degree of autocorrelation negatively impacts on the detection performance of these two schemes by increasing the number of missed detections. We also notice that the detection performance of the multiscale Shewhart chart is more robust than that of the conventional Shewhart chart because the missed detection rate of the multiscale version of Shewhart chart is much lower than the conventional one for almost all values of correlation parameters, except for extremely high degrees of autocorrelation where the performance of both charts is comparable. The multiscale Shewhart scheme provides a clear improvement over the conventional Shewhart chart by decreasing missed detections. In summary, the multiscale Shewhart scheme, which integrates the conventional Shewhart chart with the advantages of a multiscale representation

https://t.me/learningnets

of data, outperforms the conventional Shewhart scheme for detecting faults in autocorrelated time-series data.

5.4.1.2 *Effect of measurement noise on the conventional Shewhart chart*

Here, the detection efficiency of the conventional Shewhart chart and its multiscale version are investigated when the data from the monitored process is noisy. To do so, we generate 500 random Gaussian data samples, which are being used to determine the control limits of the two schemes. The testing dataset consists of 500 data samples that are completely independent of the training data and contaminated with bias fault of amplitude equal 2 introduced from samples 200 to 300 and from samples 400 to 450 of the testing data. This simulation is replicated 5000 times for different measurement noise levels taken from $\sigma = 0.03$ to $\sigma = 2$. Fig. 5.11 displays the average value for 5000 missed detection rates and false alarm rates as a function of the standard deviation of the measurement noise. In Fig. 5.11, we observe that the detection capacity decreases by increasing the missed detection rate when the noise level increases. This case study confirms the advantages of the multiscale Shewhart scheme over the single scale Shewhart scheme when applied to noisy data (Fig. 5.11). This is mainly due to the capacity of the multiscale representation to separate relevant and irrelevant features in the data. Thus, combining the advantages of a multiscale representation with those of a univariate monitoring Shewhart chart, it should be possible to obtain further improvements in fault detection.

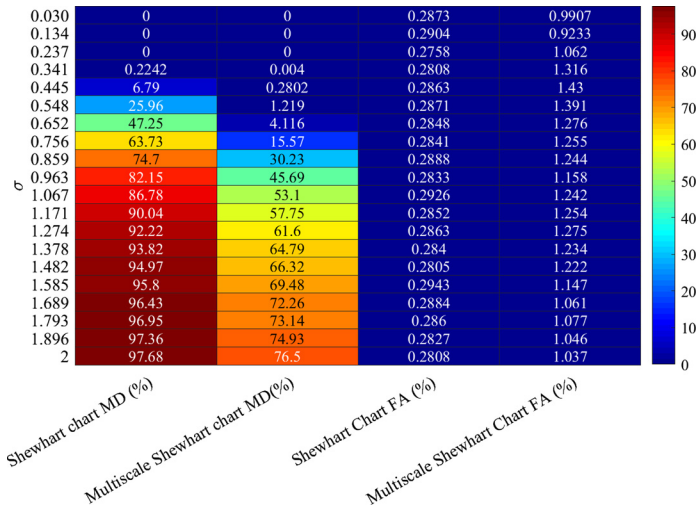


FIGURE 5.11 Comparison of the performance of Shewhart and multiscale Shewhart monitoring charts for the case of noisy data.

https://t.me/learningnets

5.4.1.3 Impact of the violation of normality assumption on the conventional Shewhart chart

In this case study, we investigate the ability of wavelet-based monitoring to deal with non-Gaussian data. Specifically, the detection performance of single and multiscale Shewhart schemes are compared when the normality assumption underlying the univariate monitoring charts is violated. In other words, we investigate the impact of violating the normality assumption on the conventional Shewhart monitoring chart. The statistical performance of the two schemes is investigated using the chi-square data, with different values of a fourth-order cumulant (or kurtosis). The kurtosis is used in this study to quantify the non-Gaussianity in the data. For univariate data $x_1, x_2, \ldots, x_N$, the formula for kurtosis is computed as

$$\text{Kurtosis} = \frac{\sum_{t=1}^{N}(x_i - \mu)^4}{(N-1)\sigma^4}, \tag{5.9}$$

where μ is the mean, σ is the standard deviation, and N is the number of data points. In fact, kurtosis quantifies the degree of peakedness of a distribution; it is 3 for a Gaussian distribution. Generally speaking, if it is greater than 3, the distribution is termed "super-Gaussian", i.e., spikier than Gaussian, and if it is less than 3, such distribution is termed "sub-Gaussian", i.e., flatter than Gaussian. To investigate the detection performance when the Gaussianity assumption is invalid, 1024 random fault-free data samples with a chi-square nonnormal distribution were generated. After computing detection limits of the two charts using the training data, another testing dataset of 1024 random samples with a chi-square distribution was generated. A step anomaly is then injected into the testing data between samples 201–250, 501–525, and 701–725. The magnitude of the considered step fault is 3σ. For different values of the kurtosis taken between 3 and 14, each simulation is repeated 5000 times to get more accurate results. The means of the 5000 missed detection rates and false alarm rates as functions of kurtosis are presented in Fig. 5.12. It shows that the missed detection rate significantly increased as the departure from Gaussianity increased (i.e., when we increased the kurtosis of data) while false alarm rate increased slightly when the kurtosis increased. Fig. 5.12, which shows the average of missed detection rate and false alarms rate as a function of kurtosis of these two charts, clearly illustrates the advantages of the multiscale Shewhart chart over the conventional chart. This case study shows that, in most instances, the performance of the multiscale Shewhart chart is superior to that of the conventional Shewhart chart in detecting process mean faults.

In essence, the wavelet-based univariate monitoring approaches offer several advantages compared to single scale monitoring schemes including: (i) decorrelating time-dependent data while keeping a Gaussian distribution; (ii) obtaining Gaussian wavelet coefficients at each scale even with non-Gaussian input data; (iii) separating noise from relevant features in the data; and (iv) compacting the deterministic features in a small number of coefficients.

https://t.me/learningnets

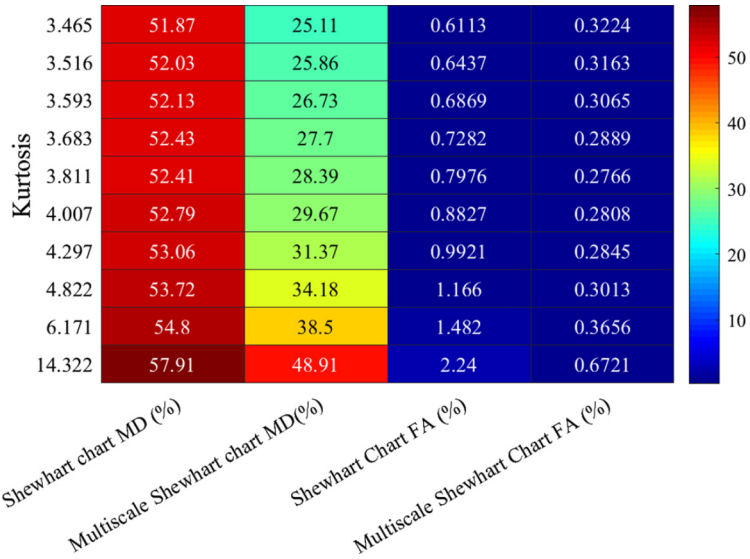


FIGURE 5.12 Comparison of the performance of Shewhart and multiscale Shewhart monitoring charts for the case of non-Gaussian data.

5.5 Multiscale LVR modeling

5.5.1 Benefits of multiscale denoising in LVR modeling

Most of the real-world data are multiscale in nature; therefore, a model estimation using such data needs to consider the multiscale modeling methods that take into account multiscale filtering, using wavelets to increase the efficacy of the latent variable regression models. In this section, we highlight the major benefits of using multiscale filtering in LVR modeling [76].

- The presence of noise in the measured data degrades the model parameter estimation of the LVR models. The measured data are passed through low- and high-pass filters constructed from orthonormal basis functions, and subsequently reduce the noise embedded in the measurements and results, yielding in a more precise prediction.
- Most of the measured data contain noise that are correlated in nature, because the source of noise is not independent and random, like the malfunctioning or error in sensor calibration, for example. Model parameter estimation becomes challenging with the presence of correlated data. Using wavelets for representing correlated data makes it decorrelated. This property is very helpful where measurement errors are not always random [24].

The model is developed by making use of the above advantages, which integrate the LVR model parameter estimation and multiscale filtering to improve model prediction.

https://t.me/learningnets

5.6 Multiscale LVR modeling

In practice, industrial processes generate multiple time series that need to be monitored. Monitoring these process variables individually does not take into account the interrelations among the variables and faults that might occur in these relationships. Furthermore, the information in the multivariate data is important for early detection and high detection performance. Hence, in this section a wavelet-based technique to monitor multivariate processes will be discussed.

The desirable features of wavelet-based multiscale representation are used to monitor multivariate processes. Indeed, the majority of monitoring methods, such as PCA and PLS, employ a single-time scale and are designed without consideration of the multiscale properties of the data. Moreover, most data from industrial processes frequently contain relevant features and measurement noise associated with both time and frequency. Also, several existing techniques assume uncorrelated input data, even though, in practice, time-dependent data are very common. Thus, combining wavelets and multivariate monitoring methods (e.g., LVR techniques) results in improved detection performance. Various multiscale monitoring methods have been applied in the literature for enhancing the prediction quality and the robustness of monitoring strategies [34,58,77,78]. For instance, Bayesian multiscale monitoring techniques need prior information about the nature of the occurred anomalies [79]. However, these methods are not frequently applied in industrial applications because of a lack of the required prior information [80]. Several methods have been designed in the literature based on information extracted from scale-based analysis using wavelets. In [81], an approach is introduced for defect detection in rotating equipment by clustering the wavelet coefficients calculated from the measured process data. In [82], a monitoring method is proposed to detect bearing anomalies of rotating equipment. This method generates features from the wavelet coefficients and then applies ANFIS to uncover bearing anomalies. In [83], the features generated from DWT are used as input to a neural network for anomaly detection and diagnosis of a gearbox system.

For multivariate process monitoring, multiscale filtering of the raw data improves the FD performance of the conventional PCA method [84]; the multiscale PCA (MSPCA) algorithm from [34] that constructs multiple PCA models using the wavelet coefficients at different scales also shows better monitoring abilities. Indeed, multiscale PCA allows the removal of time-dependence by using wavelet decomposition and employs PCA to remove the cross-correlation between process variables. In MSPCA, each process variable is first decomposed using DWT, the details coefficients are obtained at each level from the thresholded coefficients, and then PCA of the details of all variables is performed for every level. The PCA scores are then supervised via T^2 and SPE charts to uncover the significant scales that indicate process abnormalities. Generally speaking, anomalies can be observed at the final levels, and if they persist, the coarser levels are able to sense them, too. The scaling coefficient of the coarsest

https://t.me/learningnets

level is less sensitive to change because it is generated last in the decomposition procedure. To alleviate this detection delay, the multiscale techniques generally control the reconstructed data. After reconstructing data in the time domain, PCA is applied to monitor the data, with T^2 and SPE schemes used respectively to monitor the principal components and residual subspaces.

Also, several latent variable methods that use the synergy between the abilities of the LVR models are applied to describe the cross-correlation between the input–output process variables, and the DWT capability for decorrelating time-dependent processes and separating relevant features embedded in noisy data is also used [85,86]. In [87,88], multiscale PLS models are developed for modeling and fault detection. The PLS model is designed using the denoised data obtained after eliminating the low-frequency scales characterizing low-frequency components. However, multiscale PCA and PLS are designed based on a linearity assumption, which limits their applications. To bypass this limitation, several multiscale nonlinear methods are used, such as multiscale KPCA and KPLS methods [89]. Specifically, in these methods, KPCA and KPLS are used to describe process variable correlations at different levels. Comprehensive studies of MSPCA and multiscale LVR can be found in [33,87,89,90].

Data collected in many environmental and engineering processes exhibit complex structures and violate the basic assumptions of conventional approaches. Therefore, the need to design methods able to handle complex structures of data is required to overcome some of the limitations of the conventional approaches. Here, we present the basic idea behind the multiscale latent variable regression models. The essence behind multiscale LVR models is to amalgamate the benefits of multiscale denoising and LVR models parameter estimation for extracting maximum information from multivariate data, improve the prediction quality and thus enhancing fault detection. This integrated framework makes it possible to handle complex data structures frequently generated by environmental and engineering processes (e.g., multivariate input–output and multiresolution features), and for representing and analyzing data at different resolutions and time scales. Let the observed input–output data be $\mathbf{X}$ and $\mathbf{y}$, and their multiscale filtered counterpart at a particular scale (p) be $\mathbf{X}_p \in \mathbb{R}^{n\times m}$ and $\mathbf{y}_p \in \mathbb{R}^{n\times 1}$, the LVR model based on the filtered data is obtained as

$$\mathbf{y}_p = \mathbf{X}_p \mathbf{b}_p + \epsilon_p, \tag{5.10}$$

where $\mathbf{b} \in \mathbb{R}^{m\times 1}$ is the estimated model parameter vector, and $\epsilon_p \in \mathbb{R}^{n\times 1}$ is the model error when using the denoised data at scale (p).

It should be noted that when filtering the data, using wavelets without considering the relationship between the input–output may result in eliminating important features that are crucial to the model construction. Thus, a multiscale filtering algorithm must be combined with LVR model parameter estimation to proper noise denoising. This can be achieved by the following steps schematically illustrated in Fig. 5.13:

https://t.me/learningnets

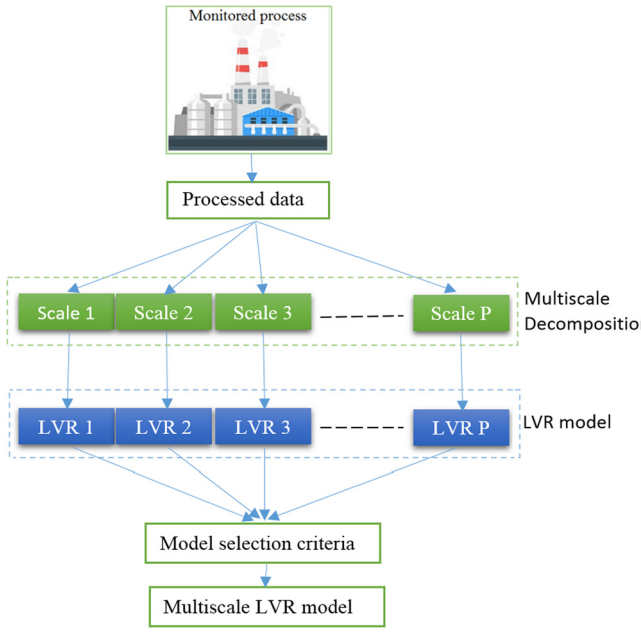


FIGURE 5.13 Conceptual schematic of multiscale LVR models.

- Split the given raw measurements in training and testing datasets
- Scale the data to have zero mean and unit variance
- Filter the input–output training data at different scales via wavelets as per the algorithm presented in Fig. 5.13
- Develop an LVR model based on the filtered training data at each scale
- Apply the cross-validation method to select the optimum number of principal components for the LVR model at each scale
- Validate the estimated model using unseen data set and calculate the mean squared error
- Choose the LVR model which gives the lowest cross-validation error

The multiscale LVR model is first constructed using anomaly-free measurements and then used for process monitoring. As discussed above, in multiscale LVR methods, each time series data is first decomposed, based on wavelet decomposition, and an LVR model is then constructed separately using the coefficients from all series at each scale (Fig. 5.13). Within the context of multivariate process monitoring based on multiscale LVR methods, two popular multivariate statistical schemes, the Hotelling T^2 and Q statistics, are generally used to monitor LV and residual subspaces, respectively. However, these two monitoring schemes are ineffective in sensing small changes and may lead to unreliable process monitoring [88]. As in single scale methods, it is also needed to uncover incipient changes. Several methods have been proposed to address this problem

https://t.me/learningnets

by combining the multiscale LVR models with sensitive detectors, such as GLR test and EWMA [88].

5.7 Results and discussions

5.7.1 Application with synthetic data

Now, the prediction capability of the above multiscale LVR models is assessed and compared with both conventional LVR modeling models and models obtained from single-scale filtering methods. The comparison is performed on two case studies based on simulated synthetic data and simulated distillation column data. The model parameters are optimized for both case studies, using a cross-validation approach; the mean squared error (MSE) is reported, using testing data set, and computed as follows:

$$\text{MSE} = \frac{1}{N}\sum_{k=1}^{n}(y(k) - \hat{y}(k))^2, \tag{5.11}$$

where $\hat{y}(k)$ is model prediction, $y(k)$ is the gathered output at time instant k, and n is the length of testing samples.

5.7.1.1 Simulation results: synthetic data

Here, simulated data is used for model comparison. The same dataset has been taken for the study in [76]. Fig. 5.14 displays a sample of the output data, where SNR = 10.

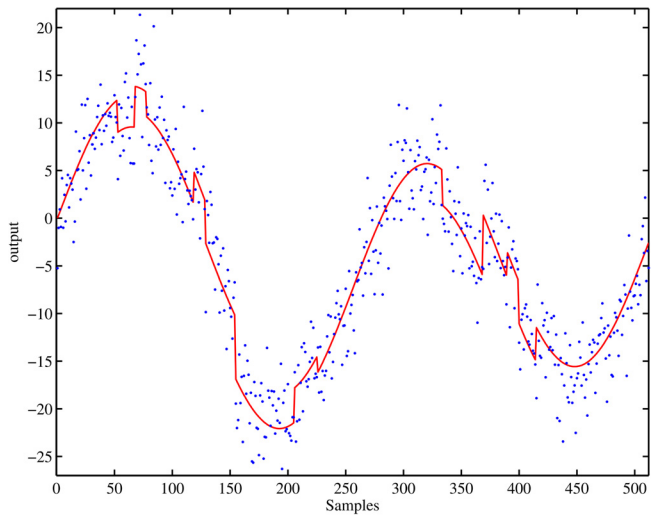


FIGURE 5.14 Output measurements with SNR = 10, where noisy data are presented using dots, while the solid line is used to represent noise-free measurements.

https://t.me/learningnets

In this section, we present the results obtained from multiscale, single-scale, and conventional LVR models, and compare them with the different modeling approaches with respect to their MSE values. We use a Daubechies wavelet filter with order 3 and a cross-validation method to optimize the parameters required for the model. To establish statistically valid conclusions, a Monte Carlo simulation of 1000 repetitions is carried out, and final results are tabulated in Table 5.1. Results in Table 5.1 show that modeling through EWMA filtering (EWMA+LVR) and mean filtering (MF+LVR) improves the performance of prediction over the conventional LVR models. The MSLVR model offers significant improvement, in comparison with all the investigated modeling algorithms, for all noise levels [76].

TABLE 5.1 The achieved MSE from the different studied modeling schemes.

Model type	MSLVR	EWMA+LVR	MF+LVR	LVR
	SNR = 5			
PLS	0.9512	1.4562	1.6106	3.6568
PCR	0.9586	1.4504	1.6101	3.6904
	SNR = 10			
PLS	0.5930	0.9325	1.0239	1.8733
PCR	0.6019	0.9211	1.0240	1.8876
	SNR = 20			
PLS	0.3928	0.5994	0.6733	0.9423
PCR	0.3946	0.5872	0.6670	0.9508

In our study, the Daubechies wavelet filter of order 3 is employed in multiscale filters, and using the cross-validation technique all studied filters are optimized. To assess the efficiency of the studied methods and guarantee valid conclusions, 1000 realizations were done and the average results are displayed in Table 5.1. They indicate that LVR modeling based on EWMA filtering (EWMA+LVR) and mean filtering (MF+LVR) achieves an important enhancement compared to the conventional LVR models (Table 5.1). Also, results show that a superior performance has been achieved when using the MSLVR algorithm over all the modeling algorithms for all considered noise levels [76].

Fig. 5.15 shows the prediction performance of the four models being studied when SNR = 10. Results show the superior prediction capability of MSPLS over the other single-scale LVR modeling strategies. It should be noted that the selection of a proper wavelet filter has a greater effect on the prediction ability of the multiscale model, which is generally related to the type of data. This result shows the limitation of conventional LVR models, and better prediction results can be obtained by integrating the wavelet-based multiscale representation with LVR models.

https://t.me/learningnets

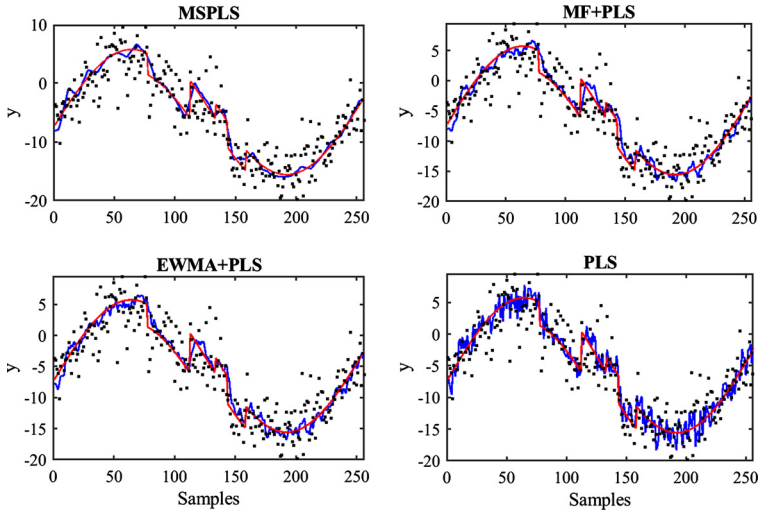


FIGURE 5.15 Comparison of the MSPLS model prediction along with other three modeling techniques using synthetic data with SNR = 10: model prediction (solid blue [dark gray in print version] line), noise-free measurements (solid red [mid gray in print version] line), and noisy measurements (blue dots).

5.7.1.2 Simulation results: distillation column

In the second case study, distillation column data are considered to assess and compare the multiscale LVR method (PLS and PCR) with six single-scale LVR models. To give a clear picture of the perturbation data with SNR = 10, training and testing data (Fig. 5.16) are used to develop the considered models. These perturbations (in the training and testing data) are displayed in Fig. 5.16E–H. Here, we only present the results obtained from different modeling algorithms.

In this simulation study, ten temperatures measured at different locations of the column along with flow rates of feed and reflux streams are considered as the input matrix (input data set). The compositions of the light component (propane) in the distillate and residue are considered as the output variables (i.e., x_D and x_B,). The prediction capability of the MSLVR modeling framework is then compared to the conventional LVR model and models estimated using single-scale filtered data. Results are displayed in Tables 5.2 and 5.3 for the prediction of x_D and x_B compositions. Results clearly show that modeling with single-scale filtering can enhance the prediction accuracy of the LVR models. It is also observed that filtering the multiscale approach further improves the prediction capability of the LVR model. It is also to be noted that the performance of the MSLVR prediction increases for a lower signal-to-noise ratio. The predictions of the top composition (x_D), using MSPLS and PLS-based models, when SNR = 10, are shown in Fig. 5.17, illustrating that the PLS model with multiscale filtering is superior to other LVR modeling schemes.

https://t.me/learningnets

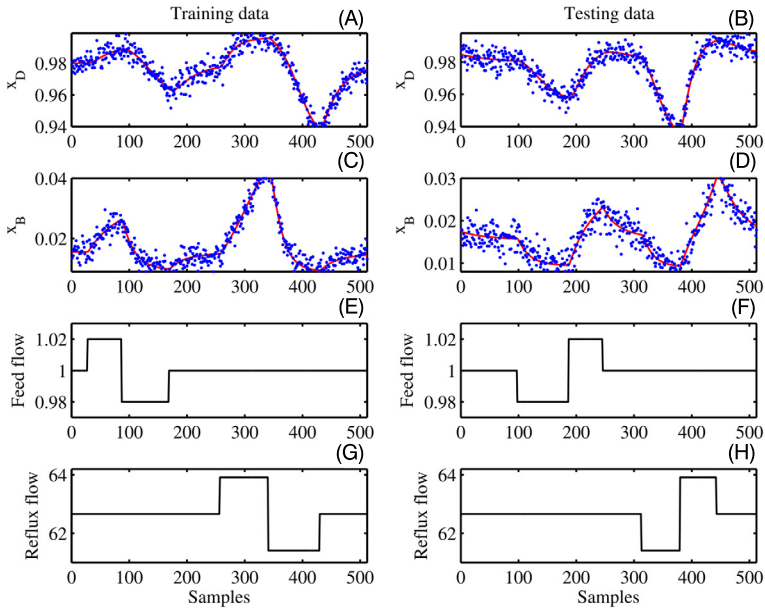


FIGURE 5.16 Input–output measurements used to train and test the model, where noisy data are presented using dots, while the solid line is used to represent noise-free measurements.

TABLE 5.2 Prediction quality of the considered models: average MSEs for x_D.

Model type	MSLVR	EWMA+LVR	MF+LVR	LVR
$\times 10^{-4}$	SNR = 5			
PLS	0.0202	0.0288	0.0303	0.0984
PCR	0.0204	0.0288	0.0357	0.0983
$\times 10^{-5}$	SNR = 10			
PLS	0.1340	0.1790	0.1891	0.5388
PCR	0.1317	0.1778	0.1879	0.5423
$\times 10^{-5}$	SNR = 20			
PLS	0.0844	0.1130	0.1218	0.3017
PCR	0.0801	0.1112	0.1200	0.3040

5.7.2 Application of monitoring distillation column

To show the benefits of multiscale LVR methods in supervising multivariate input–output systems, we compare the multiscale PLS (MSPLS)-based GLR method with the single scale PLS-based monitoring techniques, in order to show the benefits of combining the desirable features of wavelet-based representation, the flexibility of the PLS model, and the greater sensitivity of the GLR test to uncover changes. Specifically, the MSPLS model is constructed to fit

https://t.me/learningnets

TABLE 5.3 Prediction quality of the considered models: average MSEs for x_B.

Model type	MSLVR	EWMA+LVR	MF+LVR	LVR
$\times 10^{-5}$	SNR = 5			
PLS	0.0331	0.0702	0.0725	0.1979
PCR	0.0327	0.0708	0.0736	0.1961
$\times 10^{-5}$	SNR = 10			
PLS	0.0212	0.0448	0.0468	0.1063
PCR	0.0207	0.0444	0.0466	0.1063
$\times 10^{-6}$	SNR = 20			
PLS	0.1224	0.2785	0.2956	0.5676
PCR	0.1183	0.2736	0.2914	0.5703

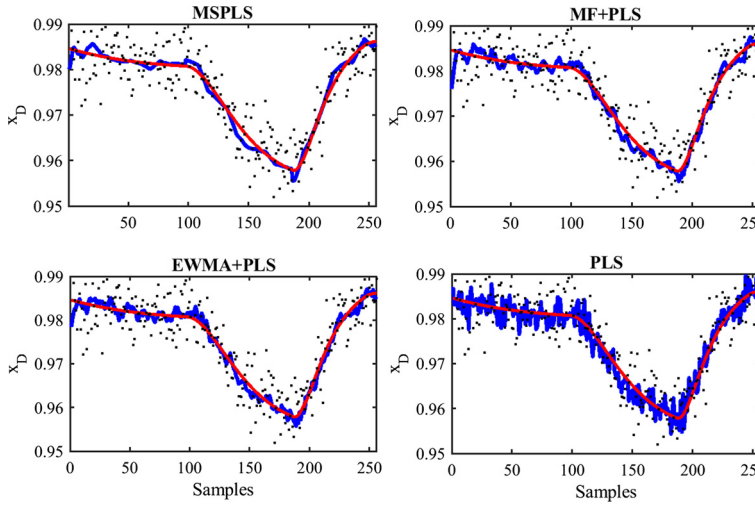


FIGURE 5.17 MSPLS and PLS model-based predictions of x_D with SNR = 10: model predictions (solid blue [dark gray in print version] line), noise-free measurements (solid red [mid gray in print version] line), and noisy measurements (blue dots).

the process data, and the GLR scheme used to monitor the generated residuals from the MSPLS model. A simulated distillation column data is used to verify the detection capacity of the MSPLS approaches and conventional PLS-based techniques. Fault-free data are generated using the Aspen simulator and used to design the MSPLS model. The parameters of the nominal conditions used in the simulated distillation column can be found in [88]. Here, only three LVs are maintained in the MSPLS model based on the cross-validation method. Three types of anomaly are considered here: bias sensor anomaly, intermittent anomaly, and drift anomaly. The MSPLS model is constructed using fault-free data, and the quality of the fit is represented in Fig. 5.18.

https://t.me/learningnets

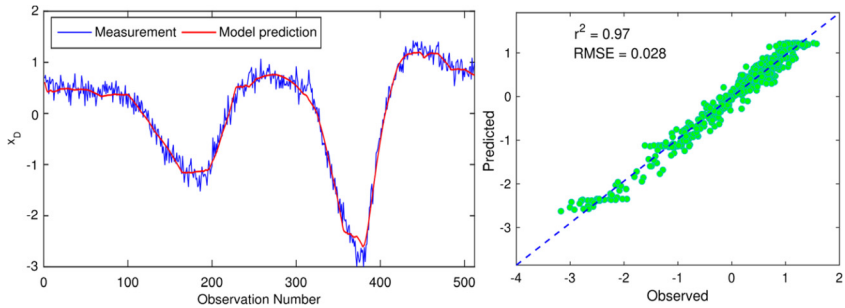


FIGURE 5.18 (Left) Measured and MSPLS estimated based on training measurements with SNR = 10. (Right) Scatter graph of observed against estimated values from the fitted MSPLS model.

Now, two different sensor faults (summarized in Fig. 5.19) are considered to assess the performances of the different FD techniques. In this study, the evaluation of the detection performance of the PLS and MSPLS is carried out by using the false alarm rate (FAR) and missed detection rate (MDR).

Fault Type	Model	Parameters
Bias Fault	$Tc_3(t) = Tc_3(t) + b \cdot I(t \in \mathcal{I})$	b: shift in temperature Tc_3 for $t \in \mathcal{I}$
Drift Fault	$Tc_3(t) = Tc_3(t) + st \cdot I(t > t_0)$	s: slope of the drift for the starting point t_0

FIGURE 5.19 Anomalies considered in the simulation study.

Here, the feasibility of the MSPLS-based approach is verified to sense atypical abrupt changes in distillation column data. First, to simulate the abrupt congestion, a small bias of 2% of the total variation in temperature, Tc_3, is added in the raw measurements, for sample ranging from 100 to 150. For data with SNR = 5, the performance of the PLS and MSPLS-based Q and GLR methods is shown in Fig. 5.20A–D, where dashed lines represent a 95% confidence interval used to identify the possible faults. The PLS-based Q and GLR approaches provide poor results and are unable to detect this fault. Results show a superior efficiency of the MSPLS-GLR approach, compared with conventional PLS-based Q and GLR and MSPLS-Q algorithms. The MSPLS-Q detects this fault but with FAR = 10.43% and MDR = 9%. However, the MSPLS-GRT approach results in clear detection without false alarms.

The purpose of the second scenario is to analyze the ability of the MSPLS-based approach to detect changes. To this purpose, a gradual anomaly is simulated by injecting drifting with a slope of 0.01 in the temperature sensor, Tc_3, from sample number 250. Fig. 5.21 displays the results obtained with the four designed methods. Results highlight that the detection capability is improved by using the MSPLS combined with the GLR approach. This is mainly due to the flexibility and efficiency of the MSPLS model in capturing relevant features in the data and generating sensitive residuals and the good ability of the GLR approach to uncover abnormal changes.

https://t.me/learningnets

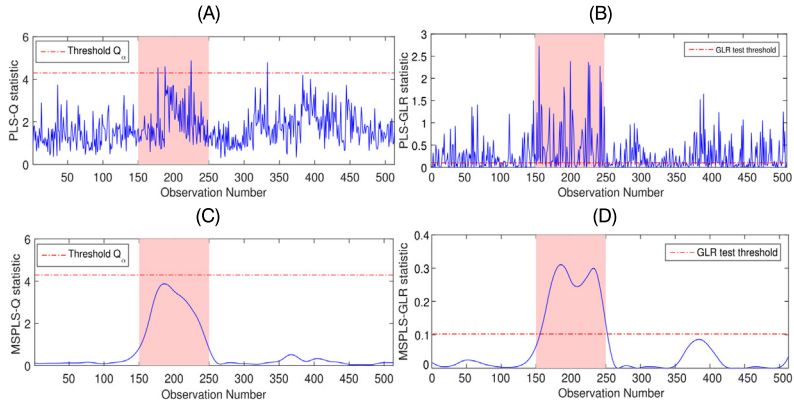


FIGURE 5.20 Results from (A) PLS-Q, (B) PLS-GLR, (C) MSPLS-Q, and (D) MSPLS-GLR detectors when a bias anomaly has happened in Tc_3 (SNR = 5).

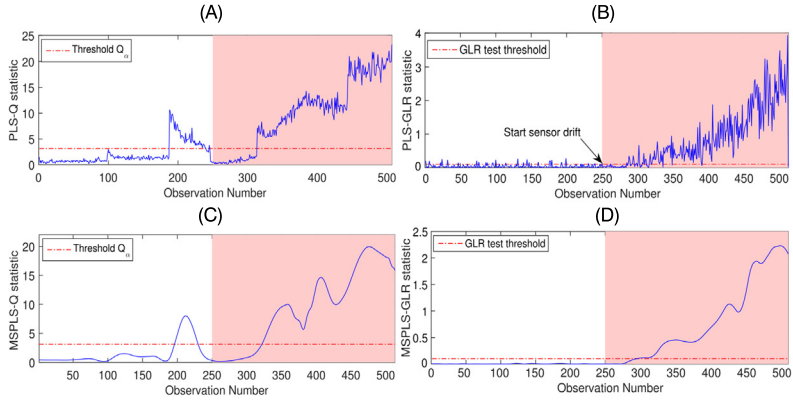


FIGURE 5.21 Results from (A) PLS-Q, (B) PLS-GLR, (C) MSPLS-Q, and (D) MSPLS-GLR detectors when gradual anomaly has happened in Tc_3 (SNR = 30).

Overall, this example shows that the MSPLS-based GLR has good capability in identifying abrupt and progressive anomalies. Also, results show the superior efficiency of the MSPLS-GLR approach compared with the conventional PLS approaches (Q and GLR). This is, in large, because of the fact that the MSPLS integrates the desirable features of wavelet-based representation with PLS, resulting in an extension of the detection performance of GLR, in comparison with the use of the single-scale PLS model.

5.8 Discussion

Early detection of possible failure in complex engineering systems or monitoring networks has proven to be particularly challenging. Conventional fault detection methods, such as EWMA, CUSUM, and GLR, and their integrated

https://t.me/learningnets

versions including the latent variable regression-based EWMA and the CUSUM methods, have shown greater capacity for fault detection, especially for small faults. These conventional monitoring schemes are designed on the statistical assumptions that the measurements are uncorrelated in time and that they follow a Gaussian distribution. However, in many industrial and environmental applications, generated measurements are autocorrelated, non-Gaussian, and noisy. Indeed, a high amount of noise in the data can hide the key features that are essential to detecting faults, ultimately deteriorating the fault detection capacity of the monitoring chart. Thus, conventional schemes are unsuited for monitoring autocorrelated or non-Gaussian and noisy processes. It has been shown in Sect. 5.4.1 that the effectiveness of fault detection in Shewhart charts is degraded when some or all of its basic assumptions are violated, such as the deviation from normality, autocorrelation in the data, and the presence of high noise levels.

This flexible wavelet-based modeling approach has been shown to be appropriate for autocorrelated, non-Gaussian, and nonstationary processes while capturing important features. The wavelet-based multiscale monitoring framework presented in this paper can efficiently deal with time-dependent and non-Gaussian measurements. Here, it has been shown that the effectiveness of the Shewhart scheme has been improved by integrating the Shewhart fault detection scheme with multiscale wavelets. Most real processes, however, are multivariate in nature. Therefore, in this paper, multiscale LVR-based monitoring schemes have been discussed to handle multiple process variables simultaneously.

It should be noted that the multiscale univariate and multivariate fault detection methods presented in this chapter are batches. In other words, they require the entire data set to be available a priori, which is due to the fact that multiscale representation can only be applied on datasets with a dyadic length, i.e., 2^J, where J is an integer. In real plants, however, measurements are continuously gathered, and in most cases, monitoring of their key variables is required online. Thus, it is necessary for developed multiscale monitoring methods to be extended to handle online processes. Also, the conventional (and multiscale) fault detection methods assume that the data or residuals are stationary, i.e., their characteristics do not change over time. Examples of nonstationarity include changes in the variance of residuals (which can be due to wearing or a change in the sensors used) and changes in the mean of residuals (which can be due to drifts from one operating condition to another). Applying fault detection methods to nonstationary data may not provide acceptable performance. Therefore, adaptive multiscale univariate and multivariate fault detection methods to deal with nonstationary processes are needed.

Furthermore, the multiscale fault detection methods presented in this paper cannot be directly applied to monitor online processes as they require a dyadic set of data a priori. This is a limitation of the wavelet-based multiscale decomposition algorithm used in these methods. Accordingly, developing online multiscale univariate and multivariate monitoring approaches that extend the benefits of these multiscale techniques for processes where online monitor-

https://t.me/learningnets

ing is required. One way to alleviate this challenge is to apply fault detection on a dataset within the moving window so that a fault captured from the most recent data sample can be used as a flag or an indicator of a fault in the process. An advantage of this approach is that it can be used to develop both univariate and multivariate multiscale fault detection methods. Also, from a fault detection point of view, since the moving window keeps the largest number of dyadic samples, a more accurate estimation of the control limits is expected, as more data become available. However, to avoid having a very large window size (which may result in computational burden), the size of the window can be fixed after a large enough window length is reached. Another issue to consider, when using this approach, is the case when a wavelet filter is not Haar. Accordingly, a boundary-corrected version of the filter is required to remedy the problem of inaccuracies at the boundaries. This could be useful because this online multiscale approach relies more heavily on the more recent data samples inside the moving window.

References

[1] L.C. Alwan, Effects of autocorrelation on control chart performance, Communications in Statistics. Theory and Methods 21 (4) (1992) 1025–1049.
[2] L.C. Alwan, H.V. Roberts, Time-series modeling for statistical process control, Journal of Business & Economic Statistics 6 (1) (1988) 87–95.
[3] D.G. Wardell, H. Moskowitz, R.D. Plante, Run-length distributions of residual control charts for autocorrelated processes, Journal of Quality Technology 26 (4) (1994) 308–317.
[4] D. Montgomery, C. Mastrangelo, F.W. Faltin, W.H. Woodall, J.F. MacGregor, T.P. Ryan, Some statistical process control methods for autocorrelated data, Journal of Quality Technology 23 (3) (1991).
[5] G.C. Runger, S.S. Prabhu, A Markov chain model for the multivariate exponentially weighted moving averages control chart, Journal of the American Statistical Association 91 (436) (1996) 1701–1706.
[6] A. Cinar, A. Palazoglu, F. Kayihan, Chemical Process Performance Evaluation, 1st ed., CRC Press, Boca Raton, FL, 2007.
[7] J.N. Dyer, B.M. Adams, M.D. Conerly, The reverse moving average control chart for monitoring autocorrelated processes, Journal of Quality Technology 35 (2) (2003) 139–152.
[8] D.C. Montgomery, Introduction to Statistical Quality Control, John Wiley & Sons, New York, 2005.
[9] E.G. Schilling, P.R. Nelson, The effect of non-normality on the control limits of $\bar{X}$ charts, Journal of Quality Technology 8 (4) (1976).
[10] S.A. Yourstone, W.J. Zimmer, Non-normality and the design of control charts for averages, Decision Sciences 23 (5) (1992) 1099–1113.
[11] P.M. Burrows, $\bar{X}$ control schemes for a production variable with skewed distribution, Journal of the Royal Statistical Society. Series D. The Statistician 12 (4) (1962) 296–312.
[12] B. Laungrungrong, C.M. Borror, D.C. Montgomery, EWMA control charts for multivariate Poisson-distributed data, International Journal of Quality Engineering and Technology 2 (3) (2011) 185–211.
[13] B. Laungrungrong, C.M. Borror, D.C. Montgomery, A one-sided MEWMA control chart for Poisson-distributed data, International Journal of Data Analysis Techniques and Strategies 6 (1) (2014) 15–42.
[14] C. Çiflikli, Development of univariate control charts for non-normal data, PhD dissertation, İzmir Institute of Technology, 2006.

https://t.me/learningnets

[15] I.W. Burr, The effect of non-normality on constants for X and R charts, Industrial Quality Control 23 (11) (1967) 563–569.
[16] B.R. Kowalski, M.B. Seasholtz, Recent developments in multivariate calibration, Journal of Chemometrics 5 (1991) 129–145.
[17] I. Frank, J. Friedman, A statistical view of some chemometric regression tools, Technometrics 35 (2) (1993) 109–148.
[18] M. Stone, R. Brooks, Continuum regression: cross validated sequentially constructed prediction embracing ordinary least squares, partial least squares and principal component regression, Journal of the Royal Statistical Society, Series B 52 (1990) 237–269.
[19] S. Wold, Soft Modeling: The Basic Design and Some Extensions, Systems Under Indirect Observations, Elsevier, Amsterdam, 1982.
[20] E. Malthouse, A. Tamhane, R. Mah, Non-linear partial least squares, Computers & Chemical Engineering 21 (8) (1997) 875–890.
[21] B. Bakshi, G. Stephanopoulos, Representation of process trends IV. Induction of real time patters from operating data for diagnosis and supervisory control, Computers & Chemical Engineering 18 (4) (1994) 303.
[22] B. Bakshi, Multiscale analysis and modeling using wavelets, Journal of Chemometrics 13 (3) (1999) 415–434.
[23] S. Palavajjhala, R. Motrad, B. Joseph, Process identification using discrete wavelet transform: design of pre-filters, AIChE Journal 42 (3) (1996) 777–790.
[24] F. Harrou, M. Madakyaru, Y. Sun, Improved nonlinear fault detection strategy based on the hellinger distance metric: plug flow reactor monitoring, Energy and Buildings 143 (2017) 149–161.
[25] A. Robertson, K. Park, K. Alvin, Extraction of impulse response data via wavelet transform for structural system identification, Journal of Vibration and Acoustics 120 (1998) 252–260.
[26] M. Nikolaou, P. Vuthandam, FIR model identification: achieving parsimony through kernel compression with wavelets, AIChE Journal 44 (1) (1998) 141–150.
[27] M.S. Reis, A multiscale empirical modeling framework for system identification, Journal of Process Control 19 (2009) 1546–1557.
[28] J. Carrier, G. Stephanopoulos, Wavelet based modulation in control-relevant process identification, AIChE Journal 44 (2) (1998).
[29] D.B. Percival, A.T. Walden, Wavelet Methods for Time Series Analysis, vol. 4, Cambridge University Press, 2000.
[30] F. Abramovich, T.C. Bailey, T. Sapatinas, Wavelet analysis and its statistical applications, Journal of the Royal Statistical Society. Series D. The Statistician 49 (1) (2000) 1–29.
[31] S.G. Mallat, A theory for multiresolution signal decomposition: the wavelet representation, IEEE Transactions on Pattern Analysis and Machine Intelligence 7 (1989) 674–693.
[32] R.X. Gao, R. Yan, Wavelets: Theory and Applications for Manufacturing, Springer Science & Business Media, 2010.
[33] M. Misra, H.H. Yue, S.J. Qin, C. Ling, Multivariate process monitoring and fault diagnosis by multi-scale PCA, Computers & Chemical Engineering 26 (9) (2002) 1281–1293.
[34] B. Bakshi, Multiscale PCA with application to multivariate statistical process monitoring, AIChE Journal 44 (1998) 1596–1610.
[35] H.-Y. Gao, Wavelet shrinkage denoising using the non-negative garrote, Journal of Computational and Graphical Statistics 7 (4) (1998) 469–488.
[36] S. Zhou, B. Sun, J. Shi, An SPC monitoring system for cycle-based waveform signals using Haar transform, IEEE Transactions on Automation Science and Engineering 3 (1) (2006) 60–72.
[37] I. Daubechies, Orthonormal bases of compactly supported wavelets, Communications on Pure and Applied Mathematics 41 (7) (1988) 909–996.
[38] C.E. Heil, D.F. Walnut, Continuous and discrete wavelet transforms, SIAM Review 31 (4) (1989) 628–666.

https://t.me/learningnets

[39] F. Harrou, B. Taghezouit, Y. Sun, Robust and flexible strategy for fault detection in grid-connected photovoltaic systems, Energy Conversion and Management 180 (2019) 1153–1166.
[40] A. Zeroual, F. Harrou, Y. Sun, N. Messai, Monitoring road traffic congestion using a macroscopic traffic model and a statistical monitoring scheme, Sustainable Cities and Society 35 (2017) 494–510.
[41] G. Strang, Introduction to Applied Mathematics, Wellesley–Cambridge Press, Wellesley, MA, 1986.
[42] G. Strang, Wavelets and dilation equations, SIAM Review (1989) 613.
[43] X. Li, Real-time detection of the breakage of small diameter drills with wavelet transform, The International Journal of Advanced Manufacturing Technology 14 (8) (1998) 539–543.
[44] X. Li, A brief review: acoustic emission method for tool wear monitoring during turning, International Journal of Machine Tools and Manufacture 42 (2) (2002) 157–165.
[45] S.G. Mallat, A theory of multiresolution signal decomposition: the wavelet representation, IEEE Transactions on Pattern Analysis and Machine Intelligence 11 (1989) 764.
[46] D.L. Donoho, I.M. Johnstone, G. Kerkyacharian, D. Picard, Wavelet shrinkage: asymptotia?, Journal of the Royal Statistical Society, Series B 57 (301) (1995).
[47] G. Box, G.M. Jenkins, Time Series Analysis: Forecasting and Control, 1976.
[48] M. Bagshaw, R.A. Johnson, The effect of serial correlation on the performance of CUSUM tests II, Technometrics 17 (1) (1975) 73–80.
[49] R.A. Johnson, M. Bagshaw, The effect of serial correlation on the performance of CUSUM tests, Technometrics 16 (1) (1974) 103–112.
[50] A.V. Vasilopoulos, A. Stamboulis, Modification of control chart limits in the presence of data correlation, Journal of Quality Technology 10 (1) (1978) 20–30.
[51] S. Psarakis, G. Papaleonida, SPC procedures for monitoring autocorrelated processes, Quality Technology & Quantitative Management 4 (4) (2007) 501–540.
[52] S. Knoth, W. Schmid, Control charts for time series: a review, in: Frontiers in Statistical Quality Control, vol. 7, Springer, 2004, pp. 210–236.
[53] G.E. Box, D.R. Cox, An analysis of transformations, Journal of the Royal Statistical Society, Series B, Methodological 26 (2) (1964) 211–243.
[54] S. Vardeman, D.-o. Ray, Average run lengths for CUSUM schemes when observations are exponentially distributed, Technometrics 27 (2) (1985) 145–150.
[55] S. Chakraborti, P. Van der Laan, M. Van de Wiel, A class of distribution-free control charts, Journal of the Royal Statistical Society. Series C. Applied Statistics 53 (3) (2004) 443–462.
[56] G.J. Ross, N.M. Adams, Two nonparametric control charts for detecting arbitrary distribution changes, Journal of Quality Technology 44 (2) (2012) 102–116.
[57] A. Cohen, T. Tiplica, A. Kobi, OWave control chart for monitoring the process mean, Control Engineering Practice 54 (2016) 223–230.
[58] R. Ganesan, T.K. Das, V. Venkataraman, Wavelet-based multiscale statistical process monitoring: a literature review, AIIE Transactions 36 (9) (2004) 787–806.
[59] X. Chang, M.L. Stein, Decorrelation property of discrete wavelet transform under fixed-domain asymptotics, IEEE Transactions on Information Theory 59 (12) (2013) 8001–8013.
[60] D.L. Donoho, J.M. Johnstone, Ideal spatial adaptation by wavelet shrinkage, Biometrika 81 (3) (1994) 425–455.
[61] A. Cohen, T. Tiplica, A. Kobi, Statistical process control for AR(1) or non-Gaussian processes using wavelets coefficients, Journal of Physics. Conference Series 659 (1) (2015) 012043.
[62] R.D. Strum, D.E. Kirk, First Principles of Discrete Systems and Digital Signal Processing, Addison-Wesley, Reading, MA, 1989.
[63] D. Donoho, I. Daubechies, V. Pierre, Wavelets on the interval and fast wavelet transforms, Applied and Computational Harmonic Analysis 1 (7) (1993) 64–81.
[64] D. Donoho, I. Johnstone, Ideal de-noising in an orthonormal basis chosen from a library of bases, Technical Report, Department of Statistics, Stanford University, 1994.
[65] D. Donoho, I. Johnstone, Ideal spatial adaptation via wavelet shrinkage, Biometrika 81 (1994) 425–455.

https://t.me/learningnets

[66] A.G. Bruce, H.-Y. Gao, Understanding WaveShrink: variance and bias estimation, Biometrika 83 (4) (1996) 727–745.
[67] F. Abramovich, Y. Benjamini, Thresholding of wavelet coefficients as multiple hypotheses testing procedure, in: Wavelets and Statistics, Springer, 1995, pp. 5–14.
[68] S. Efromovich, Quasi-linear wavelet estimation, Journal of the American Statistical Association 94 (445) (1999) 189–204.
[69] J.S. Marron, S. Adak, I. Johnstone, M. Neumann, P. Patil, Exact risk analysis of wavelet regression, Journal of Computational and Graphical Statistics 7 (3) (1998) 278–309.
[70] G. Nason, Wavelet shrinkage using cross validation, Journal of the Royal Statistical Society, Series B 58 (1996) 463–479.
[71] H.B. Aradhye, B.R. Bakshi, R.A. Strauss, J.F. Davis, Multiscale SPC using wavelets: theoretical analysis and properties, AIChE Journal 49 (4) (2003) 939–958.
[72] O. Renaud, J.-L. Starck, F. Murtagh, Wavelet-based combined signal filtering and prediction, IEEE Transactions on Systems, Man and Cybernetics. Part B. Cybernetics 35 (6) (2005) 1241–1251.
[73] M.Z. Sheriff, F. Harrou, M. Nounou, Univariate process monitoring using multiscale Shewhart charts, in: 2014 International Conference on Control, Decision and Information Technologies (CoDIT), IEEE, 2014, pp. 435–440.
[74] E.K. Lada, J.-C. Lu, J.R. Wilson, A wavelet-based procedure for process fault detection, IEEE Transactions on Semiconductor Manufacturing 15 (1) (2002) 79–90.
[75] G. Shmueli, Wavelet-based monitoring for biosurveillance, Axioms 2 (3) (2013) 345–370.
[76] M. Madakayru, M. Nounou, H. Nounou, Integrated multiscale latent variable regression and application to distillation columns, Modelling and Simulation in Engineering (2013) 730456.
[77] S. Yoon, J.F. MacGregor, Principal component analysis of multiscale data for process monitoring and fault diagnosis, AIChE Journal 50 (11) (2004) 2891–2903.
[78] X. Li, X. Yao, Multi-scale statistical process monitoring in machining, IEEE Transactions on Industrial Electronics 52 (3) (2005) 924–927.
[79] J. Kalifa, S. Mallat, Minimax restoration and deconvolution, in: Bayesian Inference in Wavelet Based Methods, Springer, New York, NY, 1999.
[80] Z.G. Stoumbos, M.R. Reynolds Jr, T.P. Ryan, W.H. Woodall, The state of statistical process control as we proceed into the 21st century, Journal of the American Statistical Association 95 (451) (2000) 992–998.
[81] S. Pittner, S.V. Kamarthi, Feature extraction from wavelet coefficients for pattern recognition tasks, IEEE Transactions on Pattern Analysis and Machine Intelligence 21 (1) (1999) 83–88.
[82] X. Lou, K.A. Loparo, Bearing fault diagnosis based on wavelet transform and fuzzy inference, Mechanical Systems and Signal Processing 18 (5) (2004) 1077–1095.
[83] N. Saravanan, K. Ramachandran, Incipient gear box fault diagnosis using discrete wavelet transform (DWT) for feature extraction and classification using artificial neural network (ANN), Expert Systems with Applications 37 (6) (2010) 4168–4181.
[84] F. Harrou, M.N. Nounou, H.N. Nounou, Enhanced monitoring using PCA-based GLR fault detection and multiscale filtering, in: 2013 IEEE Symposium on Computational Intelligence in Control and Automation (CICA), IEEE, 2013, pp. 1–8.
[85] S. Mallat, A Wavelet Tour of Signal Processing, Elsevier, 1999.
[86] B.K. Alsberg, A.M. Woodward, D.B. Kell, An introduction to wavelet transforms for chemometricians: a time-frequency approach, Chemometrics and Intelligent Laboratory Systems 37 (2) (1997) 215–239.
[87] P. Teppola, P. Minkkinen, Wavelet–PLS regression models for both exploratory data analysis and process monitoring, Journal of Chemometrics 14 (5–6) (2000) 383–399.
[88] M. Madakyaru, F. Harrou, Y. Sun, Improved data-based fault detection strategy and application to distillation columns, Process Safety and Environmental Protection 107 (2017) 22–34.
[89] Y. Zhang, C. Ma, Fault diagnosis of nonlinear processes using multiscale KPCA and multiscale KPLS, Chemical Engineering Science 66 (1) (2011) 64–72.
[90] B.V. Kini, C.C. Sekhar, Multi-scale kernel latent variable models for nonlinear time series pattern matching, in: International Conference on Neural Information Processing, Springer, 2007, pp. 11–20.

https://t.me/learningnets

Chapter 6

Unsupervised deep learning-based process monitoring methods

6.1 Introduction

Effective detection of anomalies in highly complex processes is indispensable for managing them efficiently, safely, and in a productive fashion. Essentially, early detection of anomalies is required to keep the desired performance of the monitored processes and avoid the progression of anomaly impacts that may cause a significant loss in process profitability. As major advancements are being made in areas, such as data acquisition, sensor technologies and the Internet-of-Things, high-dimensional data from monitored processes are generated. These available data should be exploited to timely detect atypical behaviors that can be caused by either faulty sensors or process faults. Recently, machine learning and deep learning models have become a hotspot of recent research to better represent the data distribution features. Here, we review some of the unsupervised anomaly-detection methods based on binary clustering algorithms and unsupervised deep-learning models.

The main challenge for unsupervised anomaly detection methods is that the anomaly detection methods should be constructed using unlabeled training data. In supervised learning, models are trained with labeled data, and when a new observation is present, the trained model assigns it to one of the preestablished classes. All over the years, numerous classification methods have been developed for fault classification, including support vector machine (SVM), neural networks (NN), and k-nearest neighbor (kNN). Nevertheless, if unknown faults had occurred and been incorporated in the data, the majority of the supervised classification models can be unsuccessful in detecting them. In other words, these methods require periodic training to preserve high detection performance. However, getting labeled data is extremely challenging and time-consuming, in particular for high-dimensional datasets; moreover, this cannot be performed in real-time. In other words, getting a large labeled dataset is generally a lot of work, and is very time-intensive. Furthermore, even with labeled data, we cannot guarantee that they contain all possible anomalies. All these challenges suggest that a reasonable solution is to use unsupervised monitoring techniques.

https://doi.org/10.1016/B978-0-12-819203-5.00012-7
Copyright © 2021 Elsevier Inc. All rights reserved.
https://t.me/learningnets

To alleviate this difficulty, unsupervised anomaly detection techniques based on binary clustering methods play a significant role in anomaly detection without considering prior data labeling. The essence of clustering schemes for anomaly detection is to determine clusters characterizing normal behavior based on the anomaly-free data and then verify whether new unlabeled observations are within identified clusters or not. Observations that are not attributed to previously defined clusters are flagged out as anomalies. Essentially, the aim is to separate the given data into clusters that have high internal similarly and external dissimilarity, in the absence of prior knowledge. Several clustering techniques have been developed in the literature including algorithms based on density (e.g., mean-shift and DBSCAN), algorithms based on partitions (e.g., k-means and k-medoids) and algorithms based on hierarchy (e.g., BIRCH and agglomerative). Alternatively, there are two commonly used unsupervised machine learning algorithms for anomaly detection, namely one-class SVM and support vector data description (SVDD) [1–3]. These unsupervised methods use kernel functions for implicitly mapping the input data to higher-dimensional feature space to clearly separate normal from abnormal data. Of course, the main appeal of unsupervised anomaly detection methods has been their capability to detect unknown anomalies without any prior knowledge or data labeling.

Practically, detecting anomalies in high-dimensional data using the above-mentioned methods is very challenging and time-intensive. In fact, high process variables lead in the curse of dimensionality phenomenon that conducts to the generalization error of shallow methods, which grows with the number of irrelevant and redundant input variables [4–6]. For instance, SVMs are nonparametric algorithms, whose complexity increases quadratically with the number of observations [7]. In addition, shallow methods are limited to appropriately represent some types of function families [8].

An alternative solution to avoid the limitations of the above-mentioned shallow methods to detect anomaly in high-dimensional data is to use a model that can describe the variability in the underlying data. Accordingly, a compacted representation of data can be used to bypass the problem of high-dimensionality and decrease the high-computationally cost and complexity in the implementation [8,9]. In other words, feature extraction is an essential step in unsupervised anomaly detection, in which data are summarized by a set of characteristics (called features) that are the most informative and less redundant. Various data representation methods have been developed to reduce data dimensionality and extract relevant features including principal component analysis (PCA) as an unsupervised approach, and partial least squares as a supervised approach. On the other hand, other methods use nonlinear dimensionality reduction to uncover the relevant information hidden in the high-dimensional data, such as kernel PCA [10] and locally linear embedding [11]. However, these learning methods, either linear or nonlinear, fall within a shallow learning framework that mainly depends on the features employed for building the prediction model.

https://t.me/learningnets

Recently, much interest has been spurred on deep learning methods due to their great learning capability, in particular for large datasets. An additional appeal of deep learning methods is their strong flexibility and assumption-free nature. Unlike the shallow methods, deep learning-based methods can usually achieve improved abstractions of the input data [12]. In other words, deep learners possess great potential for extracting relevant features from the raw data to design improved models. Unsupervised deep-learning models, such as deep belief networks (DBNs), deep Boltzmann machine (DBM), and deep stacked autoencoder (DSA), discover features from one layer at a given time based on unlabeled data. The learned features are used to train the next layer, and so on. In [13], a hybrid stereovision approach combining deep Boltzmann machines (DBM) and the ability of autoencoders (AE) to reduce the dimensionality of data is designed to uncover obstacles in a road environment. The aim of this hybrid model is to extract relevant features with less redundancy that will be used as input by OCSVM for obstacle detection.

As previously mentioned, there is a very challenging problem when using the shallow unsupervised methods to detect anomalies in complex and high-dimensional datasets. To alleviate this problem, deep learners can be used as feature extractors; unsupervised shallow algorithms, such as kNN and one-class SVMs, are then applied to these reduced features for anomaly detection purposes. A variety of hybrid methods have been developed [13] by merging the benefits of deep-feature extractors and clustering and unsupervised machine learning algorithms for improving the performance of anomaly detection in high-dimensional datasets. For instance, the method in [14] uses a deep-stacked autoencoder as a feature extraction phase for the k-nearest neighbor algorithm to give a hybrid anomaly detection. This stereovision-based method has been applied to detect obstacles in the road environment and showed better performance in comparison to standalone clustering methods. In [13], a hybrid unsupervised method is used merging the benefits of the DBM with the dimensionality reduction capability of the AE for stereovision-based obstacle detection in an urban environment. The aim of this hybrid model is to extract relevant features with less redundancy that are used by OCSVM for obstacle detection.

First, we present a few binary clustering algorithms, and then briefly describe two common unsupervised anomaly detection schemes, namely one-class SVM and support vector data description (SVDD). Then, we present several deep-learning models with a focus on models based on autoencoders, and probabilistic models and introduce a hybrid framework merging the deep learning models and clustering algorithms for improving anomaly detection in high-dimensional complex processes. Lastly, we conclude with remarks and a perspective on future research.

6.2 Clustering

The essence of clustering is to find groups of data such that data points from the same group (cluster) are relatively more similar to each other than those in

https://t.me/learningnets

other clusters. Clustering algorithms have been largely exploited in data analysis. Conventional clustering techniques can be classified into different categories of clustering algorithms including partition-based, density-based and hierarchy-based clustering. In this section, we briefly present the basic idea of common clustering.

6.2.1 Partition-based clustering techniques

The essence of the clustering techniques based on the partition concept consists of considering the center of data as the center of the correspondent cluster. The two commonly used partition-based clustering techniques are k-means [15] and k-medoids [16]. Generally speaking, in the k-means algorithm, the center of the cluster is updated iteratively in such that a selected criterion for convergence is met. To handle discrete data, k-medoids has been developed as an enhancement of k-means. Various partition-based clustering algorithms have been developed, including partitioning around medoids (PAM) [17], clustering for large applications (CLARA) [17], and clustering large applications based on randomized search (CLARANS) [18]. More details about partition-based clustering can be found in [19–21].

6.2.1.1 k-Means clustering

The k-means procedure is an iterative, data-partitioning algorithm that aims to attribute n observations to one cluster of the k clusters characterized by centroids [22]. The number of clusters k is a priori selected in such a way that every cluster has its own centroid. Each observation is attributed to the cluster with the closest centroid, and each cluster updates its own centroid according to new included observations. The assignments and updates are repeated until stabilization of structure where no centroid updates are possible (Fig. 6.1). The principal steps of the algorithm are as follows:

1. Take k observations randomly as centroids (cluster centers)
2. Attribute each observation to the nearest cluster by computing its distance to each centroid
3. Find a new cluster center by computing the average of the observations including the assigned points
4. Iterate steps 2 and 3 until the assignment of clusters is not changing any more

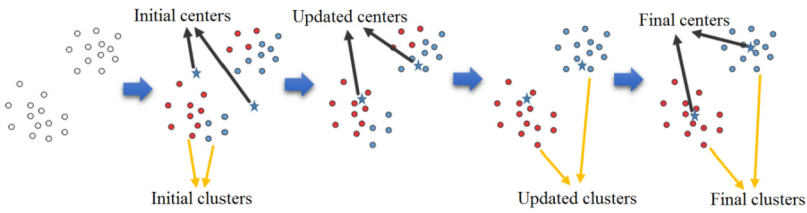


FIGURE 6.1 Conceptual illustration of the k-means clustering.

https://t.me/learningnets

In summary, the main appeal of the partition-based clustering methods is their high computational efficacy with moderate time-complexity. However, they are not suited for nonconvex data, sensitive to outliers, and easily fall into local optimum. In addition, in such methods, the number of clusters should be specified a priori defined, and the obtained results are highly sensitive to the number of clusters.

6.2.2 Hierarchy-based clustering techniques

As discussed above, the aim of clustering techniques as unsupervised learning techniques is to group a set of observations into distinguished subsets. Hierarchy-based clustering is one of the frequently used clustering techniques. The key concept of this type of clustering is based on the construction of the hierarchical relationships among data to cluster it [23]. At first, we assume that each of the observed data points represents a single cluster, we then merge the closest clusters in a new cluster until only one cluster is left. There are several existing hierarchy-based clustering procedures in the literature, such as balanced iterative reducing and clustering using hierarchies (BIRCH) [24,25], CURE [26], ROCK [27], and Chameleon [28]. Essentially, hierarchy-based clustering methods have the advantage of handling datasets having arbitrary shape and attributes of arbitrary type. In addition, the hierarchical relationship between clusters can easily be identified. However, the disadvantage of using hierarchy-based clustering is its high cost and the fact that the number of clusters should be a priori defined. For more details about hierarchy-based clustering methods, refer to [29–31].

6.2.2.1 BIRCH (hierarchical)

BIRCH is a hierarchy-based clustering procedure introduced to deal with streaming data or large datasets [25]. Basically, BIRCH clusters the data by building the clustering feature tree (CF tree), in which a subcluster is represented by one node. Specifically, the CF is a triple that summarizes the maintained information on a cluster. In fact, the CF tree will dynamically rise when a new observation arrives, and this incrementally improves the quality of subclusters. However, to obtain a good clustering performance, BIRCH needs the cluster count as input. More details about BIRCH algorithm can be found in [24,25].

The following describes the main steps in implementing the BIRCH clustering approach:

1. Construct a CF tree structure after a scan of the whole training data
2. Optimize the initial CF tree and create a compressed version of it
3. Perform global clustering by applying an existing clustering procedure on the leaves of the CF tree
4. Refine and improve the clustering quality with an additional full scan

https://t.me/learningnets

6.2.2.2 *Agglomerative clustering*

Hierarchical clustering procedures generate a nested sequence of clusters that can be visualized as a hierarchical tree [32]. The agglomerative hierarchical scheme is a nonparametric clustering that does not require the number of the clusters as input. The essence of this scheme is to merge clusters that are similar. The procedure is repeated until the intended number of clusters is achieved or the distance between the two closest clusters exceeds a certain threshold distance (see Fig. 6.2). The basic procedure of agglomerative clustering is outlined as follows:

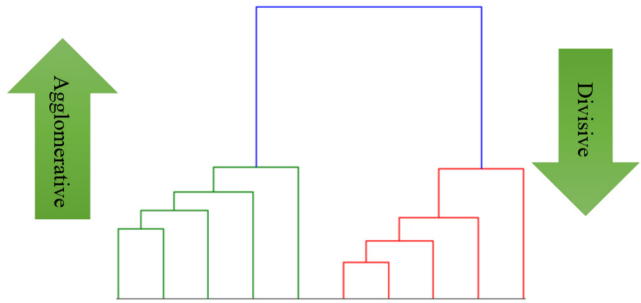


FIGURE 6.2 Illustrative example of agglomerative clustering (dendrogram).

- Consider each data point as a cluster and compute the proximity matrix
- Compute similarity/dissimilarity between each pair of data points
- Based on linkage function, group data points into a hierarchical cluster tree
- Combine closer clusters and update the proximity matrix

6.2.3 Density-based approach

The essence of the density-based clustering is to consider the data in a zone with a high density of the data points to be in the same cluster [33]. Thus, clustering procedures based on density possess the ability to uncover clusters with arbitrary shape and have the advantage of making less assumptions about the data without needing the number of clusters to be provided a priori. There are several clustering procedures in the literature designed using the density-based concept, such as DBSCAN [34], OPTICS [35], and mean-shift [36,37]. For further details on density-based clustering, we refer to [33,37,38]. The key benefit of this type of clustering is the greater capability to cluster data points with arbitrary shape. But, this approach achieves low clustering efficiency in the case when the density of data space isn't uniform. Its additional disadvantage is that it requires big memory when the volume of the data is large, and the clustering outcome is largely sensitive to the parameters.

Mean shift clustering is one of the commonly used density-based clustering procedures that will be briefly introduced next.

https://t.me/learningnets

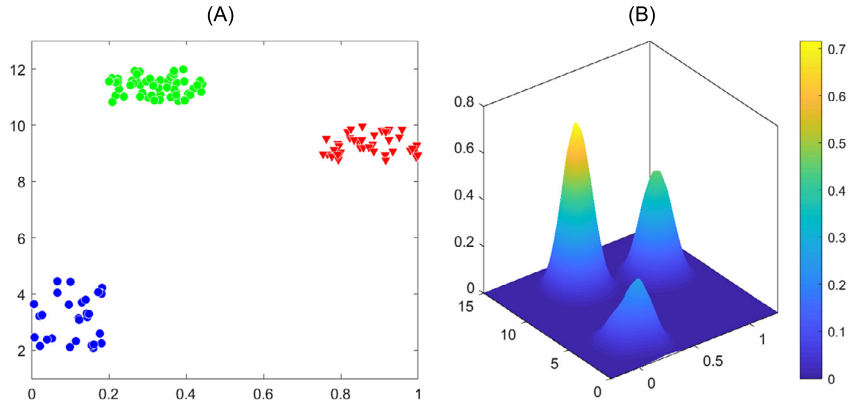


FIGURE 6.3 Mean shift clustering.

6.2.3.1 Mean shift clustering

Mean shift clustering is a nonparametric recursive hierarchy-based method. It is designed based on the concept of kernel density estimation (KDE). This scheme does not need a prefixed number of clusters. Essentially, the modes of a density estimate are used as indicators for the densest regions of the data, and these modes could be suitable cluster center estimates. The main idea of mean shift clustering is to shift all points belonging to the same region to the mean or the center of the area, and to repeat this task till convergence (Fig. 6.3). In KDE, the mean shift clustering is performed via a simple gradient procedure to estimate the modes of the KDE. There are different kernels to use in the mean shift clustering procedure, including Gaussian, exponential, Tophat, and Epanechnikov. Essentially, in the mean shift clustering, the kernel bandwidth plays a significant role in the clustering result. Indeed, if a large bandwidth is selected, then fewer clusters will be formed in the mean shift clustering, and vice versa. Accordingly, the bandwidth selection is a crucial step in the mean shift clustering procedure because it has a direct impact on the quality of the density estimation, and thus heavily impacts the mean shift clustering. Specifically, selecting a poor bandwidth estimate leads to modes that do not appropriately represent the dense regions of the data. Various schemes have been designed to address the problem of bandwidth selection [36,39,40]. For more information about shift mean procedure, we refer to [37,38].

The main steps needed for implementing mean shift clustering are the following:

1. Define a window (i.e., bandwidth of the kernel) and place it on a data points
2. Calculate the mean of all observations within the window
3. Move the center of the window to the position of the mean
4. Iterate steps 2 and 3 until convergence

https://t.me/learningnets

6.2.3.2 *k*-Nearest neighbor clustering

Here, we briefly present the k-nearest neighbor (kNN) method, which is a simple and effective nonparametric clustering procedure to cluster different features [41,42]. kNN is an assumption-free clustering procedure that does not consider a priori assumptions about the data structure [43]. This makes it very appealing, in particular to handle non-Gaussian or nonlinear features separation. Basically, kNN as a distance-based procedure rests on the following intuitive concept: for a new unlabeled observation, x, the kNN discovers the nearest observation in the training data and attributes x to the closest class and most frequently within the k-nearest neighbors.

When kNN procedure is used for anomaly detection, we have only normal operation data available as training data (i.e., only one class of data). In this case, kNN computes the distance between the newly arrived unlabeled data point, x, and the k nearest neighbors in the training data, and if the distance is relatively close to zero, then the measurement is anomaly-free data. Otherwise, it is considered as a potential anomaly. Essentially, large kNN distances are used as an anomaly indicator. Euclidean and Minkowski distances are frequently used to compute the closeness in kNN-based procedures.

Fig. 6.4 gives a simple example to illustrate the intuitive idea of the kNN procedure. Two different classes are present in the data represented respectively with blue circles and yellow squares (i.e., anomaly-free and anomalous); the green star is the observation to be classified using the kNN procedure. As an example, in the case when the number of the nearest neighbors is selected to be $k = 1$, then the green star is assigned to the yellow square class. Here we used Euclidean distance to identify nearest neighbors. On the other hand, if we fix $k = 5$, the green star will be attributed to the class with blue circles. This is because the number of blue circles is larger than the number of yellow squares in this case (within the dashed circles, $k = 5$). In fact, the kNN procedure does

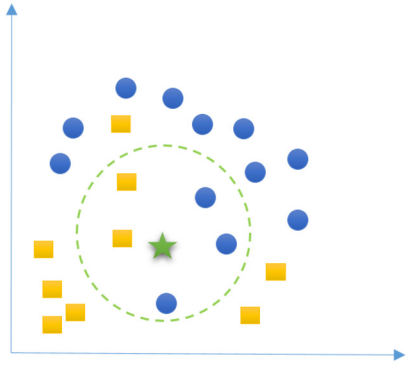


FIGURE 6.4 kNN clustering procedure.

https://t.me/learningnets

not include an explicit training step, the principal computation in such clustering is the kNN distance to a new data point.

For anomaly detection purposes, the kNN procedure assesses the closeness between the newly arrived points and the k nearest neighbors in the training (anomaly-free) data. No labeling is needed for the training stage. At first, for each data point, x_i, compute the kNN distance (e.g., Euclidean) to its nearest neighbor in the training set, D_i,

$$D_i = \sum_{j=1}^{k} d_{ij}, \tag{2}$$

where d_{ij} denotes the distance between x_i and its jth nearest neighbor. The detection threshold of kNN can be computed nonparametrically as the $(1-\alpha)$th quantile of the estimated distribution of kNN distances calculated using KDE. Then, for a new data point, the kNN distance is computed and compared to the detection threshold for anomaly detection. Other approaches, to setup the kNN detection threshold, apply monitoring charts, such as EWMA and Shewhart, to the kNN distances [44]. In particular, the detection performance can be improved when using kNN with EWMA because it considers information from past and actual data in the decision process.

6.2.4 Expectation maximization

Expectation maximization (EM) scheme tries to approximate the input data distributions for solving the problem of maximum likelihood estimation for data in which some variables cannot be explicitly observed, called latent variables that can be inferred from the values of the other observed variables [45]. The main objective of EM clustering is to approximate the observed data distributions based on mixtures of different distributions in different clusters (see Fig. 6.5). Suppose that we have a data matrix $\mathbf{X}$, the aim is to find the value of $\mathbf{\Phi}$ maximizing the log-likelihood, $\mathcal{L}(\Phi) = \log P(x|\Phi)$. Using latent variables $\mathbf{Z}$,

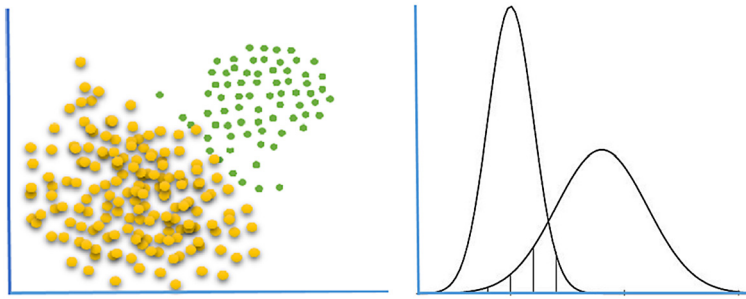


FIGURE 6.5 Expectation maximization.

https://t.me/learningnets

the log-likelihood, $\mathcal{L}$, can be expressed as $\mathcal{L}(\Phi|X, Z) = \log P(X|\Phi, Z)$. The elements of the observed data $\mathbf{X}$ are often assumed follow Gaussian distributions with the parameters $\Phi = \{\mu, \sigma\}$. Generally speaking, EM is an iterative procedure for obtaining solution to maximum likelihood estimation with latent variables. To this end, EM iteratively performs expectation (E) and maximization (M) steps. The role of the E step is to generate a function for the expectation of the log-likelihood that is assessed based on the actual estimate for the parameters. On the other hand, the M step consists of computing parameters that maximize the expected log-likelihood obtained in the previous E step. For each cluster, EM attempt to estimate both standard deviations and means of the observed data (distribution) in order to maximize the likelihood. Expectation maximization decides about memberships to a cluster through computing probabilities of *Gaussian mixture* distributions. Every observation can be considered as a member of each cluster with a certain probability but the probability maximizing the observed data likelihood is utilized to select the cluster.

6.3 One-class classification

Here, we briefly introduce the basic idea behind two commonly used one-class anomaly detection methods, SVM and SVDD.

6.3.1 One-class SVM

Supervised support vector machines (SVMs) that have been widely applied for data classification are not suited for anomaly detection because of their need for labeled data. To bypass this challenge, unsupervised methods, such as one-class SVMs (OCSVMs), are designed by building a model that describes only normal operating conditions and uses it to flag out data points that do not conform to the reference model [1,2,46]. Generally speaking, the OCSVM procedure uses a kernel function for projecting input data points to a higher-dimensional feature space, where the discrimination of normal from anomalous data becomes clearer and easier. Essentially, a kernel-based procedure has the capacity to model the process nonlinearity of normal behavior if appropriately used. OCSVM uncovers anomalies in the feature space based on the construction of a hyperplane that appropriately separates the data from the origin. In fact, the OCSVM procedure learns decision functions $\mathcal{D}(x)$ that return -1 or 1 to respectively show whether the data is an anomaly or normal. The detection function $\mathcal{D}(x)$ is given as

$$\mathcal{D}(x) = \begin{cases} +1, & \text{if } x \text{ belongs to the area including most of observations,} \\ -1, & \text{otherwise.} \end{cases} \tag{6.1}$$

Consider that $x_1, \ldots, x_j \in \mathcal{D}$ and $j \in [1, k]$ represents the training data. OCSVM projects the data points into the high-dimensional feature space $\mathcal{F}$

https://t.me/learningnets

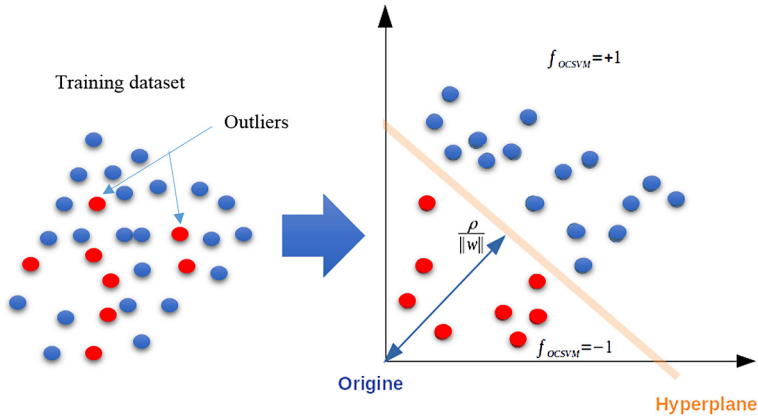


FIGURE 6.6 Schematic presentation of a one-class SVM (OCSVM).

based on kernel functions (Eq. (6.2)), like the radial basis function (RBF)

$$\mathcal{K}(x, y) = \langle \Psi(x) , \Psi(y) \rangle, \tag{6.2}$$

where x and y refer to the input vectors, Ψ denotes a feature map $\mathcal{X} \rightarrow \mathcal{F}$ and $\mathcal{X}$ denotes set of observed dataset x. As given in Fig. 6.6, the decision function $f(x)$ intends to maximize the Euclidean distance from the origin to the separating hyperplane $\mathcal{H}$. The decision hyperplane separates the training measurements in the features space $\mathcal{F}$. Thus, we get the objective function $f(x)$ given as

$$f(x) = \text{sign}(\langle w, \Psi(x) \rangle - \rho), \tag{6.3}$$

where w denotes a weight vector and ρ refers to an offset. The best hyperplane is related to the parameters w and ρ that are determined by solving the following optimization problem:

$$\begin{aligned} \min_{w \in \mathcal{F},\, \xi \in \mathbb{R}^l,\, \rho \in \mathbb{R}} & \frac{1}{2}\|w\|^2 + \frac{1}{\nu l}\sum_i^l \xi_i - \rho, \\ \text{s.t. } & \langle w, \Psi(x) \rangle \geq \rho - \xi_i, \quad \xi_i \geq 0, \end{aligned} \tag{6.4}$$

where $\nu \in [0, 1]$ refers to a parameter characterizing the solution.

6.3.2 Support vector data description (SVDD)

The SVDD is a one-class classification algorithm introduced by Tax and Duin [1] as a special case of the standard SVM. This unsupervised algorithm is beneficial for anomaly detection and has been applied to a wide range of applications, such as face recognition [47], pattern denoising [48], and anomaly

https://t.me/learningnets

detection [49,50]. The essence of the SVDD procedure is to find a spherically shaped border surrounding the majority of the training datasets while discarding some observations that are excluded as potential anomalies. This makes it clearly different from the OCSVM procedure in terms of the shape of the discriminatory boundaries, where OCSVM uses hyperplane while SVDD uses spheres (hypersphere) for separating normal from abnormal features (Fig. 6.7).

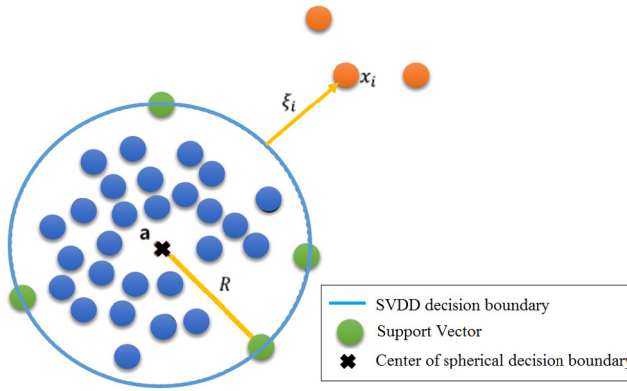


FIGURE 6.7 SVDD vs OCSVM decision boundaries.

Assume $x_i \in \mathbb{R}^n$, $i = 1, \ldots, l$ is a set of training data, the goal of the SVDD algorithm as a kernel-based method is to find the most tightly fitting hypersphere that includes most of the data points in the kernel mapping space (Fig. 6.7). At first, the input data is transformed into the high dimensional space (called feature space) via the nonlinear mapping function $\phi(\cdot)$. In this feature space, the SVDD scheme attempts to find a hypersphere with radius R and its center a to encompass the majority of the training data and exclude anomalous data points (outliers) as much as possible. Due to the possible presence of outliers in the data, in order to find the hypersphere using the SVDD scheme, we solve the following optimization problem with the help of slack variables ξ_i:

$$\begin{aligned} &\min_{R,a,\xi} R^2 + C\sum_{i=1}^{l}\xi_i \\ &\text{s.t. } \|\phi(x_i) - a\|^2 \le R^2 + \xi_i, \quad i = 1, \ldots, l, \\ &\quad \xi_i \ge 0, \quad i = 1, \ldots, l. \end{aligned} \tag{6.5}$$

The role of parameter C is to control the size of the hypersphere and the amount of the training data excluded from the hypersphere (i.e., outliers). Usually, it is expressed as $\frac{1}{\nu N}$, where ν denotes the upper limit to the permitted outliers and lower limit on the number of support vectors that establish the hypersphere frontier, and N denotes the size of data. As the training data may

https://t.me/learningnets

contain fewer outliers or anomalies, the slack variables $\xi_i > 0$ are used to permit excluding these observations as outliers from the hypersphere. Thus, the sphere is shrunk to encompass only normal data and getting better optimum for the criterion in Eq. (6.5).

Instead, the primary problem in (6.5) can be formulated as a dual problem by solving an optimization problem over $\{\alpha_i, i = 1, 2, \ldots, n\}$ $(0 \leq \alpha \leq C)$ as follows:

$$\begin{aligned} &\min(\alpha_i) = \sum_{i,j} \alpha_i \alpha_j \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle - \sum_i \alpha_i \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_i) \rangle \\ &\text{subject to } 0 \leq \alpha_i \leq C, \forall i = 1, 2, \ldots, n, \sum_i \alpha_i = 1. \end{aligned} \tag{6.6}$$

Using the kernel trick, the inner products in (6.6) can be replaced by the kernel function, $k(\mathbf{x}_i, \mathbf{x}_j) = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle$, and the optimization problem can be expressed as:

$$\begin{aligned} &\min\ L(\alpha_i) = \sum_{i,j} \alpha_i \alpha_j k(\mathbf{x}_i, \mathbf{x}_j) - \sum_i \alpha_i k(\mathbf{x}_i, \mathbf{x}_i) \\ &\text{subject to } 0 \leq \alpha_i \leq C, \forall i = 1, 2, \ldots, n, \sum_i \alpha_i = 1. \end{aligned} \tag{6.7}$$

The majority of α_i^* obtained by solving this optimization problem are null. The nonnull $0 < \alpha_i^* < C$ represent the support vectors that determine the boundary and size of the hypersphere. The center of the hypersphere, a, is computed based on all the support vectors as

$$\mathbf{a} = \sum_i \alpha_i^* \Phi(\mathbf{x}_i). \tag{6.8}$$

The radius of the hypersphere can be computed as

$$\begin{aligned} R^2 &= \frac{1}{N_b} \sum_{k=1}^{N_b} \{\|\Phi(\mathbf{x}_k) - \mathbf{a}\|^2\} \\ &= \frac{1}{N_b} \sum_{k=1}^{N_b} \left\{ k(\mathbf{x}_k, \mathbf{x}_k) - 2 \sum_i \alpha_i^* k(\mathbf{x}_k, \mathbf{x}_i) + \sum_{i,j} \alpha_i^* \alpha_j^* k(\mathbf{x}_i, \mathbf{x}_j) \right\}. \end{aligned} \tag{6.9}$$

Here $\Phi(\mathbf{x}_k)$, $k = 1, 2, \ldots, N_b$ denote the support vectors that are the boundary of the training data, and N_b denotes the total number of support vectors.

The SVDD scheme flags out an observation, x, as an anomaly if

$$\|\phi(x) - a\|^2 > R^2. \tag{6.10}$$

https://t.me/learningnets

6.4 Deep learning models

We present in this subsection the autoencoders that are among the most used algorithms to build up deep networks. We briefly present three commonly used variants of autoencoders, namely undercomplete variational autoencoders, denoising autoencoders, and contrastive autoencoders. For a more complete review, refer to [51].

6.4.1 Autoencoders

An autoencoder (AE) is basically a variant of a neural network that consists of three layers: input, hidden, and output layers that work in an unsupervised learning paradigm [9]. Essentially, an AE predicts the value of the output $\hat{\mathbf{x}}$ based on the input $\mathbf{x}$ through a hidden layer h (Fig. 6.8) [52–54]. They are frequently applied in dimensionality reduction and feature extraction. In fact, autoencoders consist of two components, an encoder and a decoder. The essence of the encoder function is to map the input data into the hidden layer, $h = f(\mathbf{x})$. An AE with nonlinear encoder functions provides more flexibility to extract more features, in comparison to principal component analysis [9]. On the other hand, the role of the decoder component is reconstructing the input data based on the hidden layer representation, $\widehat{\mathbf{x}} = g(h)$. Of course, in the encoding stage, the AE discovers a compacted representation (or latent variables) of the input data, while in the decoding stage, the AE tries to reconstruct the input based on encoded data.

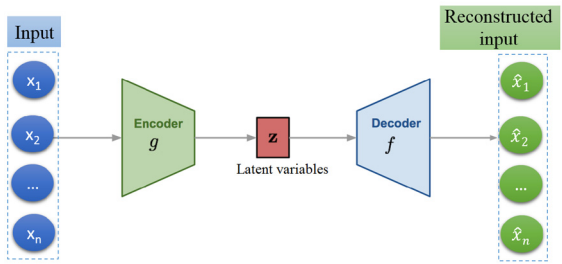


FIGURE 6.8 Schematic illustration of an autoencoder.

Assume that $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k)$ are unlabeled data points, during the encoding process, the N-dimension input data, $\mathbf{x}_i$, are mapped to M-dimension vectors $\mathbf{h}_i$ using the encoding function f. The hidden layer h is computed as

$$h(\mathbf{x}) = f(\mathbf{W}_1\mathbf{x} + \mathbf{b}_1). \tag{6.11}$$

The term $\mathbf{W}_1$ refers to the weight matrix of the encoder, and $\mathbf{b}_1$ denotes the bias vector. In the decoding phase, the decoding maps every M-dimensional vector back for reconstructing the input vector, $\mathbf{x}$, expressing the reconstructed

https://t.me/learningnets

vector, $\hat{\mathbf{x}}$, as

$$\hat{\mathbf{x}} = g(\mathbf{W}_2 h(\mathbf{x}) + \mathbf{b}_2), \tag{6.12}$$

where $\mathbf{W}_2$ is the weight matrix of the decoder, and $\mathbf{b}_2$ denotes the bias vector.

The reconstruction error, which is the difference between the original data, $\mathbf{x}$, and the reconstructed data, $\hat{\mathbf{x}}$, is usually in the form of a loss function in the training stage. Specifically, the training of an autoencoder is usually performed by minimizing the negative log-likelihood of the reconstruction, given the encoding $f(\mathbf{x})$ [9]:

$$\text{Reconstruction error} = -\log(P(\mathbf{x}|f(\mathbf{x})), \tag{6.13}$$

where P denotes the probability assigned to the input vector $\mathbf{x}$ by the model. The goal is that $f(\mathbf{x})$ captures the main factors of variation in the data.

In fact, the incorporation of latent variable models makes autoencoders behave like generative models. Stacked autoencoder models were largely employed in image denoising [55,56] and content-based image retrieval [57].

6.4.1.1 Variational autoencoder

A variational autoencoder (VAE) has a similar structure as that of a conventional AE in the way that it contains an encoder and a decoder network. The major shortcoming of the conventional AE is that the inputs are converted to discrete variables (not continuous) which makes the interpolation a challenging task. Specifically, as a generative model, the AE randomly takes samples from the latent space, and this is difficult if it is discontinuous or contains gaps. VAR is a variant of AE that is designed to estimate the distribution of the feature data [58–60]. Its training is regularized to avoid overfitting and guarantee that the latent space possesses suitable properties that enable the generative process. VAE possesses a continuous latent space which makes it a powerful generative model. In fact, differing from the conventional AE, a VAE possesses an additional layer in charge of sampling the latent vector $\mathbf{z}$, and its loss function contains an extra term that constrains the generation of a latent vector with a roughly predefined distribution, $p(\mathbf{z})$, frequently considered as a standard Gaussian. Essentially, in the training phase, by using the input data, the encoder generates vectors of means, μ, and standard deviation, σ of the latent vector distribution. Then the decoder reconstructs the input data by using the latent variable, which is sampled from the latent vector distribution (Fig. 6.9).

Generally speaking, VAEs are generative models that use latent space for mapping the input data to the feature spaces, making them able to generate new samples based on the learned model parameters [58–60]. In fact, the encoding procedure performed by the VAE is also known as approximate inference network, the $q_\theta(\mathbf{z}|\mathbf{x})$ is used on the training to get $\mathbf{z}$ (i.e., learn a probability distribution of the underlining input dataset), the next step is the decoder transformation performed using $p_\phi(\mathbf{x}|\mathbf{z})$ to reconstruct the input data.

https://t.me/learningnets

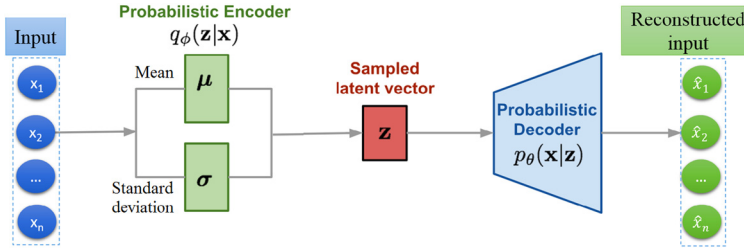


FIGURE 6.9 Schematic illustration of VAE.

The central idea behind the VAEs is that its training is performed by minimizing the loss function $\mathcal{L}(\mathbf{x}; \theta, \phi)$ associated with data point $\mathbf{x}$.

$$\mathcal{L}(\mathbf{x}; \theta, \phi) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log(p_\theta(\mathbf{x}|\mathbf{z}))] - D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z})). \tag{6.14}$$

The first term in this loss function, $\mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x, z)\right]$, refers to the reconstruction error. It is basically the expectation of the log-likelihood that the input can be generated using the sampled values of the inferred distribution. It pushes the decoder to learn the reconstruction of the input. The key idea behind the VAE is to train a parametric encoder (called an inference network) that produces the parameters of q. The second term, $D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}))$, denotes the Kullback–Leibler divergence between the distribution of the encoded latent vectors $q_\phi(\mathbf{z}|\mathbf{x})$ and the desired distribution $p_\theta(\mathbf{z})$ which is usually assumed as the standard Gaussian. This extra term can be viewed as an adjustment enforced in the feature space. It should be noted that to minimize the objective function and get $\mathbf{x}$ as close as possible to the input vector $\hat{\mathbf{x}}$, the term $D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}))$, enforces μ and σ towards the zero and the unity vectors, respectively.

6.4.1.2 *Denoising autoencoder*

In this section, we discuss denoising autoencoders (DAE), which are a stochastic version of conventional autoencoders [55,56,61]. In fact, in the case when the AE contains more nodes in the hidden layer than inputs, it can lead to learning identically the input and making the autoencoder useless. This is usually called learning of *identity function* that results in the output equaling the input. To overcome this difficulty, DAE introduces noise to the training data. Then, in DEA the raw input data are reconstructed from the noisy data after the encoding and decoding steps. Basically, the essence of the DAE is denoising the noisy input by reconstructing the original input in a clean version. Essentially, the DAE learns how to denoise an input by reconstructing a clean input $\hat{\mathbf{x}}$ from a corrupted version of the original input $\mathbf{x}$ (Fig. 6.10).

DAEs have proven to be effective in learning more robust features and able to reduce sensitivity to small stochastic disturbances. As unsupervised learning algorithms, they are widely applied in various applications [62,63].

https://t.me/learningnets

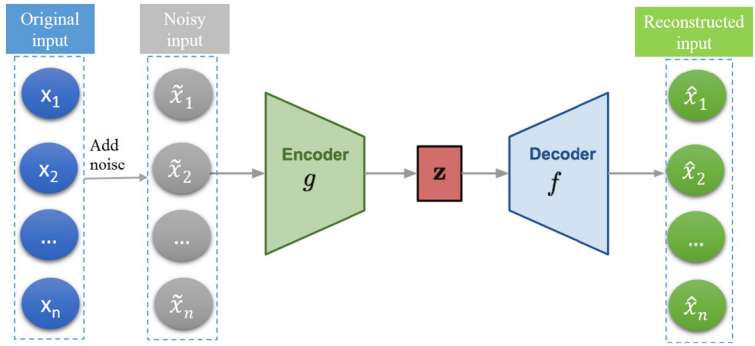


FIGURE 6.10 Schematic illustration of a denoising autoencoder.

DAE training is performed as follows. At first, the input $\mathbf{x}$ is corrupted with noise by adding some random, $\hat{\mathbf{x}} \sim q_D(\hat{\mathbf{x}}|\mathbf{x})$. Then, the corrupted input $\hat{\mathbf{x}}$ is encoded as in the conventional AE, the coded features of the hidden layer are given as $\mathbf{y} = f_\theta(\hat{\mathbf{x}})$. Lastly, the obtained features in the hidden layer are decoded and the reconstruction of the input is expressed as $\mathbf{z} = g_{\theta'}(h) = g_{\theta'}(f(\hat{\mathbf{x}}))$.

The parameters of DAE are obtained in the training phase by minimizing the following loss function:

$$\arg\min_{\theta,\theta'} L(\mathbf{x}, g(f(\hat{\mathbf{x}}))) \tag{6.15}$$

where L refers to a loss function such as the Euclidean norm, and θ and θ' are the DAE parameters.

In summary, by introducing a penalty term to the cost function, the DAE learns robust features by changing the reconstruction error term of the loss function. The model parameters are optimized using the same procedure used in the autoencoder called stochastic gradient descent.

6.4.1.3 Contrastive autoencoder

The aim of the contractive autoencoder (CAE) is to achieve a robust learned representation that is less sensitive to small variations in the data by introducing a penalty term to the loss function [64,65]. To this end, an explicit regularizer is added on the code $h = f(\mathbf{x})$ for encouraging the derivatives of f to be as small as possible:

$$\|J_f(\mathbf{x})\|_F^2 = \sum_{ij}\left(\frac{\partial h_j(\mathbf{x})}{\partial h_i}\right)^2. \tag{6.16}$$

In other words, the loss function of the CAE is based on the penalty term, Frobenius norm of the Jacobian matrix, $\|J_f(\mathbf{x})\|_F^2$, to make the encoding less sensitive to small variations in the training dataset. Indeed, Frobenius norm reduces the

https://t.me/learningnets

sensitivity of representation learned towards the training input, where penalizing promotes the feature space mapping to be contractive in the neighborhood of the training data.

The aim of the training phase of the CAE is to determine parameters $\theta = \{W, b_h, b_y\}$ for which $\mathbf{y}$ is the reconstructed version of $\mathbf{x}$ that minimizes the reconstruction error on a training dataset D, which corresponds to minimizing the objective function:

$$\mathcal{J}_{CAE}(\theta) = \sum_{\mathbf{x} \in D} \left(\mathcal{L}(\mathbf{x}, g(f(\mathbf{x}))) + \lambda \| J_f(\mathbf{x}) \|_F^2 \right) \tag{6.17}$$

where $\mathcal{L}$ refers to the reconstruction error, the cross-entropy loss when the activation function s_g used is the sigmoid and with inputs $\mathcal{X} \in [0, 1]$. The term $\mathcal{L}$ is expressed as follows: $\mathcal{L}(\mathbf{x}, \mathbf{y}) = -\sum_i \log(y_i) + (1 - x_i) \log(1 - y_i)$. To enforce robustness of the representation $f(\mathbf{x})$ obtained for a training input $\mathbf{x}$, the penalization is measured as the Frobenius norm of the Jacobian $J_f(\mathbf{x})$ of the nonlinear mapping applied to the input. The mapping is contracting the data efficiently and hence the name contractive autoencoder. The central role of the Jacobian term is a mapping filter by ignoring low variations and focusing more on significant representations to allow reconstruction of the training with small errors.

6.4.2 Probabilistic models

In this subsection, we present two commonly used energy-based models Boltzmann machine (BM) and restricted Boltzmann machine (RBM) that can be used to build up deep learning models.

6.4.2.1 Boltzmann machine

A Boltzmann machine (BM) is a stochastic neural network that comprises visible and hidden units, where the visible units are the input data (Fig. 6.11) [66]. In a BM, every unit is connected to all other units. The hidden units act as a set of latent variables (features) allowing BM to model distributions over visible state vectors (Fig. 6.11). The BM model aims to understand the distribution underlying the input data and regenerate the data using that distribution. It is represented by an undirected graph, part of Markov random field, which forms a network of symmetrically connected units (neurons), capable to make stochastic decisions or binary decision (0, 1).

The joint probability distribution of BM, which is an energy-based model, is computed based on an energy function as a Boltzmann distribution [66]:

$$P(\mathbf{v}) = \frac{1}{Z(\theta)} \exp(-E(\mathbf{v})), \tag{6.18}$$

https://t.me/learningnets

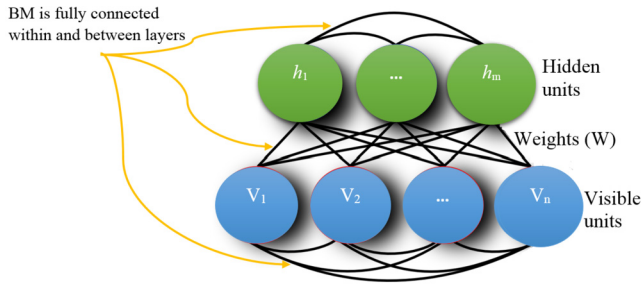


FIGURE 6.11 Schematic presentation of a Boltzmann machine.

where $P(\mathbf{v})$ refers to the energy function and Z is the partition function that guarantees that $\sum_{\mathbf{v}} P(\mathbf{v}) = 1$. The energy function of the BM is expressed as

$$E(\mathbf{v}) = -\mathbf{v}^T \mathbf{W}\mathbf{v} - \mathbf{b}^T \mathbf{v}, \tag{6.19}$$

where $\mathbf{W}$ denotes the weight matrix of model parameters and $\mathbf{b}$ refers to the vector of bias parameters.

In the training stage, the parameters of the BM model (i.e., weights and biases) are determined so that the likelihood of the observed data is maximized. Specifically, the gradient descent on the log of the likelihood function is performed to find the BM parameters. In other words, the goal is to retrieve weights and biases that define a Boltzmann distribution in which the training vectors have high probability.

BMs are unsupervised models, which involve learning a probability distribution from the training dataset. They are also known as generative models due to their capability to generate new samples from the learned data distribution. However, although with fewer nodes, too many connections are needed in a BM model to make effective computations. To alleviate this problem, numerous models have been introduced including conditional Boltzmann machines [67] and Restricted Boltzmann Machine (RBM) [68]. Next, we briefly describe the RBM model, in which they are no connections between units in the same layer.

6.4.2.2 *Restricted Boltzmann machine*

RBMs are a special form of Boltzmann machine that have no intralayer connections, that is, connections visible-to-visible and hidden-to-hidden are not available [69–71] (Fig. 6.12). RBMs are undirected probabilistic graphical models composed of two layers, namely visible and hidden layers. In RBMs, there are connections between each visible unit and every hidden unit. RBM is usually employed as a layerwise training model in the design of deeper models, such as deep belief networks (DBN) and the hierarchical probabilistic model deep Boltzmann machine (DBM), by stacking several RBMs [72]. Essentially, RBMs are stochastic neural networks that possess m visible units, $\mathbf{v} \in \{0, 1\}^m$

https://t.me/learningnets

and n hidden units, $\mathbf{h} \in \{0, 1\}^n$. Numerous learning procedures have been designed to train RBMs models based on Markov chain Monte Carlo (MCMC) and Gibbs sampling to get an estimator of the log-likelihood gradient [9].

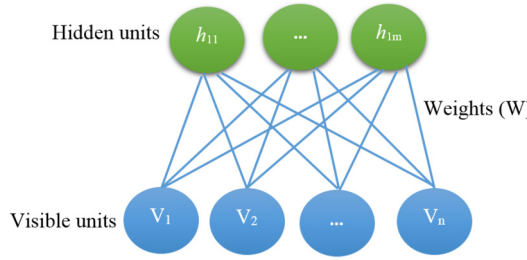


FIGURE 6.12 Diagrammatic illustration of RBM.

RBMs are a particular kind of energy-based models. The energy function of the RBM configuration is given as [73]:

$$\begin{aligned} E(\mathbf{v}, \mathbf{h}; \theta) &= -\sum_{i=1}^{m} b_i v_i - \sum_{j=1}^{n} a_j h_j - \sum_{i=1}^{m}\sum_{j=1}^{n} w_i v_i h_j \\ &= -\mathbf{b^T v} - \mathbf{a^T h} - \mathbf{v^T W h}, \end{aligned} \tag{6.20}$$

where W_{ij} denotes the weight matrix between visible variable v_i and hidden variable h_j, and b_i and a_i denote respectively bias terms of visible and hidden units. The aim of the training is to determine the suitable values of these parameters based on some datasets.

As an energy-based model, the joint probability distribution for the RBM model is defined based on energy function as follows:

$$P(\mathbf{v}, \mathbf{h}; \theta) = \frac{1}{Z(\theta)} \exp(-E(\mathbf{v}, \mathbf{h}; \theta)) \tag{6.21}$$

$$= \frac{1}{Z} \prod_{ij} e^{W_{ij} v_i h_j} \prod_i e^{b_i v_i} \prod_j e^{a_j h_j}, \tag{6.22}$$

where Z refers to a partition function defined as

$$Z(\theta) = \sum_{\mathbf{v}} \sum_{\mathbf{h}} E(\mathbf{v}, \mathbf{h}; \theta). \tag{6.23}$$

As RBM is restricted to have no connections between units in the same layer, the visible units are conditionally independent given the state of the hidden units, and vice versa,

$$p(\boldsymbol{h}|\boldsymbol{v}) = \prod_{i=1}^{n} p(h_i|v), \tag{6.24}$$

https://t.me/learningnets

$$p(\boldsymbol{v}|\boldsymbol{h}) = \prod_{i=1}^{m} p(v_i|h). \tag{6.25}$$

The marginal distribution of the visible variables can be easily computed because hidden units are not connected to visible units:

$$p(\boldsymbol{v}) = \frac{1}{Z} \sum_{h} p(\boldsymbol{v}, \boldsymbol{h}). \tag{6.26}$$

In the case of binary visible unit $\mathbf{v} \in \{0, 1\}^m$ and hidden units $\mathbf{h} \in \{0, 1\}^n$, the marginal probability distributions of the RBM can be compute as

$$p(h_j = 1|\mathbf{v}) = \sigma\left(\sum_{j=1}^{m} w_{ij} v_j + a_i\right), \tag{6.27}$$

$$p(v_i = 1|\boldsymbol{h}) = \sigma\left(\sum_{j=1}^{n} w_{ij} h_i + b_j\right), \tag{6.28}$$

$$\sigma(x) = \frac{1}{1 + \exp(-x)}, \tag{6.29}$$

where $\sigma(\cdot)$ refers to the logistic function. In [71], an improved version of RBM, called Gaussian–Bernoulli RBM, has been introduced to handle various data types, such as real data, rather than binary inputs, where $\mathbf{v} \in \mathbb{R}^m$ and hidden units $\mathbf{h} \in \{0, 1\}^n$.

Generally speaking, the core purpose of model training is to tune the model parameters (weights matrix $\mathbf{W}$) to maximize the probability attributed to the training data under the model. Formally, proper parameters are obtained by maximizing the log-likelihood function, where the derivative of the log-likelihood is computed with respect to $\mathbf{W}$ takes the following form [74–76]:

$$\Delta w_{ij} = \alpha(E(v_i, h_j) - \hat{E}(v_i, h_j)). \tag{6.30}$$

The term α denotes the learning rate and $\hat{E}(v_i, h_j)$ refers to the energy expected from the distribution learned by the model, which is intractable [71], Gibbs sampling is used to overcome this problem. In [77], a learning procedure called contrastive divergence (CD) that can avoid the expensive computation has been introduced by Hinton to train RBM. This CD procedure is becoming a standard way for training RBM and its extensions.

6.4.3 Deep neural networks

In this section, three commonly used deep learning models, namely deep belief network (DBN), deep Boltzmann machine (DBM), and stacked autoencoder (SAE), are briefly described.

https://t.me/learningnets

6.4.3.1 Deep belief networks

DBNs are among the first introduced nonconvolutional models addressing the training of deep structures [78,79]. DBNs are efficient learning models that combine the advantages of dimensionality reduction and data representation, without imposing assumptions on the underlying data structures. Basically, DBNs are multilayer probabilistic generative models that are designed by the stacking of several RBMs (Fig. 6.13), in which each hidden layer (RBM) is a Markov random field. From Fig. 6.13, the first RBM (RBM1) is built up from the input layer and the first hidden layer, and the second RBM (RBM2) consists of the first and the second hidden layers, and the third RBM (RBM3) is formed based on the second and the third hidden layers. That is, a DBN model possesses multiple hidden layers, $\mathbf{h}_1, \ldots, \mathbf{h}_\ell$ and a single visible layer $\mathbf{v}$. Here, we denote by $\mathbf{W}_j$ the weight matrix between the previous layer $j-1$ and the actual layer j.

DBN layers consist of stochastic binary variables (called latent variables) having weighted connections that play a role of feature detectors. The DBN's visible units could be binary or real. Notice that there are no connections between units in the same layer as in RBM. In addition, there are connections between units in successive separate layers of DBNs (Fig. 6.13). This allows constructing more sparse connected DBNs. Moreover, the DBN is characterized by its hybrid nature that involves directed and undirected connections. Specifically, undirected connections are used between the top two layers and directed connections are between all other layers.

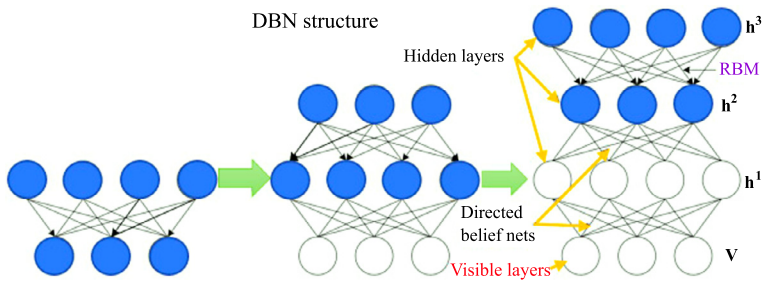


FIGURE 6.13 Schematic presentation of a Deep belief networks (DBN).

Essentially, a DBN that possesses ℓ hidden layers comprises ℓ weight matrices $(\mathbf{W}_1, \ldots, \mathbf{W}_\ell)$. It comprises also $\ell+1$ bias vectors, $\mathbf{b}_0, \ldots, \mathbf{b}_\ell$, where $\mathbf{b}_0$ represents the biases from the visible layer.

Hence, for the DBN, the joint distribution of the observed vector $\mathbf{x}$ and ℓ hidden layers $\mathbf{h}_k$ $(k = 1, \ldots, \ell)$ is calculated as [78]

$$P(\mathbf{x}, \mathbf{h}_1, \ldots, \mathbf{h}_\ell) = \left(\prod_{k=0}^{\ell-2} P(\mathbf{h}_k | \mathbf{h}_{k+1}) \right) \cdot P(\mathbf{h}_{\ell-1}, \mathbf{h}_\ell). \tag{6.31}$$

https://t.me/learningnets

Here $\mathbf{x} = \mathbf{h}_0$, the term $P(\mathbf{h}_k|\mathbf{h}_{k+1})$ denotes the conditional distribution of units of the layer k given the units of the layer $k+1$, and $P(\mathbf{h}_{\ell-1}, \mathbf{h}_\ell)$ is the joint distribution in the top-level RBM of the units in the layers $\ell - 1$ and ℓ.

At first, the process of constructing a DBN model for parameter-learning is done based on an unsupervised greedy layer-by-layer method that was demonstrated to be efficient in extracting important features from the input data [80,81]. In fact, this proved to be very effective to discover layer-by-layer complex nonlinearity. Specifically, the first layer is trained like an RBM model with $\mathbf{x} = \mathbf{h}^0$. It is used then to represent the input that will be used as input observation for the second layer. Now, the second layer is trained like an RBM by using the transformed data from the previous layer as training data. This process is repeated for the selected number of layers, each time propagating upward the transformed data. In practice, fine-tuning based on backpropagation and (stochastic) gradient descent is performed to optimize the DBN model. Then, the DBN model with the trained weights and biases can be used to predict new data [82–84].

Deeper DBN architecture can be obtained by incorporating more layers in the network. Generally speaking, deep architectures could help improve learning algorithms to achieve a more accurate expression of energy. In addition, it can reduce the training time because of a one-step can be sufficient to reach the learning of maximum likelihood [85].

6.4.4 Deep Boltzmann machine

Here, we discuss another type of deep generative model based on the Boltzmann machine called deep Boltzmann machine [72]. The DBM model is formed of several layers of hidden variables with undirected connections in contrast to RBMs that have just one (Fig. 6.14). The visible layer acquires the input data, and the hidden layers aim to extract features. Similarly to the RBM, the variables are mutually independent within each layer and conditioned on the variables in the precedent layers. Unlike DBN, this model is a fully undirected model. Essentially, the DBM model can be viewed as the result of stacking RBM to form a fully undirected graph. To handle real-valued data, in [86], a Gaussian–Bernoulli DBM (GDBM) that employed the Gaussian neurons in the visible layer of the DBM was introduced.

DBM has proven to be effective in automatically representing complex and nonlinear data and incorporating uncertainty related to ambiguous and missing or noisy inputs. In fact, every layer of DBM captures higher-order correlations between the hidden features in the precedent layer. DBMs are capable to learn complex statistical structures and have been applied to a variety applications including object recognition [87], document modeling [88], and computer vision [89].

Considering a two-layer DBM (Fig. 6.14) with a single visible layer, $\mathbf{v}$, and two hidden layers, $\{\mathbf{h}_1, \mathbf{h}_2\}$, the DBM energy function of the state $\{\mathbf{v}, \mathbf{h}_1, \mathbf{h}_2\}$ in

https://t.me/learningnets

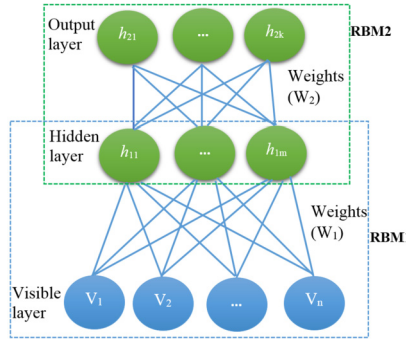


FIGURE 6.14 Illustration of DBM structure with two stacked RBMs.

this case is expressed as

$$E(\mathbf{v}, \mathbf{h}_1, \mathbf{h}_2; \boldsymbol{\theta}) = -\mathbf{v}^T \mathbf{W}_1 \mathbf{h}_1 - \mathbf{h}_1^T \mathbf{W}_2 \mathbf{h}_2, \tag{6.32}$$

where $\boldsymbol{\theta} = \{\mathbf{W}_1, \mathbf{W}_2\}$ refers to the model parameters, the vector of visible units $\mathbf{v} \in \{0, 1\}^D$ and the vectors of hidden units $\mathbf{h}_1, \mathbf{h}_2 \in \{0, 1\}^P$. For the sake of simplicity, the bias parameters are omitted below. The joint probability is defined as

$$P(\mathbf{v}; \boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} \sum_{\mathbf{h}_1} \sum_{\mathbf{h}_2} \exp(-E(\mathbf{v}, \mathbf{h}_1, \mathbf{h}_2; \boldsymbol{\theta})). \tag{6.33}$$

In the general case of DBM with one visible layer and L hidden layers, the joint probability is expressed as:

$$P(\mathbf{v}, \boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} \sum_{\mathbf{h}_1} \cdots \sum_{\mathbf{h}_L} \exp(-E(\mathbf{v}, \mathbf{h}_1, \ldots, \mathbf{h}_L, \boldsymbol{\theta})), \tag{6.34}$$

where $\boldsymbol{\theta} = \{\mathbf{W}_1, \ldots, \mathbf{W}_L\}$ refers to the parameters of the DBM model and the energy function is given by

$$E(\mathbf{v}, \mathbf{h}_1, \ldots, \mathbf{h}_L, \boldsymbol{\theta}) = -\mathbf{v}^T \mathbf{W}_1 \mathbf{h}_1 - \sum_{l=2}^{L} \mathbf{h}_{l-1}^T \mathbf{W}_l \mathbf{h}_l. \tag{6.35}$$

In [72], a greedy layer-by-layer pretraining algorithm for the DBM was considered by making each successive pair of layers in the DBM as an RBM.

6.4.4.1 *Deep stacked autoencoder*

A stacked autoencoder (SAE) model is an unsupervised deep-learning method composed of several layers, each of which is an autoencoder (Fig. 6.15). The output of each layer is the input of the next layer. The encoding process is performed by encoding every layer in a forwarding order, and the decoding process

https://t.me/learningnets

is done in the reverse order. The SAE model is trained based on a layer-by-layer greedy training procedure. This means that the first autoencoder is trained using the input data and produces the learned feature vector. The extracted features are fed to the next layer; this process is repeated until the training is completed. SAE models have been frequently used in different applications, including gearbox fault diagnosis [52], image denoising [55,56], and obstacle detection in autonomous vehicle [14].

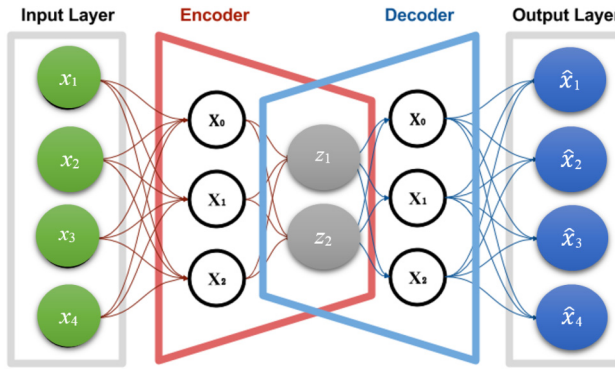


FIGURE 6.15 Diagrammatic illustration of a stacked autoencoder.

6.5 Deep learning-based clustering schemes for process monitoring

As mentioned above, anomaly detection in high-dimensional data based on binary clustering procedures is challenging and time-consuming. For instance, OCSVM with a selected kernel function could encounter challenges to learn from complex datasets, whereas it can get robust decision hyperplanes when applied to reduced and well-structured features. To overcome this problem, here we present a hybrid framework by combining the advantages of deep learning models and unsupervised clustering procedures (Fig. 6.16). In this hybrid approach, an unsupervised deep learning model, such as DBN, DBM, and SAE, will be used to extract relevant features from complex and high-dimensional datasets. The considered deep learning model will be trained in such a way that the used clustering algorithm (e.g., OCSVM) can efficiently discriminate normal data from anomalies based on the learned feature space.

Anomaly detection using hybrid methods can be done in two steps. The deep learning models are constructed through unlabeled training datasets that are devoid of anomalies. Then, the extracted features from the last layer of the considered deep-learning model are passed to the used clustering or one-class machine learning algorithm for training purposes. In the testing step, the learned model is used to extract features from new arrival unlabeled data, and the clus-

https://t.me/learningnets

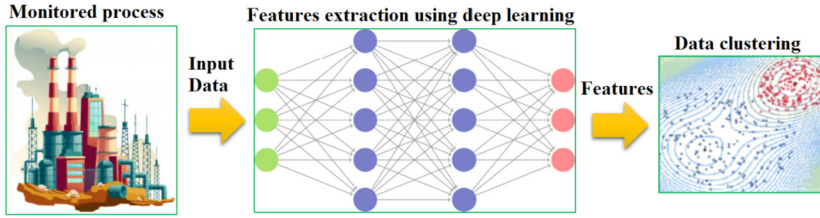


FIGURE 6.16 Deep learning-based clustering schemes for process monitoring.

tering approach is applied to extracted features to separate between normal and anomalies data points.

Several hybrid methods have been developed to benefit from the complementary advantages of deep learning models and unsupervised binary clustering procedures. The method in [90] uses the DBN model and the OCSVM to detect anomalies in an actual wastewater treatment plant. It exhibits good detection performance in comparison to other hybrid methods including DBN-based KNN and k-means algorithms. Also, in [91], several hybrid models including DBNs-based, DSA-based and RBM-based clustering methods (i.e., k-means, BIRCH, and expectation maximization) have been compared when applied to abnormal ozone measurements. The results in [91] showed that hybrid models achieved better results in comparison to the stand-alone binary clustering algorithms. Also, it has been shown that the DBN-OCSVM outperformed the other hybrid models. Here, the DBN is constructed as a dimensionality-reduction procedure to generate a lower-dimensional set of features. The computed features are used as input to train the OCSVM. Subsequently, the established hybrid DBN-OCSVM is used for testing. Such methods are assumption-free and do not require any data labeling or assumption related to the distribution underlying the datasets. A hybrid anomaly detection algorithm merging DBM and AE models is employed for obstacle detection in road environments based on stereovision [13]. The extracted features form this hybrid model are used by OCSVM for anomaly detection. This model merges the greedy learning features of DBM with the AE's dimensionality reduction capability. Results indicate that anomaly detection performance based on hybrid models is improved compared to the stand-alone models. In summary, coupling the extraction features capacity of deep learning models with the sensitivity to changes of the binary clustering methods is advantageous since it bypasses the complexity issues of clustering methods in particular when applied to large-scale datasets.

6.6 Discussion

This chapter reviewed the basic features of commonly used clustering algorithms and one-class algorithms (i.e., OCSVM and SVDD). Then, some of deep learning approaches that effectively analyze and model high-dimensional data,

https://t.me/learningnets

such as DBN, RBM and DSA, were presented. To bypass the shortcoming of binary clustering methods when applied to high-dimensional data, we briefly presented hybrid methods that combine the benefits of deep learning models with the clustering approaches to reveal faults in monitored processes. Essentially, extracting relevant and compact information from large datasets using deep learning models improves the detection and discrimination capacity of binary clustering methods.

Note that the above discussed deep learning models have been created using a single scale (i.e., time scale) as they relate process variables only at the scale of the sampling interval. This single-scale feature extraction is suitable for the data including contributions at one scale only. However, data from engineering and environmental processes are multiscale in nature. This is mainly because events can occur with different localization in time and frequency, and process variables can be gathered at different sampling rates or include missing values. Wavelet-based multiresolution is a powerful tool for the multiscale representation of data. Thus, in the future, an interesting research direction would be to develop multiscale deep-learning models and use them to further improve anomaly detection in complex processes.

References

[1] D.M. Tax, R.P. Duin, Support vector data description, Machine Learning 54 (1) (2004) 45–66.
[2] D. Wang, D.S. Yeung, E.C. Tsang, Structured one-class classification, IEEE Transactions on Systems, Man and Cybernetics. Part B. Cybernetics 36 (6) (2006) 1283–1295.
[3] B. Schölkopf, A.J. Smola, F. Bach, et al., Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond, MIT Press, 2002.
[4] S. Erfani, M. Baktashmotlagh, S. Rajasegarar, S. Karunasekera, C. Leckie, R1SVM: a randomised nonlinear approach to large-scale anomaly detection, in: Twenty-Ninth AAAI Conference on Artificial Intelligence, 2015, pp. 432–438.
[5] F.J. Huang, Y. LeCun, Large-scale learning with SVM and convolutional for generic object categorization, in: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), vol. 1, IEEE, 2006, pp. 284–291.
[6] D. Liu, H. Qian, G. Dai, Z. Zhang, An iterative SVM approach to feature selection and classification in high-dimensional datasets, Pattern Recognition 46 (9) (2013) 2531–2537.
[7] S. Vempati, A. Vedaldi, A. Zisserman, C. Jawahar, Generalized RBF feature maps for efficient detection, in: BMVC, 2010, pp. 1–11.
[8] Y. Bengio, Y. LeCun, et al., Scaling learning algorithms towards AI, in: Large-Scale Kernel Machines, vol. 34(5), 2007, pp. 1–41.
[9] Y. Bengio, et al., Learning deep architectures for AI, Foundations and Trends® in Machine Learning 2 (1) (2009) 1–127.
[10] B. Schölkopf, A. Smola, K.-R. Müller, Nonlinear component analysis as a kernel eigenvalue problem, Neural Computation 10 (5) (1998) 1299–1319.
[11] S.T. Roweis, L.K. Saul, Nonlinear dimensionality reduction by locally linear embedding, Science 290 (5500) (2000) 2323–2326.
[12] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, T. Darrell, DECAF: a deep convolutional activation feature for generic visual recognition, in: International Conference on Machine Learning, 2014, pp. 647–655.
[13] A. Dairi, F. Harrou, M. Senouci, Y. Sun, Unsupervised obstacle detection in driving environments using deep-learning-based stereovision, Robotics and Autonomous Systems 100 (2018) 287–301.

https://t.me/learningnets

[14] A. Dairi, F. Harrou, Y. Sun, M. Senouci, Obstacle detection for intelligent transportation systems using deep stacked autoencoder and k-nearest neighbor scheme, IEEE Sensors Journal 18 (12) (2018) 5122–5132.
[15] J. MacQueen, et al., Some methods for classification and analysis of multivariate observations, in: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Oakland, CA, USA, vol. 1(14), 1967, pp. 281–297.
[16] H.-S. Park, C.-H. Jun, A simple and fast algorithm for k-medoids clustering, Expert Systems with Applications 36 (2) (2009) 3336–3341.
[17] L. Kaufman, P.J. Rousseeuw, Finding Groups in Data: an Introduction to Cluster Analysis, vol. 344, John Wiley & Sons, 2009.
[18] R.T. Ng, J. Han, CLARANS: a method for clustering objects for spatial data mining, IEEE Transactions on Knowledge and Data Engineering 14 (5) (2002) 1003–1016.
[19] D. Boley, M. Gini, R. Gross, E.-H.S. Han, K. Hastings, G. Karypis, V. Kumar, B. Mobasher, J. Moore, Partitioning-based clustering for web document categorization, Decision Support Systems 27 (3) (1999) 329–341.
[20] A.K. Jain, Data clustering: 50 years beyond k-means, Pattern Recognition Letters 31 (8) (2010) 651–666.
[21] M. Verma, M. Srivastava, N. Chack, A.K. Diswar, N. Gupta, A comparative study of various clustering algorithms in data mining, International Journal of Engineering Research and Applications (IJERA) 2 (3) (2012) 1379–1384.
[22] D. Arthur, S. Vassilvitskii, k-means++: the advantages of careful seeding, in: Proceedings of the Eighteenth Annual ACM–SIAM Symposium on Discrete Algorithms, Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035.
[23] S.C. Johnson, Hierarchical clustering schemes, Psychometrika 32 (3) (1967) 241–254.
[24] B. Lorbeer, A. Kosareva, B. Deva, D. Softić, P. Ruppel, A. Küpper, Variations on the clustering algorithm BIRCH, Big Data Research 11 (2018) 44–53.
[25] T. Zhang, R. Ramakrishnan, M. Livny, BIRCH: an efficient data clustering method for very large databases, ACM Sigmod Record 25 (2) (1996) 103–114.
[26] S. Guha, R. Rastogi, K. Shim, CURE: an efficient clustering algorithm for large databases, ACM Sigmod record 27 (2) (1998) 73–84.
[27] S. Guha, R. Rastogi, K. Shim, Rock: a robust clustering algorithm for categorical attributes, Information Systems 25 (5) (2000) 345–366.
[28] G. Karypis, E.-H. Han, V. Kumar, Chameleon: hierarchical clustering using dynamic modeling, Computer 32 (8) (1999) 68–75.
[29] C.K. Reddy, B. Vinzamuri, A survey of partitional and hierarchical clustering algorithms, in: Data Clustering, Chapman and Hall/CRC, 2018, pp. 87–110.
[30] F. Murtagh, A survey of recent advances in hierarchical clustering algorithms, The computer journal 26 (4) (1983) 354–359.
[31] G. Carlsson, F. MÃŠmoli, Characterization, stability and convergence of hierarchical clustering methods, Journal of Machine Learning Research 11 (Apr 2010) 1425–1470.
[32] A.K. Jain, R.C. Dubes, Algorithms for Clustering Data, Prentice-Hall, Inc., 1988.
[33] H.-P. Kriegel, P. Kröger, J. Sander, A. Zimek, Density-based clustering, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 1 (3) (2011) 231–240.
[34] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, et al., A density-based algorithm for discovering clusters in large spatial databases with noise, in: KDD, vol. 96(34), 1996, pp. 226–231.
[35] M. Ankerst, M.M. Breunig, H.-P. Kriegel, J. Sander, OPTICS: ordering points to identify the clustering structure, ACM Sigmod record 28 (2) (1999) 49–60.
[36] D. Comaniciu, P. Meer, Mean shift: a robust approach toward feature space analysis, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (5) (2002) 603–619.
[37] K.-L. Wu, M.-S. Yang, Mean shift-based clustering, Pattern Recognition 40 (11) (2007) 3035–3052.
[38] Y. Chen, L. Tu, Density-based clustering for real-time stream data, in: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2007, pp. 133–142.

https://t.me/learningnets

[39] K. Fukunaga, Introduction to Statistical Pattern Recognition, Elsevier, 2013.
[40] G. Beni, X. Liu, A least biased fuzzy clustering method, IEEE Transactions on Pattern Analysis and Machine Intelligence 16 (9) (1994) 954–960.
[41] J. Han, J. Pei, M. Kamber, Data Mining: Concepts and Techniques, Elsevier, 2011.
[42] T. Cheng, F. Harrou, Y. Sun, T. Leiknes, Monitoring influent measurements at water resource recovery facility using data-driven soft sensor approach, IEEE Sensors Journal 19 (1) (2018) 342–352.
[43] X. Wu, V. Kumar, J.R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G.J. McLachlan, A. Ng, B. Liu, S.Y. Philip, et al., Top 10 algorithms in data mining, Knowledge and Information Systems 14 (1) (2008) 1–37.
[44] F. Harrou, B. Taghezouit, Y. Sun, Improved kNN-based monitoring schemes for detecting faults in PV systems, IEEE Journal of Photovoltaics 9 (3) (2019) 811–821.
[45] A.P. Dempster, N.M. Laird, D.B. Rubin, Maximum likelihood from incomplete data via the em algorithm, Journal of the Royal Statistical Society, Series B, Methodological 39 (1) (1977) 1–22.
[46] B. Schölkopf, J.C. Platt, J. Shawe-Taylor, A.J. Smola, R.C. Williamson, Estimating the support of a high-dimensional distribution, Neural Computation 13 (7) (2001) 1443–1471.
[47] S.-W. Lee, J. Park, S.-W. Lee, Low resolution face recognition based on support vector data description, Pattern Recognition 39 (9) (2006) 1809–1812.
[48] J. Park, D. Kang, J. Kim, J.T. Kwok, I.W. Tsang, SVDD-based pattern denoising, Neural Computation 19 (7) (2007) 1919–1938.
[49] A. Banerjee, P. Burlina, R. Meth, Fast hyperspectral anomaly detection via SVDD, in: 2007 IEEE International Conference on Image Processing, vol. 4, IEEE, 2007, pp. 101–104.
[50] E.J. Pauwels, O. Ambekar, One class classification for anomaly detection: support vector data description revisited, in: Industrial Conference on Data Mining, Springer, 2011, pp. 25–39.
[51] Y. Bengio, A. Courville, P. Vincent, Representation learning: a review and new perspectives, IEEE Transactions on Pattern Analysis and Machine Intelligence 35 (8) (2013) 1798–1828.
[52] G. Liu, H. Bao, B. Han, A stacked autoencoder-based deep neural network for achieving gearbox fault diagnosis, Mathematical Problems in Engineering 2018 (2018).
[53] E. Li, P. Du, A. Samat, Y. Meng, M. Che, Mid-level feature representation via sparse autoencoder for remotely sensed scene classification, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing 10 (3) (2016) 1068–1081.
[54] G.E. Hinton, R.S. Zemel, Autoencoders, minimum description length and Helmholtz free energy, in: Advances in Neural Information Processing Systems, 1994, pp. 3–10.
[55] P. Vincent, H. Larochelle, Y. Bengio, P.-A. Manzagol, Extracting and composing robust features with denoising autoencoders, in: Proceedings of the 25th International Conference on Machine Learning, 2008, pp. 1096–1103.
[56] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.-A. Manzagol, Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion, Journal of Machine Learning Research 11 (Dec 2010) 3371–3408.
[57] A. Krizhevsky, G.E. Hinton, Using very deep autoencoders for content-based image retrieval, in: ESANN, vol. 1, Citeseer, 2011, p. 2.
[58] D.P. Kingma, M. Welling, Auto-encoding variational Bayes, preprint, arXiv:1312.6114, 2013.
[59] C. Doersch, Tutorial on variational autoencoders, preprint, arXiv:1606.05908, 2016.
[60] D.J. Rezende, S. Mohamed, D. Wierstra, Stochastic backpropagation and variational inference in deep latent Gaussian models, in: International Conference on Machine Learning, vol. 2, 2014.
[61] C. Lu, Z.-Y. Wang, W.-L. Qin, J. Ma, Fault diagnosis of rotary machinery components using a stacked denoising autoencoder-based health state identification, Signal Processing 130 (2017) 377–388.
[62] S.-Z. Su, Z.-H. Liu, S.-P. Xu, S.-Z. Li, R. Ji, Sparse auto-encoder based feature learning for human body detection in depth image, Signal Processing 112 (2015) 43–52.

https://t.me/learningnets

[63] L. Deng, A tutorial survey of architectures, algorithms, and applications for deep learning, APSIPA Transactions on Signal and Information Processing 3 (2014).
[64] S. Rifai, P. Vincent, X. Muller, X. Glorot, Y. Bengio, Contractive auto-encoders: explicit invariance during feature extraction, in: International Conference on Machine Learning, 2011, pp. 833–840.
[65] S. Rifai, G. Mesnil, P. Vincent, X. Muller, Y. Bengio, Y. Dauphin, X. Glorot, Higher order contractive auto-encoder, in: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Springer, 2011, pp. 645–660.
[66] G.E. Hinton, Boltzmann machine, Scholarpedia 2 (5) (2007) 1668.
[67] D.H. Ackley, G.E. Hinton, T.J. Sejnowski, A learning algorithm for Boltzmann machines, Cognitive Science 9 (1) (1985) 147–169.
[68] G.E. Hinton, T.J. Sejnowski, et al., Learning and relearning in Boltzmann machines, in: Parallel Distributed Processing: Explorations in the Microstructure of Cognition, vol. 1(2), 1986, pp. 282–317.
[69] P. Smolensky, Information processing in dynamical systems: foundations of harmony theory, Tech. Rep., Colorado University at Boulder, Department of Computer Science, 1986.
[70] Y. Freund, D. Haussler, A fast and exact learning rule for a restricted class of Boltzmann machines, Advances in Neural Information Processing Systems 4 (1992) 912–919.
[71] K.H. Cho, T. Raiko, A. Ilin, Gaussian–Bernoulli deep Boltzmann machine, in: The 2013 International Joint Conference on Neural Networks (IJCNN), IEEE, 2013, pp. 1–7.
[72] R. Salakhutdinov, G. Hinton, Deep Boltzmann machines 5 (2) (2009) 448–455.
[73] A.-r. Mohamed, G. Hinton, G. Penn, Understanding how deep belief networks perform acoustic modelling, in: 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2012, pp. 4273–4276.
[74] U. Fiore, F. Palmieri, A. Castiglione, A. De Santis, Network anomaly detection with the restricted Boltzmann machine, Neurocomputing 122 (2013) 13–23.
[75] A. Fischer, C. Igel, Training restricted Boltzmann machines: an introduction, Pattern Recognition 47 (1) (2014) 25–39.
[76] C.P. Chen, C.-Y. Zhang, L. Chen, M. Gan, Fuzzy restricted Boltzmann machine for the enhancement of deep learning, IEEE Transactions on Fuzzy Systems 23 (6) (2015) 2163–2173.
[77] G.E. Hinton, Training products of experts by minimizing contrastive divergence, Neural Computation 14 (8) (2002) 1771–1800.
[78] G.E. Hinton, Learning multiple layers of representation, Trends in Cognitive Sciences 11 (10) (2007) 428–434.
[79] G.E. Hinton, To recognize shapes, first learn to generate images, Progress in Brain Research 165 (2007) 535–547.
[80] F. AlThobiani, A. Ball, et al., An approach to fault diagnosis of reciprocating compressor valves using Teager–Kaiser energy operator and deep belief networks, Expert Systems with Applications 41 (9) (2014) 4113–4122.
[81] Y. Chen, X. Zhao, X. Jia, Spectral–spatial classification of hyperspectral data based on deep belief network, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing 8 (6) (2015) 2381–2392.
[82] P. O'Connor, D. Neil, S.-C. Liu, T. Delbruck, M. Pfeiffer, Real-time classification and sensor fusion with a spiking deep belief network, Frontiers in Neuroscience 7 (2013) 178.
[83] H. Lee, P. Pham, Y. Largman, A.Y. Ng, Unsupervised feature learning for audio classification using convolutional deep belief networks, in: Advances in Neural Information Processing Systems, 2009, pp. 1096–1104.
[84] S. Kang, X. Qian, H. Meng, Multi-distribution deep belief network for speech synthesis, in: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, IEEE, 2013, pp. 8012–8016.
[85] G. Zhong, X. Ling, L.-N. Wang, From shallow feature learning to deep learning: benefits from the width and depth of deep architectures, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 9 (1) (2019) e1255.

https://t.me/learningnets

[86] R. Salakhutdinov, G. Hinton, Deep Boltzmann machines, in: Proceedings of the International Conference on Artificial Intelligence and Statistics, 2009.
[87] B. Leng, X. Zhang, M. Yao, Z. Xiong, A 3D model recognition mechanism based on deep Boltzmann machines, Neurocomputing 151 (2015) 593–602.
[88] N. Srivastava, R.R. Salakhutdinov, G.E. Hinton, Modeling documents with deep Boltzmann machines, preprint, arXiv:1309.6865, 2013.
[89] Q. Gan, C. Wu, S. Wang, Q. Ji, Posed and spontaneous facial expression differentiation using deep Boltzmann machines, in: Affective Computing and Intelligent Interaction (ACII), 2015 International Conference on, IEEE, 2015, pp. 643–648.
[90] F. Harrou, A. Dairi, Y. Sun, M. Senouci, Statistical monitoring of a wastewater treatment plant: a case study, Journal of Environmental Management 223 (2018) 807–814.
[91] F. Harrou, A. Dairi, Y. Sun, F. Kadri, Detecting abnormal ozone measurements with a deep learning-based strategy, IEEE Sensors Journal 18 (17) (2018) 7222–7232.

https://t.me/learningnets

Chapter 7

Unsupervised recurrent deep learning scheme for process monitoring

7.1 Introduction

Recent advances in network and data acquisition technologies have enabled the automatic collection of large datasets in engineering and environmental processes. Such advances have made multivariate time series measurements ubiquitous in everyday life, such as data outputs from wind turbines plants, the effluent measurements in wastewater treatment plants (WWTP), and air quality data, to name just a few. In addition, data on process operations may also be very useful for prediction and supervision of the quality of end products.

Handling and extracting the relevant features in multivariate time series data from real processes is challenging because of the dynamic dependencies between multiple variables. Vector autoregression (VAR) models are commonly used to capture correlations in multivariate time series data because of their simplicity [1,2]. As a natural extension of univariate AR models, VAR models fail to recognize the dependencies among output variables. To address this, a variety of VAR models have been designed, such as structured VAR models, to provide a better description of the dependencies between a large number of variables [3]. However, the main limitation of using VAR for a large dataset is its linear growth with increasing temporal window size, and its quadratic growth with the increasing number of variables. Thus, VAR models are unsuitable and prone to overfitting when handling long-term temporal patterns. In other studies, linear regression models such as support vector regression and ridge regression LASSO [4] models have been used for modeling multivariate time series data. Similarly to VARs, these modeling approaches are not suitable for observing nonlinear relationships between process variables. Gaussian processes (GP) can also be applied to model multivariate time series [5]. However, the use of GP models is restricted by their high computational complexity. For instance, implementing GP models to forecast multivariate time series can have cubic complexity in the number of samples because of the inversion of the kernel matrix.

Deep learning techniques have emerged in recent years as an efficient tool to extract pertinent information from large and complex datasets. Several deep learning techniques have been designed to address practical problems such as

https://doi.org/10.1016/B978-0-12-819365-5.00013-9
Copyright © 2021 Elsevier Inc. All rights reserved.
https://t.me/learningnets

pattern recognition and regression analysis and prediction [6–8]. Indeed, one of the major limitations of traditional neural networks is their feed-forward structure with no recurrence, which is inappropriate for handling input data with periodicity (sequence) and time dependencies. In analogy, the human brain, in order to better understand information from text, speech, or even video, can make use of both historical and actual received situations. In text or speech, for instance, previously received words help in understanding current words and predicting the following words. The output of the last cycle is used to feed the new cycle, forming a loop; hence the recurrent architecture. Recurrent neural networks' (RNNs) architecture benefits from the inner loop and state (memory) to maintain information, making them suitable to address time-dependent learning problems. They were originally designed for language models, due to their capacity to memorize long-term dependencies. RNNs are considered to be deep in time, which becomes evident when they are unfolded in time. This property is helpful for the training phase, especially for updating weights (gradients error) based on backpropagation through time (BPTT). RNNs have been widely and successfully exploited by researchers for a wide range of real applications. Simple RNNs can deal with input data containing short-term dependencies very well, but they are unfortunately not efficient at learning and discovering dependencies over time when historical information becomes too large. Specifically, when time lags become too large, gradients of RNNs may disappear through unfolding RNNs into very deep feed-forward neural networks. In short, RNNs usually fail to capture very long-term dependencies due to gradient vanishing.

A new extension of RNNs has been designed to address this problem, called long short-term memory network (LSTM) and gated recurrent unit (GRU), that incorporate two important mechanisms, the states (memory) and gate, into the conventional RNNs [9,10]. In LSTM and GRU, the memory cells have the ability to determine when certain information needs to be forgotten and determine the optimal time lags. In addition, gates provide a way to regulate the information flow-through, using a sigmoid neural network layer followed by a pointwise multiplication operation. RNNs-based algorithms have been demonstrated to be efficient in several applications, such as polyphonic music generation, intrusion detection, and gesture recognition.

In recent years, enhanced extensions have been designed by merging the desirable features of RNN and LSTM with the function to delineate complex distributions from restricted Boltzmann machines (RBM) and deep belief networks (DBN). Here, we provide a brief description of hybrid RNN-RBM, LSTM-RBM, and LSTM-DBM methods. These deep learning hybrid models have shown promising results in modeling dependency in time series data. The purpose of this chapter is to design a reliable and flexible anomaly/fault detection scheme to uncover anomalies in multivariate time series data. To this end, we integrated the powerful RNNs-based models with various clustering algorithms to uncover temporal dependencies in multivariate time series. We employed the RNNs models for modeling under normal conditions, and we then

https://t.me/learningnets

used the prediction errors to identify anomalies. We assessed the detection sensitivity of RNN, RBM, and hybrid models-based clustering algorithms using actual measurements from a coastal municipal WWTP.

The chapter is structured as follows. Section 7.2 presents foundational concepts of RNN, LSTM, and GRU models and describes the hybrid deep learning techniques, including RNN-RBM, LSTM-RBM, and LSTM-DBN. Section 7.3 presents the anomaly detection problem by integrating recurrent deep learning with binary clustering algorithms. Section 7.4 verifies the effectiveness of the studied approaches using real data, and conclusions are summarized in Sect. 7.5.

7.2 Recurrent neural networks approach

Accurate modeling of serially dependent sequences is important because data from engineering and environmental processes are inherently sequential. In traditional neural networks, such as feedforward neural networks, modeling or decision-making is based on the actual inputs, and no memory on the past is considered. In other words, these networks ignore past data and assume the nonexistence of dependencies since there are no cycle loops in the network. However, in practice, the decision at time step t can be affected by the output of the network at time step $t-1$. Thus, these models are not suitable for describing sequential dependent data. This limitation can be mitigated by incorporating the information from the actual and past measurements in the modeling process. RNNs can deal with temporal dependencies because their design permits the consideration of the actual input and also the previously received inputs, and the possibility of memorizing previous inputs due to its internal memory. Another important feature of the RNNs approach is the capability to predict and forecast the future of a sequence based only on historical data.

7.2.1 Basics of recurrent neural networks

As discussed above, RNNs are an efficient tool used to deal with complex and nonlinear dependencies in multivariate times series data. Generally speaking, RNNs are able to review information at each time point and choose the pertinent information to appropriately generate the outputs. RNNs can be trained to retain information in the long term through discovering features and modeling sequential (time) dependencies from the training dataset. The efficiency of RNNs has been proven over the last decade through several applications involving sequential or temporal data [11,10,12,13]. This success can be attributed to their ability to extract complex nonlinearity between data point in the training dataset and project them onto a new feature space. RNNs have been widely exploited in speech recognition, natural language processing, and machine translation.

The so-called simple RNNs, or vanilla RNNs, with a single hidden layer comprise a memory $\mathbf{h}$ that permits summarizing the past in order to predict the future. Generally speaking, RNNs predict the output $\mathbf{O}_t$ by using the input

https://t.me/learningnets

vector, $\mathbf{x}_t$, and the memory state, $\mathbf{h}_t$. The memory state $\mathbf{h}$ of VRNNs is updated with the recurrence formula

$$\mathbf{h}_t = \sigma(\mathbf{V}\mathbf{h}_{t-1} + \mathbf{U}\mathbf{x}_t), \tag{7.1}$$

where σ is the activation function, and matrices $\mathbf{V}$ and $\mathbf{U}$ are trained to properly update the history vector $\mathbf{h}$. Then, the predicted output $\mathbf{O}_t$ can be obtained by

$$\mathbf{O}_t = \mathbf{W}\mathbf{h}_t, \tag{7.2}$$

where the weight matrix $\mathbf{W}$ is trained to use the history $\mathbf{h}$ and predict the next output. The weights matrices, $\mathbf{V}$, $\mathbf{U}$, and $\mathbf{W}$, represent the internal parameters of the RNN. The dynamic behavior and prediction of the RNN changes by updating these internal parameters, which can be computed by backpropagation. RNNs uncover and learn temporal dependencies in time series data by using the former output as inputs in addition to the actual input and recent past inputs to generate new outputs, as shown in Fig. 7.1. The architecture of the RNN model was designed using nonlinear stacked units, where links between units form a directed cycle (Fig. 7.1).

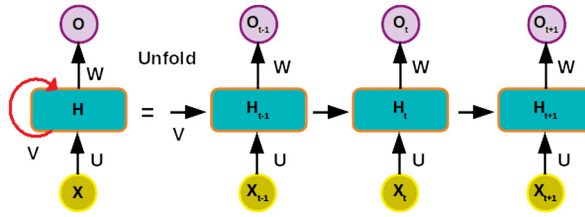


FIGURE 7.1 A basic illustrative representation of RNN.

Note that RNNs are very deep in time when unrolled [14]. The architecture of RNNs improves feature extraction and discovery of long-term dependencies from sequential time-series data. RNN is designed to model time varying or sequential patterns of input, where the input can have fixed or variable size, while the size of the RNN output is fixed. Of course, RNNs are characterized by feedback. RNN topology is represented as closed loop connections, equipped with a memory that captures and stores the information processed so far.

The ability of multiple mapping schemes is not supported in the traditional neural network based on the feed-forward mechanism; this kind of network architecture supports only a fixed input and output size. Another desirable property of RNNs is their capability to handle input variables with various sizes, which makes them very useful for operating over sequences of vectors. In other words, RNNs are able to map input sequences to produce output sequences, where the length or size of the inputs depends on data nature and structure (Fig. 7.2). Below are a few examples to clarify this more concretely.

https://t.me/learningnets

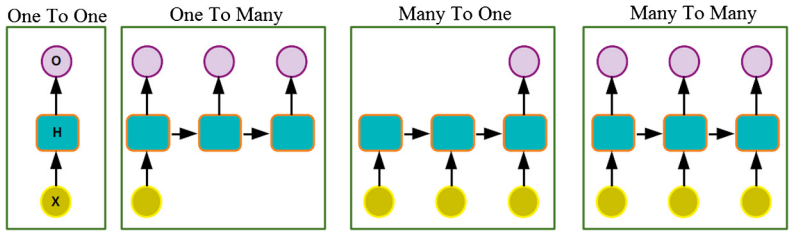


FIGURE 7.2 Different configurations of the RNN model.

Single to multiple. In the case of image captioning, the network takes the input pixels of an image and generates a sequence of words.

Multiple to single. Such a situation appears in sentiment analysis, where the input is a sequence of words, while for the output the network generates a single Boolean classification of *true or false* (i.e., positive or negative sentiments). Another example is in voice classification which is based on a sequence of voice records, and the network identifies the speaker.

Multiple to Multiple (a). This happens in the case of language translation, for instance, from English to Arabic.

Multiple to Multiple (b). This case is of video captioning and video classification.

RNNs have been widely exploited to capture relevant dependencies in time series. However, in training deep neural networks, RNNs can face two common problems of vanishing and exploding gradients, where the error gradients can increase to explosion or decrease to vanishing (close to zero). Indeed, the error gradients are employed for updating weights. The problem becomes more severe when increasing the depth of the network. Several techniques have been developed to alleviate the problem of exploding gradients: changing the network design to make it shallower, using a gradient clipping approach that aims to threshold the error gradients, and using weight regularization to reduce the severity through L_1 or L_2 penalties on the recurrent weights.

7.2.2 Long short-term memory

Machine learning has been researched extensively over the past three decades. Conventional neural networks are one such intensively used machine learning approach. As mentioned above, the main characteristics underlying these networks are the presence of full connections between adjacent layers and the absence of connections between the nodes within the same layer. Thus, this type of network is suited to handle sequential data and describe temporal dependencies in the data because it considers only the current measurement and without memorizing past measurements. As discussed above, to overcome this problem, the internal memory of an RNN has been used to handle sequential data by considering the actual and previously received measurements. In other

https://t.me/learningnets

words, the hidden units in an RNN receive feedback from the previous state to the current state. Because the depth of the RNN is the time span, information can be lost through time and the error can propagate back. Accordingly, the accuracy of the RNN can be degraded when the time span becomes longer due to the vanishing gradient and exploding gradient problems. To address this issue, long short-term memory (LSTM), which is an extended version of RNN, was first introduced by Hochreiter and Schmidhuber [15] in 1997. LSTM models have been largely utilized in many applications, including handwriting recognition [16], language modeling [17] and translation [18,19], acoustic modeling of speech [20], speech synthesis [21,22], protein structure prediction [23–25], and analysis of audio and video data [26–30]. Moreover, in the modern LSTM architecture, there are peephole connections between internal cells and the gates in the same cell for learning the accurate timing of the outputs [31]. This section first provides an overview of LSTMs and then discusses how they can be used for modeling and process monitoring. We also describe gated recurrent units (GRU), another improved version of RNNs.

The LSTMs are designed as an extension of simple RNNs to solve the vanishing gradient problem by explicitly incorporating a memory unit into the network. They are based on memories and gates making them suitable for learning long-term dependencies. Fig. 7.3 displays a schematic representation of an LSTM.

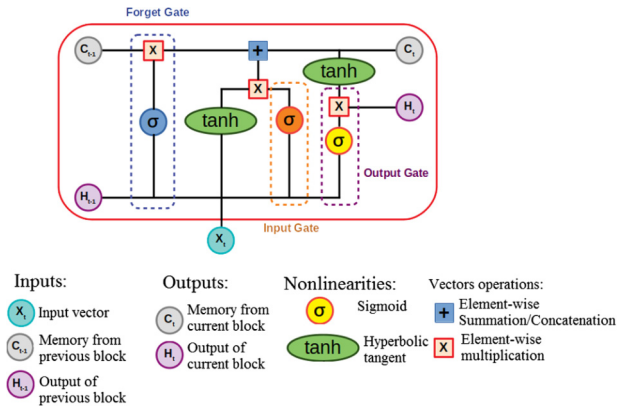


FIGURE 7.3 A basic illustrative representation of an LSTM unit.

The principal component of LSTM resides in its cell state, which is the horizontal chain shown in the top of the graph (Fig. 7.4). In LSTM, information can be removed or added to the cell by using structures called gates.

We designed a common LSTM model as a concatenation of several cell units instead of conventional neural network layers. We now briefly investigate the properties of the LSTM unit. As shown in Fig. 7.3, an LSTM cell comprises three inputs: X_t is the input observation at the current time point, h_{t-1} represents the output generated from the preceding LSTM cell, and C_{t-1} denotes

https://t.me/learningnets

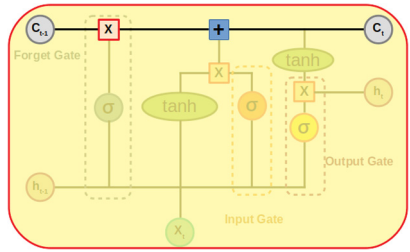


FIGURE 7.4 Cell state in an LSTM model.

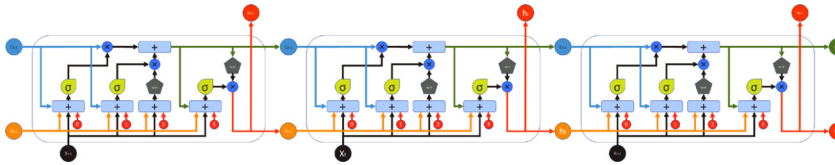


FIGURE 7.5 Schematic representation of an LSTM when unrolled in time.

the memory of the preceding cell. Also, each LSTM cell contains two outputs, h_t and C_t, which are the output of the actual network and the memory of the current unit, respectively. It is composed of three kinds of gates, namely the input gate, the forget gate, and the output gate (Fig. 7.3). Each gate comprises a sigmoid neural net layer and a pointwise multiplication operation. LSTMs can be seen when unfolded as a chain-like structure. In other words, it is a loop repeating module with a different structure (Fig. 7.5). Depending on the input and the internal feedback (memory state), the output will be generated in a special manner based on gates and four embedded layers structure. The layers here have activation units based on sigmoid and tanh.

This LSTM unit provides a decision based on the actual input, precedent output and memory, and then produces a new output and upgrades its memory.

Input gate The key role of the input gate is to control the flow of input activation into the memory cell. Here, the sigmoid layer controls the flow by generating an output between 0 and 1 into the memory cell.

Output gate The aim of the output gate is to control the output flow generated from cell activation to be injected into the network. This gate has the sigmoid layer that controls how much memory should be fed into the next LSTM unit.

Forget gate The sigmoid layer output removes information from the cell state that is no longer required (when the output is 0, otherwise it will keep it). In other words, the forget gate overcomes the weakness of LSTM models by preventing them from processing continuous input streams [32]. By scaling the content of block memory, it forgets the cell's memory content in an adaptive way.

https://t.me/learningnets

7.2.2.1 LSTM implementation steps

The input of each LSTM cycle is $(t-1)$th memory state C_{t-1} and hidden layer units h_{t-1} (output), but for the first cycle we start with zero or randomized values. The main steps when implementing LSTM are described below.

- The first step of LSTM consists of deciding which information is to be forgotten or retained in the particular time instant. This step is particularly useful when the past information is no longer useful for the actual cycle which is mainly related to the current input. In other words, the aim of this step is to reveal the information that is not needed and can be omitted from the cell state. This is achieved by the sigmoid function. It looks at the past state (h_{t-1}) and the actual input x_t and calculates the function accordingly (Fig. 7.6).

$$f_t = \sigma\left(x_t U^f + h_{t-1} W^f\right), \tag{7.3}$$

  where w_f is weight and h_{t-1} is the output from the previous time.

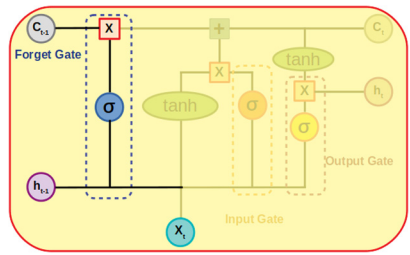


FIGURE 7.6 First step in LSTM: decide what information is to be forgotten or retained.

- The second step in the LSTM consists of updating the content of the memory cell. This step selects the new information that will be stored in the cell state (Fig. 7.7). The second layer (input gate) contains two parts, namely the sigmoid function and the tanh. The sigmoid layer chooses which values are to be updated. In the case of 1, the value of the input gate is unchanged, and in the case of 0 it is dropped. Next, a tanh layer creates a vector of new candidate values that can be added to the state. It provides weight to the selected values

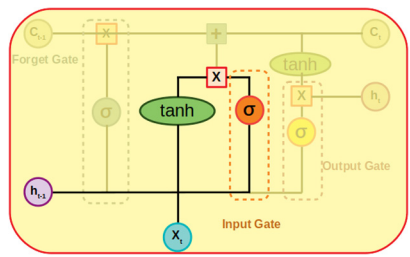


FIGURE 7.7 Second step in LSTM: updating the content of the memory cell.

https://t.me/learningnets

for their level of importance (−1 to 1). After that, the two are combined to update the state. The $\hat{C}$ is combined with C_{t-1} to update both the memory state C_t. The H_t (output) is computed according to the output gate sigmoid and the tanh of C_t:

$$i_t = \sigma\left(x_t U^i + h_{t-1} W^i\right), \tag{7.4}$$

$$\tilde{C}_t = \tanh\left(x_t U^g + h_{t-1} W^g\right), \tag{7.5}$$

$\tilde{C}_t$ is the candidate memory cell, W^i, W^g are weight parameters.

- In this step, the old cell state, C_{t-1}, is actualized with the new cell state, C_t, and the past cell state, C_{t-1}, is actualized with the new cell state, C_t (Fig. 7.8). To do so, the old state is multiplied by f_t to forget the irrelevant information, and the candidate, $\tilde{C}_t$, is added. This represents the new candidate values scaled by how much we choose to update every state value:

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t. \tag{7.6}$$

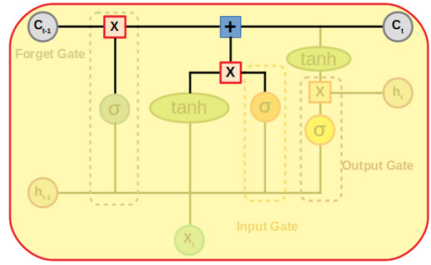


FIGURE 7.8 Step 3 in LSTM modeling.

- Finally, the output is computed in two steps (Fig. 7.9). First, a sigmoid layer is utilized to select the relevant portions of the cell state to be transmitted to the output (Eq. (7.7)):

$$o_t = \sigma\left(x_t U^0 + h_{t-1} W^0\right). \tag{7.7}$$

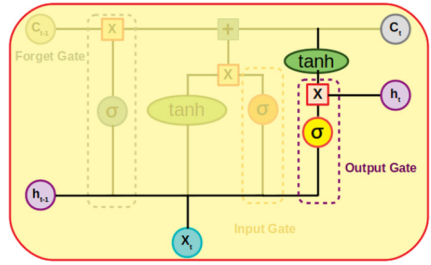


FIGURE 7.9 Step 4 in LSTM modeling.

https://t.me/learningnets

The cell state is then passed via tanh (for normalizing values within the range −1 and 1) and multiplied by the output of the sigmoid gate; thus we keep only the portions we selected to output (Eq. (7.8)):

$$h_t = \tanh(C_t) \cdot o_t. \tag{7.8}$$

At the end of the cycle, the H_t hidden layer units which represent the output of the cycle and the C_t memory state are ready for the next cycle usage. In summary, in the LSTM model, the three gates are trained for learning what information can be maintained in the memory, how long it can be stored, and when it can be read out. Combining several memory cells into blocks permits them to share the same gates, and hence the number of adaptive parameters is reduced.

7.2.3 Gated recurrent neural networks

In this section we present another extended version of RNN, called a gated recurrent unit (GRU), which was primarily introduced by Cho et al. [9]. GRU can be considered as a reduced version of LSTM. It is also designed to alleviate the vanishing gradient problem, which is the main weakness of the standard RNN. GRU is built without a cell state and comprises only two gates instead of three, as in LSTM [9]. GRU models have demonstrated their efficiency during the last decade in several applications involving sequential or temporal data [11,10,12, 13]. Their successful application comes from their ability to model complex nonlinearity between data points and extract relevant features from time series data.

In comparison with LSTM, GRU comprises two gates instead of three, i.e., update and forget (also called reset). Indeed, the update gate in GRU can be viewed as a combination of the forgetting gate and input gate in LSTM (Fig. 7.10). These two gates consist of a sigmoid layer and a pointwise multiplication operation represented by two vectors, resulting in values within the interval [0, 1]. Similar to LSTM, the impact of the previous historical information on the actual cycle is controlled by these gates. When the output of the reset gate is zero, the memory information is ignored; otherwise it will influence the evolution of the final response.

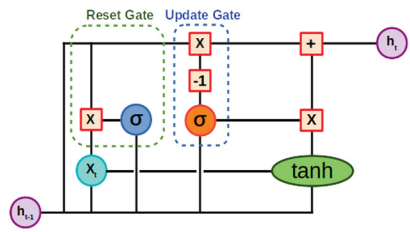


FIGURE 7.10 Gated recurrent unit.

https://t.me/learningnets

Generally speaking, the forget or reset gate determines how to merge the new input with the previous memory [10]. In other words, this gate is employed to decide the amount of historical information to quash:

$$r_t = \sigma\left(x_t U^r + h_{t-1} W^r\right). \tag{7.9}$$

On the other hand, the update gate, which is acting similarly to the forget and input gate of an LSTM, decides how much of the previous memory to keep and what new information to add:

$$z_t = \sigma\left(x_t U^z + h_{t-1} W^z\right). \tag{7.10}$$

The new hidden state output at the current time point merges current and previous time points and can be computed as

$$h_t = (1 - z_t) \cdot h_{t-1} + z_t \cdot \tilde{h}_t. \tag{7.11}$$

We can see that the complement of gate-update is utilized to decide the information to maintain from the h_{t-1} rather than creating a new gate for this purpose. The new candidate, $\tilde{h}_t$, for hidden layer output is computed as

$$\tilde{h}_t = \tanh\left(x_t U^h + (r_t \cdot h_{t-1}) W^h\right). \tag{7.12}$$

Here, r_t is used to manage what part of h_{t-1} is needed to incorporate when computing a new candidate. Table 7.1 summarizes the main features of LSTM and GRU models.

TABLE 7.1 LSTM versus GRU.

LSTM	GRU
Three gates	Two gates
Control the exposure of memory content (cell state)	Expose the entire cell state to other units in the network
Has separate input and forget gates	Performs both of these operations together via update gate
More parameters	Fewer parameters

7.3 Hybrid deep models

In practice, several data sequences are high-dimensional, such as words in a text and images in a video. Thus, using only received observations at the previous time points cannot guarantee a good prediction of the expected value at the next time point. With such high-dimensional objects, using RNNs alone to uncover correlated patterns would be very expensive. To overcome this issue, energy-based models such as the Boltzmann machine (RBM) are integrated into

https://t.me/learningnets

RNN models. This section presents some hybrid deep learning models that combine variants of recurrent neural networks for temporal modeling and shallow or deep probabilistic and generative models to learn complex and hierarchical features with a high level of representation. Specifically, we discuss three flexible and efficient models to capture temporal dependencies in a multivariate setting: RNN-RBM, LSTM-RBM, and LSTM-DBN architectures. These architectures were first designed in the context of polyphonic music generation and transcription [33–36].

7.3.1 RNN-RBM

Accurate modeling of sequences plays a core role in several real applications where collected data is inherently sequential, such as text, speech, and videos. Extracting the relevant features from these data sources is useful for solving discrimination prediction and anomaly detection tasks. Indeed, since an anomaly is a deviation from the expected value from the reference model, good modeling facilitates the anomaly detection.

As hybrid deep models, the recurrent temporal RBM (RTRBM), RNN-RBM and LSTM-RBM and DBN have proven their ability to learn dependency in temporal patterns. Before we go into details of hybrid models, let us first present the most simple one, i.e., RTRBM model, which is a probabilistic model. The structure of this hybrid model is displayed in Fig. 7.11, where in each RNN time step, the internal feedback is passed to the RBM in order generate an output that feeds the next cycle of the model. Generally speaking, RTRBM is a succession of conditional RBMs (Fig. 7.11), and each RBM contains a hidden state that is obtained from the precedent RBM and employed to modulate its hidden units bias [33]. At every time point, the learning algorithm tries to extract relevant features through a conditional RBM using the contrastive divergence (CD) learning approach. RTRBM has shown a good ability in describing complex probability distributions over high-dimensional sequences [33].

In the RTRBM model, the parameters are time dependent. The RTRBM parameters are b_v, b_h, $W^{(t)}$, and

$$\mathcal{M}^{(t)} \equiv \{v^{(\tau)}, \hat{h}^{(\tau)} | \tau < t\} \tag{7.13}$$

where $\hat{h}^{(t)}$ represents the value of the mean-field of $h^{(t)}$, and the biases are related to $\hat{h}^{(t-1)}$. The joint probability distribution of RTRBM is defined by [33]

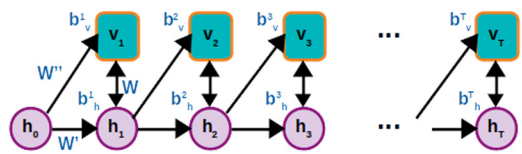


FIGURE 7.11 Diagram of a recurrent temporal RBM.

https://t.me/learningnets

$$P(v^{(t)}, h^{(t)}) = \prod_{t=1}^{T} P(v^{(t)}, h^{(t)} | \mathcal{M}^{(t)}). \tag{7.14}$$

In Eq. (7.14), $P(v^{(t)}, h^{(t)} | \mathcal{M}^{(t)})$ is the joint probability of the tth RBM, with Eqs. (7.15) and (7.16) defining its parameters:

$$b_h = b_h + W' \hat{h}^{(t-1)}, \tag{7.15}$$

$$b_v = b_v + W'' \hat{h}^{(t-1)}. \tag{7.16}$$

RTRBM contains six parameters: $W, b_v, b_h, W', W'', \hat{h}^{(0)}$. Note that Eq. (7.17) is defined as a single-layer RNN, with hidden units $\hat{h}^{(0)}$,

$$\hat{h}^{(t)} = \sigma(W v^{(t)} + b_h) = \sigma(W v^{(t)} + W' \hat{h}^{(t-1)} + b_h). \tag{7.17}$$

Note that RTRBM is a succession of conditional RBMs whose parameters are obtained as the deterministic RNN output constrained by the hidden units required to represent the conditional distributions. Several hybrid models have been designed to bypass this constraint by merging a complete RNN with distinct hidden units, such as RNN-RBM and LSTM-based RBM and DBN.

7.3.2 RNN-RBM method

The RNN-RBM model was introduced as a natural extension of RTRBM [33] by combining the RNN and RBM models [37]. RNN-RBM is an unsupervised generative model like traditional RBM, which means that it is suitable for directly modeling the probability distribution of a training dataset without the need for data labeling. We aimed to further exploit the prediction capacity of the RNN and RBM models and design a flexible model, allowing to uncover and predict the temporal dependencies in high-dimensional data [37]. As an energy-based model, the principle idea of RNN-RBM is to extend the RNN model by including an RBM at every time step [37]. The architecture is designed using an RBM whose parameters are computed from an RNN (Fig. 7.12). In other words, in the RNN-RBM model, the output layer of the RNN is laying the groundwork for the parameters for the RBM model, as illustrated in Fig. 7.12. Graphically, in the RNN-RBM architecture, the lower layer represents the RNN and the upper two layers represent the RBM model (Fig. 7.12). The RNN-RBM model comprises nine parameters: W, b_v and b_h, $\hat{h}^{(0)}$, W_2 (RBM), W_3 and $b_{\hat{h}}$ (RNN model), W' and W'' to link the two models. Usually, the matrices W, W_2, W_3, W' and W'' are initialized to small random normalized values, while b_v, b_h and $\hat{h}^{(0)}$ are zero-initialized.

Note that when combining RNN with RBM, the new hybrid model becomes deep in time when unfolded. An unfolded RNN-RBM can be represented as a series or chain of RBMs, where each time step instead of generating an output from sigmoid or tanh layer we invoke the learning of probability distributions

https://t.me/learningnets

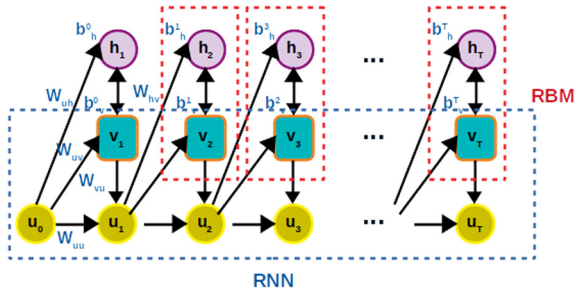


FIGURE 7.12 Diagram of a single-layer RNN-RBM.

of an input coming from recurrent connection, providing past information to the current time step. This will have a direct impact on the output of the hybrid model since its biased with the historical information, which means that the RBM parameters are determined by an RNN.

The hidden unit $\hat{h}^{(t)}$ of the RNN layer at time t is connected to its predecessor $\hat{h}^{(t-1)}$ and to V^t and is computed as [37]:

$$\hat{h}^{(t)} = f(W_2 v^{(t)} + W_3 \hat{h}^{(t-1)} + b_{\hat{h}}), \tag{7.18}$$

where the activation function, f, is usually taken as the σ function, as proposed in [37]. Also, the tanh function is used as the activation function [38]. By placing $\hat{h}^{(t)}$ in Eq. (7.14), we can compute the joint probability distribution of RNN-RBM. Indeed, RTRFM is constrained by the hidden units required for representing conditional distributions and for transmitting temporal information. This restriction could be alleviated by merging a full RNN with distinct hidden units $\hat{h}^{(t)}$. The model is trained by performing the following steps:

1. Use Eq. (7.18) to generate the hidden unit $\hat{h}^{(t)}$ of the RNN layer
2. Compute the parameters of RBM (i.e., $b_v^{(t)}$ and $b_h^{(t)}$) via Eqs. (7.15) and (7.16) for $\hat{h}^{(t-1)}$ and realize n-step Gibbs sampling to generate a representation of the visible units $v^{(t)*}$
3. Utilize the contrastive divergence (CD) technique described in [39] to compute the log-likelihood gradient with respect to W, $b_v^{(t)}$ and $b_h^{(t)}$
4. Compute the gradient with respect to W, W_2, W_3, W', W'', b_v, b_h, and $\hat{h}^{(0)}$ by propagating the gradient with respect to $b_v^{(t)}$, $b_h^{(t)}$ backwards in time

7.3.3 LSTM-RBM model

This section is dedicated to an overview of a gated recurrent network which is deep in time with a shallow neural network composed of two layers. More specifically, we present another hybrid model called LSTM-RBM [34], which can be viewed as of the RNN-RBM proposed in [37], where the RNN units are changed by LSTM units, as illustrated in Fig. 7.13. LSTM-RBM combines (1)

https://t.me/learningnets

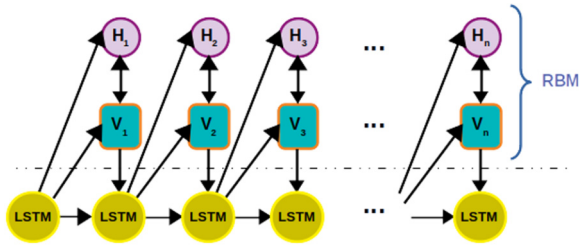


FIGURE 7.13 Diagram of an LSTM-RBM model.

the effectiveness of LSTM equipped with a separate memory component and gates mechanism for capturing long term temporal dependencies in time series, with (2) the capacity of RBM to approximate unknown data distribution.

Furthermore, the LSTM-RBM can be understood as an extension of the recurrent temporal RBM (RTRBM), which combines a simple RNN cell and RBM in the same architecture [34]. As RNN is used in this combination, the RTRBM is not suited to memorize a long historical range of temporal data dependencies. Thus, LSTM-RBM was designed to overcome this problem by permitting the hybrid model to be more flexible and robust. The memory state, also known as internal feedback, of the recurrent neural networks makes them a suitable technique for modeling sequence or temporal dependencies, while, on the other hand, the hybrid model is able to generate new sequences. As discussed above, LSTM is designed to capture short-term memory with the possibility of deciding what information in the memory is irrelevant and must be replaced with new information by using the forget gate layer. The RBM make the sequence generation possible because of its nature as a probabilistic model used as a block to build a deep learning model such as deep belief networks (DBN) and deep Boltzmann machine (DBM).

The RBM part in the LSTM-RBM model comprises three parameters: the weight matrix, W, and the bias vectors for the hidden and visible layers, $b_h(t)$ and $b_v(t)$, at time step t. The RBM biases are computed at every time point based on the LSTM unit at the preceding time point, which can be understood as the LSTM transmitting temporal information. The RBM's hidden and visible layer biases are updated via LSTM at time point t as

$$b_h(t) = b_h + \mathbf{W}' S_t, \tag{7.19}$$

$$b_v(t) = b_v + \mathbf{W}'' S_t, \tag{7.20}$$

where S_t is the external state of LSTM hidden unit U_t.

7.3.4 LSTM-DBN

Several hybrid recurrent network models have recently been reported to suitably handle dependencies in multivariate time series data. In this section, we present

https://t.me/learningnets

another sophisticated hybrid model called the deep belief network–long short-term memory (DBN-LSTM) network, which is more complicated and deeper than the above presented models [36]. This model integrates the ability of DBN with improved learning features for uncovering relevant nonlinear features, and LSTM to appropriately describe the temporal dependencies (Fig. 7.14). The DBN-LSTM model permits a simultaneous deeper description of time series data and tracking of temporal information in data. Indeed, DBN-LSTM can be viewed as an improved version of the RNN-DBN model proposed in [35]. Using an LSTM instead of RNN enables an efficient modeling of temporal dependencies across large time points. In other words, this guarantees that the DBN-LSTM model keeps information on the sequence produced for a longer period.

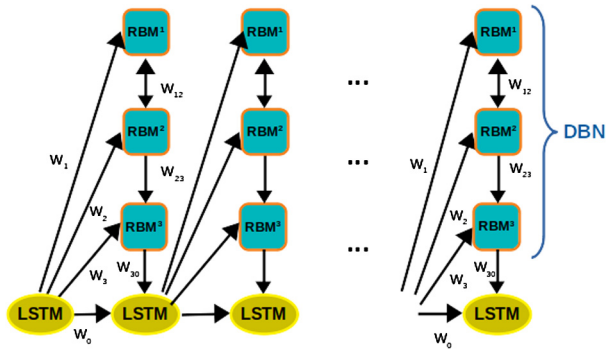


FIGURE 7.14 LSTM-DBN.

Combining the capacity of LSTM in modeling long term temporal dependencies and the improved learning capacity of DBN would result in more expressive RNNs. In the DBN-LSTM architecture, we can see that DBN essentially interacts with LSTM in two points. The first point of interaction is between the visible layer of DBN, v, which feeds the input to the LSTMs, as illustrated in the following equation:

$$q_t = \phi(b_{qt}) + W_{vq} v_t + W_{qq} q_{t-1}. \tag{7.21}$$

The second interaction is created where the biases of the visible and hidden units of the DBN are computed as

$$b_{vt} = b_{v0} + W_{uv} u_{t-1} + W_{qv} q_{t-1}, \tag{7.22}$$

$$b_{h_t}(n) = b_h + W_{uh}(n) u_{t-1} + W_{qh}(n) q_{t-1}, \tag{7.23}$$

where $b_{h_t}(n)$ and b_{vt} are the bias vector for the nth hidden layer and the bias of the visible layer at the tth time point of the recurrence for the DBN, respectively.

https://t.me/learningnets

7.4 Recurrent deep learning-based process monitoring

Anomaly detection in multivariate processes such as modern industrial processes and environmental processes requires a model that efficiently learns and captures relevant information, thereby summarizing the evolution of the process under nominal operating conditions. Information learned from anomaly-free data enables to quantify the degree of deviation from the desired behavior and specification in the detection stage. An anomaly in an industrial plant can propagate very quickly to reach a critical point and the plant must be shut down to avoid catastrophic problems. Accordingly, anomaly detection in time series measurements of technological processes is vital for their reliability and efficiency.

The improved ability of the RNN-based models described above to model dependencies and learn relevant features in complex multivariate sequences makes them a promising tool for anomaly detection. They can first be used to model normal behavior of the inspected process, then applied to detect deviations from normal behavior without any labeling of data. RNN-based approaches have become increasingly important to their capacity to extract complex nonlinear features, and have been employed for an early detection of anomalies in industrial multivariate time series data [40,41]. In the literature, RNN-based techniques show an effective performance in different applications including intrusion detection [41], anomalies in industrial processes [41–44], aircraft data monitoring [40]. In [40], an RNN-based anomaly detection approach has been introduced to uncover anomalies in multivariate time-series data from aircraft's flight data records, which are very helpful to identify abnormal events that may reduce safety margins. In this approach, the RNN has been trained in a supervised manner. It has been shown that RNNs-based algorithms (a simple RNN, LSTM, and GRU) outperform the multiple kernel anomaly detection (MKAD) approach based on one-class SVM, data clustering algorithms, and machine learning models such as SVM. However, this anomaly detection system needs more tuning to improve its overall performance. An anomaly detector based on an LSTM method was developed in [41] for intrusion detection on a car's controller area network (CAN) bus. The LSTM-based method has been applied in an unsupervised way to predict the next sequence of communication and classify it as malicious behavior or not. A deep LSTMs approach stacking several LSTM units is proposed in [45] for detecting anomalies (cardiac arrhythmias) in electrocardiography (ECG) signals, which are commonly utilized to check the health of the human heart. The proposed model here is built in unsupervised learning. Here, we base detection on the probability distribution of the prediction errors generated from the deep LSTM in case of normal or abnormal behavior. In [46], a multiscale LSTM (MS-LSTM) model is employed for detecting anomalies in border gateway protocol (BGP) traffic, which is a protocol usually utilized on the internet for autonomous systems to communicate routing and reachability information to improve the security of the internet. The model is trained based on the traffic patterns from historical

https://t.me/learningnets

features over a sliding time window. The results indicate that the MS-LSTM method is promising and helpful in enhancing the security and robustness of the Internet. In [43], an anomaly detection approach has been presented using a LSTM-based encoder–decoder scheme. In this approach, the proposed LSTM-based model is first constructed using normal time series that reflects the nominal behavior of the inspected process. Then, the reconstruction error for an unseen dataset is employed to detect anomalies. This approach was applied to real datasets, including electrocardiogram (ECG), power demand, and space shuttle, and showed promising anomaly detection results. An unsupervised method using a denoising autoencoder (DAE) that feed bidirectional LSTM with auditory spectral features is proposed in [47]. The autoencoder is built using typical in-home situations without abnormal events. Abnormal event detection is accomplished using the input reconstruction error between the input and the output of the constructed model. In [48], an LSTM-based approach is designed for anomaly detection and applied for intrusion detection using a KDD 1999 dataset. First, the LSTM model is constructed based on normal data and used to predict several time steps ahead of the input data. The anomaly is detected by checking the predicted error in a circular array, which comprises the prediction errors from a certain number of recent time points. An anomaly is flagged if the predicted error exceeds the predetermined decision threshold.

In this section, we present two approaches based on RNN models for anomaly detection in multivariate time series data, namely residuals-based monitoring approaches and RNN-based clustering approaches.

7.4.1 Residuals-based process monitoring approaches

Anomaly detection in multivariate time-series data from industrial and environmental processes is important to identify abnormal events and trends that reduce safety margins. Here, we briefly describe the basic idea behind the residuals-based anomaly detection methodology using recurrent models (e.g., RNN, LSTM, GRU, and RNN-RBM). As discussed above, the benefits of the RNN models are due to their simplicity and capability to appropriately capture time dependencies in multivariate time series data. The used recurrent model is trained based on anomaly-free data. Only the normal sequences collected from the supervised process under nominal conditions are used for training. As the recurrent model is used to model normal behavior, training only has to be done once offline. The constructed recurrent model is adopted for monitoring new test data. The model is used to generate residuals for anomaly detection (Fig. 7.15). The residuals, $\mathbf{E} = [e_1, e_2, \ldots, e_n]$, are the difference between the real measurements, $\mathbf{Y} = [y_1, y_2, \ldots, y_n]$, and the prediction from the RNN model $\hat{\mathbf{Y}} = [\widehat{y}_1, \widehat{y}_2, \ldots, \widehat{y}_n]$. The residuals are defined as

$$\mathbf{E} = \mathbf{Y} - \hat{\mathbf{Y}}. \tag{7.24}$$

https://t.me/learningnets

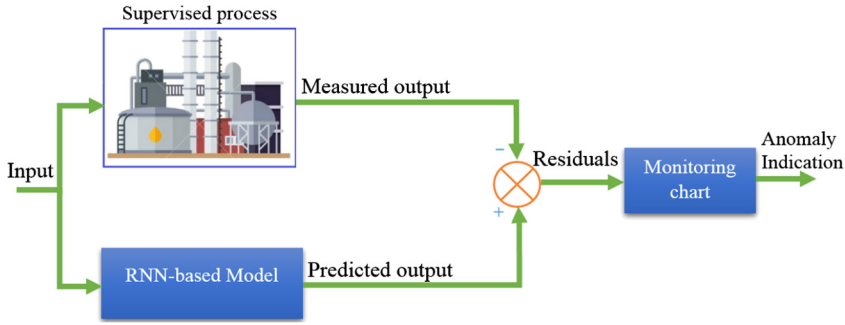


FIGURE 7.15 Residuals-based process monitoring approaches.

Under normal conditions, residual values fluctuate closer to zero, whereas in the presence of an anomaly, residuals deviate importantly from zero. Univariate or multivariate monitoring charts (e.g., EWMA, CUSUM, and GLRT) can be applied to residuals to uncover anomalies. If the charting statistic exceeds the decision threshold, then it can be inferred that there is an anomaly in the inspected process. An unsupervised anomaly detection approach is required that needs only anomaly-free data to construct the recurrent model and no data labeling.

7.4.2 Recurrent deep learning-based clustering schemes for process monitoring

7.4.2.1 RNN-RBM clustering

In Sect. 7.4.1, we explained that deep recurrent and hybrid deep models can be used for early detection of anomalies in multivariate time series data, where the detection is performed by using statistical monitoring charts. Since statistical monitoring charts generally require that the data must be Gaussian and uncorrelated, here we present an alternative and more flexible strategy. Specifically, we integrate the desirable features of RNN-based models with binary clustering algorithms. One major feature of binary clustering algorithms such as OCSVM is their ability to deal with nonlinear and non-Gaussian data. The primary objective of this approach is to exploit the deep recurrent models to accurately describe the anomaly-free time series data and use clustering algorithms to reliably detect the presence of anomalies.

For the sake of simplicity, in this section we focus on early detection with an RNN-RBM model and an OCSVM clustering algorithm. Other recurrent models can be combined in a similar way with a binary clustering algorithm for anomaly detection. We first present the outline of the proposed method and the training procedure of the RNN-RBM model. Then, we explain the basic concept of combining OCSVM together with a RNN-RBM model to efficiently detect anomalies in multivariate time-series data.

https://t.me/learningnets

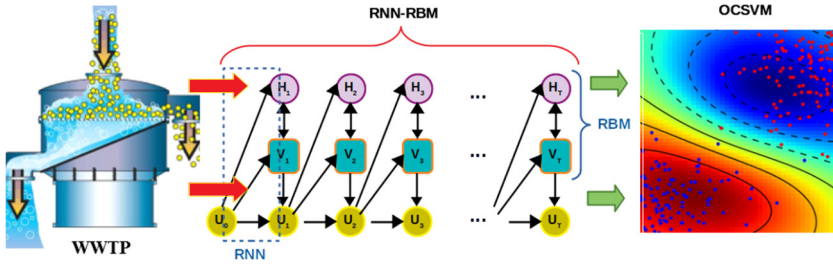


FIGURE 7.16 Illustrative graphic of the RNN-RBM-based OCSVM method.

As described above, the major characteristic of an RNN-RBM is to simultaneously combine the desired features of RNNs and RBMs. This model is based on a powerful data distribution estimator, namely RBM, conditioned by the RNN to describe local and long-term dependence in the data. The training of RNN-RBM model is performed in an unsupervised manner where only anomaly-free data are needed. After the training of the RNN-RBM model has been accomplished by using anomaly-free data, a feature space is obtained with the capability of reconstructing the input with small error via encoding and decoding tasks. The designed RNN-RBM model can be then applied to predict the evolution of the inspected system (e.g., inflow characteristics of WRRFs). To design a suitable RNN-RBM model that captures temporal dependencies in the data well, minimizing cross-entropy error [49] is used as the cost function during the training phase. The aim of RNN-RBM is to recreate the inputs as accurately as possible. Here, the cross-entropy is used to quantify the accuracy of the designed RNN-RBM during the training phase, by showing the dissimilarity between the probability distributions of input and reconstruction from the RNN-RBM model. Then the designed RNN-RBM model in combination with OCSVM algorithm is adopted to detect abnormalities in multivariate time series data (see Fig. 7.16). Here, to detect anomalies, we used the extracted features from the RNN-RBM model as input to the OCSVM. Of course, the RNN-RBM-based OCSVM method consists of three main stages: (1) building RNN-RBM model based on training dataset without anomaly, (2) training in an unsupervised way the OCSVM with the features discovered by the RNN-RBM model, and (3) applying the constructed approach to new data for anomaly detection. This approach is outlined in Table 7.2.

7.5 Applications: monitoring influent conditions at WWTP

WWTPs represent promising solutions to mitigate problems of water scarcity. The quality of influent measurements (IMs) at WWTPs can impact treatment units states, ongoing process mechanisms, and product qualities. Abnormalities in IMs, frequently generated by anomalous events, require early detection for

https://t.me/learningnets

TABLE 7.2 Main steps of the building the recurrent probabilistic features extractor of the proposed system.

Stage 1: Modeling	
Step 1	Normalize the collected time-series data from the system
Step 2	Determine model parameters based on unsupervised training of the RNN-RBM, wherein each time-step only one RBM is used
Step 3	Calculate the weights and bias for RNN-RBM model
Step 4	Develop the OCSVM model using the extracted features from the RNN-RBM model
Step 5	Find the optimum hyperplane that separates normal from abnormal features to construct the anomaly detector
Stage 2: Detecting anomalies	
Step 1	Gather and scale the new test data
Step 2	Perform mapping of the scaled data $\hat{\mathcal{X}}$ into feature space $\mathcal{F}_{\hat{\mathcal{X}}}$ via the designed RNN-RBM model
Step 3	Verify if $\mathcal{F}_{\hat{\mathcal{X}}}$ is an anomaly or normal data using the previously defined hyperplans

enhancing system resilience. The feasibility of RNN, RBM, and RNN-RBM based clustering methods is verified by seven years IMs measurements (from September 1, 2010 to September 1, 2017) from a coastal municipal WWTP.

We utilize measurements collected from September 1, 2010 to May 14, 2011 to build the studied models. The IM data is first smoothed with the exponential smoothing to reduce the effect of noise measurements and then normalized. We used this data to train the RNN, RBM, and RNN-RBM-based models and stand-alone individual algorithms. The selected values of parameters of the three models are given in Table 7.3.

TABLE 7.3 Parameters in RBM, RNN, and RNN-RBM.

Models	Parameter	Value
RNN-RBM	batch size	10
	learning rate	0.001
	loss function	cross-entropy
	number of hidden units by layer	40
	number of recurrent hidden units	20
	number of visible units by layer	20
	optimizer	Adam
	training epochs	200
RBM	Learning rate	0.001
	Training epochs	200
DBM	Layers	03
	Learning rate	0.001
	Training epochs	200

https://t.me/learningnets

In this study, we apply binary clustering algorithms for anomaly detection, in which new data in the testing set will either be normal or abnormal. Indeed, clustering algorithms used here are first trained in an unsupervised way (without involving their labels in learning) using training data, i.e., anomaly-free data collected when the WWTP is under normal operating conditions. Note that the clustering procedures used in this study are parametric, and the number of clusters has been fixed in the training phase as binary, where anomaly-free data have been clustered in a dominant cluster as they have comparable patterns. Furthermore, these unsupervised clustering methods do not require labeled data or prior knowledge of different types of anomalies to guarantee an appropriate detection performance, which makes them appropriate for real-time monitoring. Parameters of each clustering method are summarized in Table 7.4. To quantitatively assess the detection efficiency of the proposed procedures, the following metrics were employed: true positive rate (TPR, or recall), false positive rate (FPR), area under the receiver operating characteristic curve (AUC), accuracy, precision, and F1-score.

TABLE 7.4 Values of parameters employed in the investigated methods.

Models	Parameter	Value
OCSVM	gamma kernel nu	2 radial basis function 0.001
Mean-Shift	bandwidth	0.44
Agglomerative	affinity linkage	Euclidean ward
k-means	init init iteration	*k*-means++ 10 300
Spectral clustering	affinity gamma	rbf 1.0
EM	covarianceType covar iteration	full 1e–06 100

Our main objective is to verify the ability of recurrent models, such as RNN and RNN-RBM models, to detect abnormal changes in IM time series data. Once these models are built using anomaly-free data, then they are employed with binary clustering algorithms to detect anomalous observations in IM measurements. We employed the measurements gathered from May 15, 2011 to September 1, 2017 for testing the detection methods. These data include numerous anomalies including seawater intrusion, and hypochlorite dosage.

https://t.me/learningnets

The detection results obtained from RNN-RBM-based clustering approaches are summarized as radar chart in Fig. 7.17. The detection approach based on RNN-RBM-based OCSVM method can adequately detect the abnormal conditions by achieving an AUC of 0.98 (Fig. 7.17). This result indicates that RNN-RBM-based OCSVM uncovers almost all abnormal events in IM measurements declared by the operator KAUST WRRF. In addition, the results show that OCSVM outperforms the other methods (Fig. 7.17). This is mainly due to the flexible capacity of RNN-RBM to model time-series data and the ability of OCSVM to detect small deviations in the features.

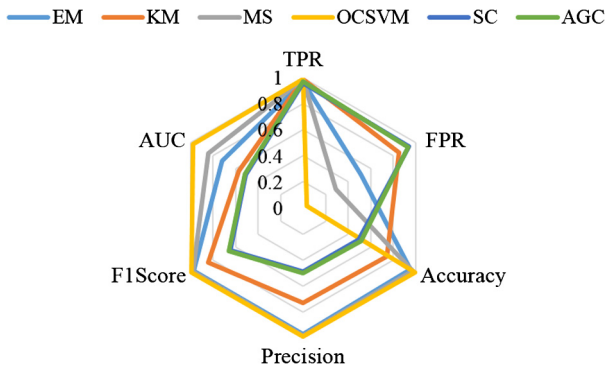


FIGURE 7.17 Detection efficiency of RNN-RBM-based clustering approaches based on testing data.

We then compared the detection results of the hybrid RNN-RBM-based clustering algorithms to other deep models, RNN, RBM, and DBM. In RNN, RBM, and DBM-based approaches, the binary clustering schemes (i.e., OCSVM, K-means, BIRCH, mean-shift, maximum expectation, and spectral clustering) are applied to the output features of the RNN, RBM, and DBM models. A summary of the performance of RNN, RBM, and DBM-based clustering schemes are respectively presented in Figs. 7.18, 7.19 and 7.20.

From Figs. 7.17, 7.18, 7.19, and 7.20, we can see that the RNN-RBM-based approaches have a better detection performance compared to RNN, RBM, and DBM-based techniques. In fact, RBM and DBM models do not capture time dependence in modeling time-series data, which misses important features, and the pattern might be due to unsuited detection performance when they are used as features extractors (RBM and DBM) for anomaly detection. RNN-RBM model is a powerful hybrid neural architecture able to model temporal correlation across large time steps in the multivariate data and thus leads to a more representative model. The results demonstrate that the RNN-RBM-based approach offers enhanced performance compared to RNN, RBM, and DBM-based approaches in monitoring ICs of KAUST WRRF. We believe this is because RNN-RBM model is appropriate for approximating a complicated distribution for each time step.

https://t.me/learningnets

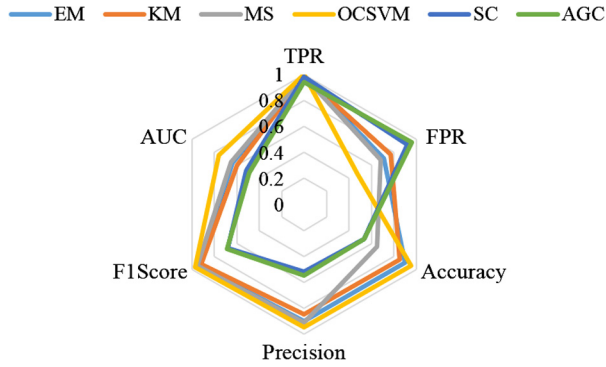


FIGURE 7.18 Detection efficiency of RNN-based clustering approaches based on testing data.

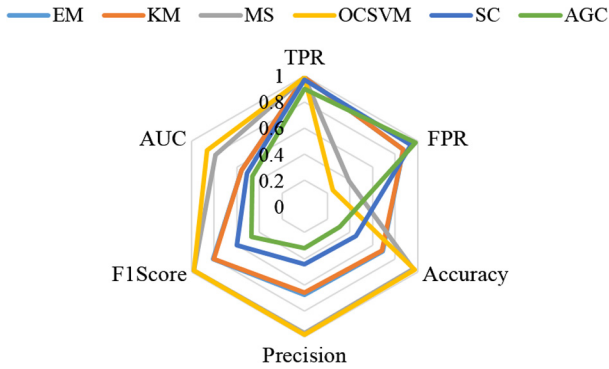


FIGURE 7.19 Detection efficiency of RBM-based clustering approaches based on testing data.

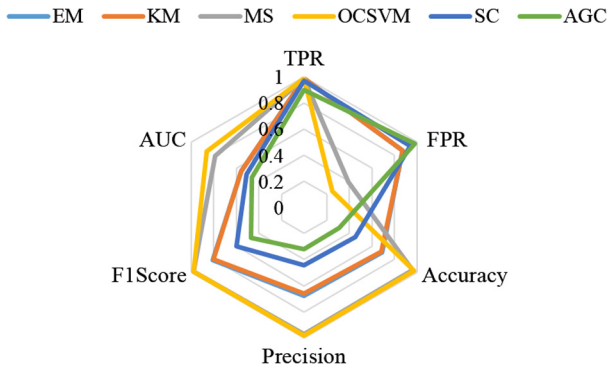


FIGURE 7.20 Detection efficiency of DBM-based clustering approaches based on testing data.

In addition, we also compared the efficiency of the proposed approach with stand-alone clustering techniques in detecting abnormalities in IC data. Fig. 7.21 presents the results obtained when applying stand-alone clustering techniques

https://t.me/learningnets

to IC data. One can see that the achieved monitoring results by the stand-alone clustering methods to monitor IC data are unsatisfactory. Experimental results on IC data show that RNN-RBM-based OCSVM outperforms the baseline competitors by a large margin (Fig. 7.21).

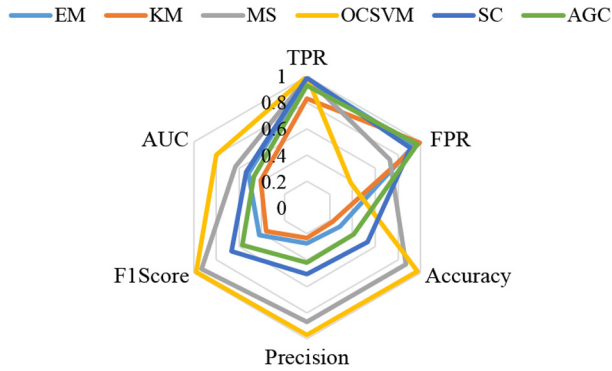


FIGURE 7.21 Detection performances of stand-alone clustering detectors.

Furthermore, to make the comparison of the results easier, Fig. 7.22 displays the AUC comparison between the studied algorithms. The results indicate that the RNN-RBM-based OCSVM is better at detecting anomalies in comparison to the other methods presented in this study.

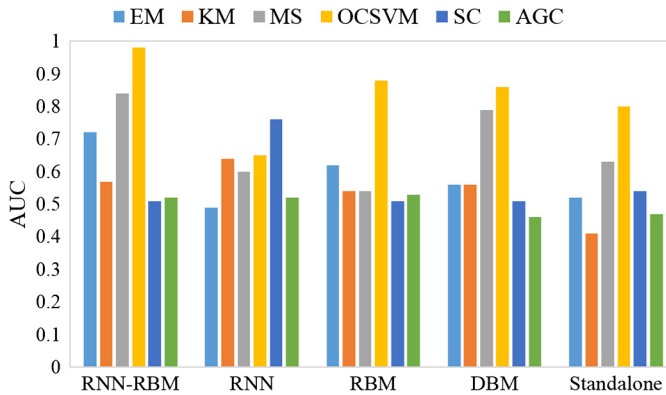


FIGURE 7.22 AUC metric of RNN-RBM, RBM, and DBM.

In summary, the results indicate that the hybrid RNN-RBM model is an intuitive choice for modeling IMs multivariate time-series data. The RNN-RBM modeling approach has been combined with binary clustering algorithms to monitor IMs data of KAUST WWTP. RNN-RBM with OCSVM can adequately distinguish normal observations form abnormal IMs observations. This study shows that the combination of RNN-RBM and OCSVM results in better detection. RNN-RBM are designed to handle multivariate time series data

https://t.me/learningnets

and OCSVM is efficient in separating normal from abnormal features. From the results above, we can summarize that the RNN-RBM-based OCSVM are more appropriate and efficient than the stand-alone clustering schemes, and the RBM and DBM-based and RNN-based clustering approaches, which justifies the superior feature extraction of the RNN-RBM model. Furthermore, OCSVM, which is trained in an unsupervised way, can accurately distinguish the normal from abnormal conditions compared to other studied binary clustering algorithms. The power of OCSVM consists is that it is a nonlinear kernel-based classifier, which defines hyperplane separating anomaly-free samples and abnormal samples by means of the projection in features space. The OCSVM demonstrated its highest performance by dominating the other competitors in all cases presented in this study.

7.6 Discussion

Anomaly detection has applications in various domains, such as air quality monitoring, intrusion detection, product quality monitoring, and system health monitoring. Recurrent deep learning models present new ways to model complex time series data due to their extended capability to capture nonlinear interdependencies, which goes beyond a traditional mindset. As discussed in this chapter, recurrent models such as RNN, LSTM, GRU are becoming well-reputed with the recent breakthrough in deep learning and they are very efficient in describing temporal dependencies in multivariate time series data. It should be noted that RNN units are subjected to the vanishing/exploding gradient issue, which degrades the model's capacity to capture long-term dependencies. To alleviate these problems, LSTM and GRU units have been designed and yield improved results as described in Sects. 7.2 and 7.3.

To uncover anomalies in time series data, recurrent models are used to extract relevant information from massive data and generate residuals which are then evaluated by existing monitoring tools (e.g., statistical monitoring schemes and clustering algorithms) that are proven effective. One of the major advantages of using these unsupervised deep recurrent models to detect anomalies is that they do not need hand-coded features and prior knowledge of anomalies to work. For the purpose of anomaly detection, residuals generated by the reference recurrent models can be checked by the statistical monitoring schemes such as univariate and multivariate CUSUM and EWMA or by using binary clustering algorithms. In this chapter, we discussed the general framework of recurrent deep learning based anomaly detection in Sect. 7.4.

In this chapter, to improve process monitoring in case of high-dimensional data, we have described some hybrid models integrating the benefits of recurrent models and energy-based models such as RBM and DBN. These recurrent hybrid models enjoy certain desirable properties when modeling high-dimensional time-series datasets. Here, three flexible hybrid models to capture temporal dependencies in a multivariate setting were discussed: RNN-RBM, LSTM-RBM,

https://t.me/learningnets

and LSTM-DBN architectures. In this chapter, deep hybrid recurrent models were coupled with binary clustering algorithms for anomaly detection in multivariate times series data.

The efficacy of RNN, RBM, and RNN-RBM based methods, or stand-alone binary clustering methods, has been verified by using seven years ICs data from an actual WWTP where more than 150 anomalies happened. Results show that RNN-RBM-based OCSVM outperformed the state-of-the-art competitors by a large margin. As a perspective, other powerful architectures that help learn temporal dependencies in data can be explored by merging recurrent models and energy-based models and autoencoders.

References

[1] H. Lütkepohl, New Introduction to Multiple Time Series Analysis, Springer Science & Business Media, 2005.
[2] J.D. Hamilton, Time series analysis, in: Economic Theory. II, Princeton University Press, USA, 1995, pp. 625–630.
[3] I. Melnyk, A. Banerjee, Estimating structured vector autoregressive models, in: International Conference on Machine Learning, 2016, pp. 830–839.
[4] J. Li, W. Chen, Forecasting macroeconomic time series: lasso-based approaches and their forecast combinations with dynamic factor models, International Journal of Forecasting 30 (4) (2014) 996–1015.
[5] R. Frigola, F. Lindsten, T.B. Schön, C.E. Rasmussen, Bayesian inference and learning in Gaussian process state-space models with particle MCMC, in: Advances in Neural Information Processing Systems, 2013, pp. 3156–3164.
[6] Z. Chen, Y. Chen, L. Wu, S. Cheng, P. Lin, L. You, Accurate modeling of photovoltaic modules using a 1-D deep residual network based on IV characteristics, Energy Conversion and Management 186 (2019) 168–187.
[7] F. Harrou, A. Dairi, Y. Sun, F. Kadri, Detecting abnormal ozone measurements with a deep learning-based strategy, IEEE Sensors Journal 18 (17) (2018) 7222–7232.
[8] F. Harrou, A. Dairi, Y. Sun, M. Senouci, Statistical monitoring of a wastewater treatment plant: a case study, Journal of Environmental Management 223 (2018) 807–814.
[9] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using RNN encoder–decoder for statistical machine translation, preprint, arXiv:1406.1078, 2014.
[10] J. Chung, C. Gulcehre, K. Cho, Y. Bengio, Empirical evaluation of gated recurrent neural networks on sequence modeling, preprint, arXiv:1412.3555, 2014.
[11] G.-B. Zhou, J. Wu, C.-L. Zhang, Z.-H. Zhou, Minimal gated unit for recurrent neural networks, International Journal of Automation and Computing 13 (3) (2016) 226–234.
[12] T. Mikolov, A. Joulin, S. Chopra, M. Mathieu, M. Ranzato, Learning longer memory in recurrent neural networks, preprint, arXiv:1412.7753, 2014.
[13] Q.V. Le, N. Jaitly, G.E. Hinton, A simple way to initialize recurrent networks of rectified linear units, preprint, arXiv:1504.00941, 2015.
[14] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436.
[15] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Computation 9 (8) (1997) 1735–1780.
[16] A. Graves, M. Liwicki, S. Fernández, R. Bertolami, H. Bunke, J. Schmidhuber, A novel connectionist system for unconstrained handwriting recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence 31 (5) (2009) 855–868.
[17] W. Zaremba, I. Sutskever, O. Vinyals, Recurrent neural network regularization, preprint, arXiv: 1409.2329, 2014.

https://t.me/learningnets

[18] X. Huang, H. Tan, G. Lin, Y. Tian, A LSTM-based bidirectional translation model for optimizing rare words and terminologies, in: 2018 International Conference on Artificial Intelligence and Big Data (ICAIBD), May 2018, pp. 185–189.
[19] S.P. Singh, A. Kumar, H. Darbari, L. Singh, A. Rastogi, S. Jain, Machine translation using deep learning: an overview, in: 2017 International Conference on Computer, Communications and Electronics (Comptelix), July 2017, pp. 162–167.
[20] B. Athiwaratkun, J.W. Stokes, Malware classification with LSTM and GRU language models and a character-level CNN, in: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), March 2017, pp. 2482–2486.
[21] E. Song, F.K. Soong, H. Kang, Effective spectral and excitation modeling techniques for LSTM-RNN-based speech synthesis systems, IEEE/ACM Transactions on Audio, Speech and Language Processing 25 (11) (Nov 2017) 2152–2161.
[22] M. Bi, H. Lu, S. Zhang, M. Lei, Z. Yan, Deep feed-forward sequential memory networks for speech synthesis, in: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), April 2018, pp. 4794–4798.
[23] M. Rozenwald, E. Khrameeva, G. Sapunov, M. Gelfand, Prediction of 3D chromatin structure using recurrent neural networks, in: 2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), 2018, p. 2488.
[24] L.T. Hattori, C.M.V. Benitez, M. Gutoski, N.M.R. Aquino, H.S. Lopes, A novel approach to protein folding prediction based on long short-term memory networks: a preliminary investigation and analysis, in: 2018 International Joint Conference on Neural Networks (IJCNN), 2018, pp. 1–8.
[25] X. Pan, P. Rijnbeek, J. Yan, H.-B. Shen, Prediction of RNA-protein sequence and structure binding preferences using deep convolutional and recurrent neural networks, BMC Genomics 19 (1) (2018) 511.
[26] Y. Jiang, Z. Wu, J. Tang, Z. Li, X. Xue, S. Chang, Modeling multimodal clues in a hybrid deep learning framework for video classification, IEEE Transactions on Multimedia 20 (11) (Nov 2018) 3137–3147.
[27] Y. Yang, J. Zhou, J. Ai, Y. Bin, A. Hanjalic, H.T. Shen, Y. Ji, Video captioning by adversarial LSTM, IEEE Transactions on Image Processing 27 (11) (2018) 5600–5611.
[28] L. Gao, Z. Guo, H. Zhang, X. Xu, H.T. Shen, Video captioning with attention-based LSTM and semantic consistency, IEEE Transactions on Multimedia 19 (9) (Sep. 2017) 2045–2055.
[29] A. Ullah, J. Ahmad, K. Muhammad, M. Sajjad, S.W. Baik, Action recognition in video sequences using deep bi-directional LSTM with CNN features, IEEE Access 6 (2018) 1155–1166.
[30] W. Du, Y. Wang, Y. Qiao, Recurrent spatial-temporal attention network for action recognition in videos, IEEE Transactions on Image Processing 27 (3) (March 2018) 1347–1360.
[31] F.A. Gers, N.N. Schraudolph, J. Schmidhuber, Learning precise timing with LSTM recurrent networks, Journal of Machine Learning Research 3 (2002) 115–143.
[32] F.A. Gers, J. Schmidhuber, F. Cummins, Learning to forget: continual prediction with LSTM, Neural Computation 12 (10) (2000) 2451–2471.
[33] I. Sutskever, G.E. Hinton, G.W. Taylor, The recurrent temporal restricted Boltzmann machine, in: Advances in Neural Information Processing Systems, 2009, pp. 1601–1608.
[34] Q. Lyu, Z. Wu, J. Zhu, Polyphonic music modelling with LSTM-RTRBM, in: Proceedings of the 23rd ACM International Conference on Multimedia, ACM, 2015, pp. 991–994.
[35] K. Goel, R. Vohra, J. Sahoo, Polyphonic music generation by modeling temporal dependencies using an RNN-DBN, in: International Conference on Artificial Neural Networks, Springer, 2014, pp. 217–224.
[36] R. Vohra, K. Goel, J. Sahoo, Modeling temporal dependencies in data using a DBN-LSTM, in: 2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA), IEEE, 2015, pp. 1–4.
[37] N. Boulanger-Lewandowski, Y. Bengio, P. Vincent, Modeling temporal dependencies in high-dimensional sequences: application to polyphonic music generation and transcription, preprint, arXiv:1206.6392, 2012.

https://t.me/learningnets

[38] A. Bäärnhielm, Multiple time-series forecasting on mobile network data using an RNN-RBM model, 2017.
[39] G.E. Hinton, Training products of experts by minimizing contrastive divergence, Neural Computation 14 (8) (2002) 1771–1800.
[40] A. Nanduri, L. Sherry, Anomaly detection in aircraft data using recurrent neural networks (RNN), in: 2016 Integrated Communications Navigation and Surveillance (ICNS), IEEE, 2016, pp. 5C2-1–5C2-8.
[41] A. Taylor, S. Leblanc, N. Japkowicz, Anomaly detection in automobile control network data with long short-term memory networks, in: 2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA), IEEE, 2016, pp. 130–139.
[42] Z. Li, J. Li, Y. Wang, K. Wang, A deep learning approach for anomaly detection based on SAE and LSTM in mechanical equipment, The International Journal of Advanced Manufacturing Technology (2019) 1–12.
[43] P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, G. Shroff, LSTM-based encoder–decoder for multi-sensor anomaly detection, preprint, arXiv:1607.00148, 2016.
[44] P. Filonov, F. Kitashov, A. Lavrentyev, RNN-based early cyber-attack detection for the Tennessee Eastman process, preprint, arXiv:1709.02232, 2017.
[45] S. Chauhan, L. Vig, Anomaly detection in ECG time signals via deep long short-term memory networks, in: 2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA), IEEE, 2015, pp. 1–7.
[46] M. Cheng, Q. Xu, J. Lv, W. Liu, Q. Li, J. Wang, MS-LSTM: a multi-scale LSTM model for BGP anomaly detection, in: 2016 IEEE 24th International Conference on Network Protocols (ICNP), IEEE, 2016, pp. 1–6.
[47] E. Marchi, F. Vesperini, F. Eyben, S. Squartini, B. Schuller, A novel approach for automatic acoustic novelty detection using a denoising autoencoder with bidirectional LSTM neural networks, in: 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2015, pp. 1996–2000.
[48] L. Bontemps, J. McDermott, N.-A. Le-Khac, et al., Collective anomaly detection based on long short-term memory recurrent neural networks, in: International Conference on Future Data and Security Engineering, Springer, 2016, pp. 141–152.
[49] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.-A. Manzagol, Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion, Journal of Machine Learning Research 11 (2010) 3371–3408.

https://t.me/learningnets

Chapter 8

Case studies

8.1 Introduction

Efficient management of road traffic is becoming necessary to reduce the economic losses caused by traffic congestion and accidents. A survey carried out by the Organization for Economic Cooperation and Development (OECD) revealed that 10 million people are involved in a road accidents each year globally. Of these, 20% to 30% are severely injured and 400,000 die. Hence, researchers and practitioners are working to design intelligent transportation systems and autonomous vehicles. Numerous technologies have been developed to effectively detect obstacles in road environments using advanced sensors, including RADAR and LIDAR systems and 3D and 360-degree cameras [1,2]. Reliable detection and localization of obstacles are central problems that need to be addressed in order to avoid collision during autonomous driving. Studies have examined several practical applications involving obstacle detection, including swarm robotics, unmanned aerial vehicles, agricultural applications, and smart wheelchairs [2,3]. The aim of this chapter is to detect obstacles faced by autonomous robots in indoor environments using stereovision.

Recently, based on the emerging concept of smart cities, considerable work has been carried out to improve traffic management and intelligent transportation systems, such as vehicle-to-vehicle or vehicle-to-infrastructure (e.g., traffic signs) systems, and to build intelligent roads equipped with sensors. This is intended to considerably reduce the risk of accidents and financial burden and improve human safety [4–10]. Obstacle detection and localization are the main challenges that must be overcome in designs for driving assistance systems (DASs). Over the last two decades, researchers and practitioners have made efforts to develop real-time obstacle detection techniques for autonomous vehicles. Currently, most automated decision-making systems that have been deployed are based on advanced sensors, such as RADAR and LIDAR systems and 3D cameras [8,1]. In [11], an obstacle detection technique was designed for an outdoor environment using Microsoft Kinect. In [12], an intelligent fiber grating (FG)-based 3D vision sensory system was proposed for real-time obstacle detection and tracking using a charge-coupled device (CCD) CCD camera equipped with laser technology. In [13], the authors proposed two obstacle detection approaches; one used a monocular camera and the other used 360-degree vertical cameras. Much work has been done to develop improved obstacle de-

https://doi.org/10.1016/B978-0-12-819366-5.00014-0
Copyright © 2021 Elsevier Inc. All rights reserved.
https://t.me/learningnets

tection techniques using Stixel World [14,15] by looking for free spaces. In this program, stixels represent 3D scenes in a compact manner, bridging the gap between objects and pixels [14]. In other words, they offer an efficient way to encode free space and obstacles in front of the vehicle in which 3D objects are represented by a set of rectangular sticks called stixels. Stixel-based approaches are based on image processing for detection and recognition by identifying regions of interest (ROIs), which serve as the input for training a classifier [14, 15]. Semiglobal matching is implemented on the field-programmable gate array (FPGA) board and a graphics processing unit (GPU) is used to compute dense optical flow for tracking purposes.

Nadav and Katz introduced an approach for obstacle detection in an off-road environment based on a monocular camera [16]. This approach used both 2D and 3D video analysis methods. Woo and Kim also proposed a vision-based approach to detect obstacles and estimate the collision risk of an unmanned surface vehicle [17]. It was implemented by transforming visual information regarding obstacles into motion information [17]. Furthermore, Burlacu et al. designed a stereovision technique for obstacle detection based on multiple representations of the disparity map (i.e., V-disparity, U-disparity, θ-disparity) [18]. This technique identifies obstacles in a 3D environment with disparity processing [18]. Other approaches use descriptors extracted from images, including local binary patterns, scale-invariant feature transforms, and histograms of oriented gradients, for obstacle detection [19]. Unfortunately, these approaches require full processing of images to detect obstacles and involve a high level of computation.

Guaranteeing road safety remains challenging because of the complexity of this topic. To date, more focus has been placed on vision sensing for obstacle detection, recognition, localization, and tracking systems. Computer vision provides an efficient way to observe the surrounding environment using different techniques, such as stereovision, which allows a camera to perceive the world in 3D by estimating depth. Currently, of the computer-vision techniques, stereovision has attracted the most attention among researchers because of its simplicity, robustness, low cost of computing, and potential for 3D perception. Stereovision-based obstacle detection has a central role in the design of autonomous and intelligent transportation systems to reduce the risk of accidents [20,21,4,7,22].

Recently, detecting obstacles via machine learning approaches has become a hot research area [23–27]. There are two classes of machine learning approaches: shallow and deep learning [28,29]. Numerous shallow algorithms have been applied for supervised obstacle detection and classification, including support vector machines (SVMs) and neural networks with one or two layers [29]. For instance, an approach using a histogram of oriented gradients (HOG) and SVM was applied for human detection using a single view. Shallow learning methods are appealing for their simplicity and ease of implementation. However, they are unsuitable for uncovering relevant features when handling complex and high-dimensionality data [30,31]. On the other hand, the grow-

https://t.me/learningnets

ing complexity of the gathered data has resulted in the development of deep learning methods that can account for features such as time-dependent measurements, seasonality, and non-Gaussianity. In comparison to shallow methods, deeper networks are more accurate and are able to learn more comprehensive information and relevant features of obstacles from the training data set. Deep convolutional neural networks (CNNs), which have demonstrated a high capacity for image classification, have been widely used to detect and recognize obstacles. In [32], deep CNNs were applied to detect and recognize obstacles using 2D images. Also, in [27], deep CNNs were used to detect unexpected obstacles for self-driving cars. Although they were effective for identifying obstacles using 2D images, CNN-based methods were not able to approximate the distribution of data, reduce dimensionality, or learning in an unsupervised manner, which are indispensable for real-time obstacle detection [30]. To alleviate these shortcomings, researchers have developed unsupervised deep learning methods, such as restricted Boltzmann machines and autoencoders [33]. Usually, in these unsupervised approaches, an image is fully scanned and then the ROI is surrounded. However, this process can be performed for both free scenes and scenes that contain obstacles, which makes it computationally costly. In other methods, the obstacle detection problem is viewed as an anomaly detection problem. First, a reference model is constructed based on data that is devoid of obstacles, and then this model is used to detect obstacles in the remainder of the data.

Congestion phenomena, road traffic monitoring, and management have been the subject of much research conducted by researchers and engineers during the last few decades due to their detrimental impacts in many areas, such as human health, the environment, and economics. Additionally, effective detection of obstacles is essential for developing reliable autonomous vehicles. Since the road is shared by several traffic participants (e.g., cars, bikes, and pedestrians), especially in urban environments, full awareness of obstacles and traffic participants is indispensable for avoiding accidents that might lead to catastrophic scenarios. However, detection in urban environments is challenging due to the presence of different types of obstacles (i.e., dynamic and static). Additionally, the similarity of the obstacles to the background and the presence of cast shadows or reflections can make it difficult to detect obstacles. Stereovision is vital for understanding scenes, as it provides relevant information about obstacles and traffic participants (e.g., size and shape), which helps with early detection. Also, deep learning techniques can offer improved detection performance compared to shallow methods, which are mainly based on hand-crafted features and thus are difficult and time-consuming to design and limited in their representation abilities.

This chapter presents an unsupervised deep learning-based obstacle detection approach based on stereovision. This approach amalgamates the desirable characteristics of the deep-stacked autoencoders (DSA) model and the detection capacity of the k-nearest neighbor (kNN) clustering procedure. Specifically, the

https://t.me/learningnets

V-disparity data distribution is computed from original images and then is used as an input for the DSA model, which is a powerful way to model complex data. V-disparity is used due to its sensitivity to the presence of obstacles [34]. Then, the kNN algorithm is applied to features extracted from the DSA model for obstacle detection. This coupled approach is implemented in an entirely unsupervised way. Three available datasets (i.e., the Malaga and Daimler urban segmentation datasets and the Bahnhof data set) are used to evaluate the proposed approach. The results demonstrate that this approach has a suitable capability to detect obstacles.

The main contribution of this research is an effective unsupervised stereovision-based obstacle detection approach for autonomous vehicles in driving environments. In Sect. 8.2, stereovision and its use in obstacle detection are briefly introduced, and in Sect. 8.2.1 the DSA-kNN approach is outlined. In Sect. 8.2.2, the applied data set is presented. Section 8.2.3 presents the results and some concluding remarks.

8.2 Stereovision

In recent years, there have been great improvements in image processing tools, computer vision, and vision sensors. For instance, multiple views can be used to obtain 3D information to estimate the depth and thus determine the distance of an object from cameras mounted on moving vehicles. Using two views for 3D perception, which is commonly called stereoscopic vision, or simply stereovision, has proven especially useful in applications such as autonomous mobile robots. Stereovision requires left and right rectified images, which are processed by applying the epipolar geometry constraint to align the y-axis of the two images [4,6]. Thus, stereovision can be defined as a procedure that computes depth information by comparing two rectified images from the same scene. The extracted information is usually called a disparity map, which encodes deviation in the horizontal coordinates of corresponding image pairs. In other words, disparity maps point out disparities that represent the difference in an object's position in two corresponding rectified images.

The value of the disparity expresses the distance of an object from the camera. Small values indicate little distance between the object and camera, while larger values indicate a greater distance. The disparity map is a matrix of the distances between two corresponding pixels in the image pair (left and right). It is computed for every image pair to construct 3D data. Numerous techniques have been developed in the literature for computing disparity maps, $\mathcal{D}$, based on matching correlation metrics, including absolute differences, squared differences, the sum of absolute differences (SAD), and normalized cross-correlation [4,6,35,36]. The SAD (defined in Eq. (8.1)) is one of the most frequently used metrics to compute disparity maps due to the simplicity of implementing it in real time [37,38]. The SAD measures the absolute difference between the intensity of each pixel in the left image, I_{left}, and the corresponding

https://t.me/learningnets

pixel in the right image, I_{right}:

$$\mathcal{D}_{\text{SAD}}(i, j, d) = \sum_{u=-\omega}^{\omega} \sum_{u=-\omega}^{\omega} \left| I_{\text{left}}(i+u, j+v) - I_{\text{right}}(i+u, j-d+v) \right| \quad (8.1)$$

where d represents the disparity interval $[d_{\min}, d_{\max}]$, which is delimited between the minimum and maximum disparity values ($d_{\min}$ and $d_{\max}$, respectively), and i, j are the coordinates (rows and columns, respectively) of the center pixel of the SAD. Differences are summed over the support window, ω, to obtain a disparity map.

Estimating the position of the road profile is helpful for identifying obstacles that are standing in the monitored scene. After generating the disparity map, it is essential to compute the U-disparity and V-disparity maps [39]. V-Disparity maps that effectively estimate a road's profile using transform and depth estimation have been developed [4,6]. These maps have shown a good ability to provide information about obstacles' height and position in reference to the ground [4,6]. For illustration purposes, Fig. 8.1A–B depicts V-disparity maps of a free scene and a busy scene. The steps for computing the V-disparity are sketched in Algorithm 1.

Algorithm 1: V-disparity computation steps.

Input: Disparity map DispMap(rows, cols)
Input: $D_{\max}$: Max disparity value.
Output: V-disparity $DispMap_v$ (rows, $D_{\max}$)

```
for Each row r in DispMap do
    for Each column c in DispMap do
        currentDisparity ← DispMap(r, c)
        if currentDisparity > 0 then
            DispMap_v(r, c) ← (currentDisparity + 1)
```

In order to surround obstacles in the ROI, the obstacles' width must be determined. U-disparity expresses histograms over the disparity values for each image column (the U coordinate) [4,6,40]. In other words, both the V-disparity and U-disparity can be defined as the number of pixels with the same level of disparity at rows' Y-axis and columns' X-axis in the disparity map. Algorithm 2 describes the main steps taken to obtain the U-disparity.

Relevant information about urban environments can be learned from the U–V-disparity. Fig. 8.1A–B illustrates the U–V-disparity by providing examples of V-disparity and U-disparity analysis of a free scene and a scene containing obstacles, respectively. Interestingly, the road surface is viewed as an inclined line in the V-disparity map, while the static environment is represented by a vertical line (cloud of points) on the lower disparity map. Its thickness is related to the

https://t.me/learningnets

Algorithm 2: U-disparity computation steps.

Input: Disparity map DispMap(rows, cols)
Input: $D_{\max}$: Max Disparity value.
Output: U-disparity $DispMap_u$ ($D_{\max}$, cols)

```
for Each row r in DispMap do
    for Each column c in DispMap do
        currentDisparity ← DispMap(r, c)
        if currentDisparity > 0 then
            DispMap_u(r, c) ← (currentDisparity + 1)
```

richness of textures, such as those of buildings and trees (Fig. 8.1A). Based on the V-disparity map (Fig. 8.1B), we can see that obstacles on a road are represented by vertical lines with high intensities. The V-disparity map in Fig. 8.1B highlights obstacles (walking pedestrians) via vertical lines on the road profile. If the vertical line is near the right part of the V-disparity map, then the obstacle is positioned close to the front of the vehicle, and vice versa. Furthermore, the length of the vertical line indicates the height of the obstacle in front of the monitored vehicle, and the thickness of the vertical line is proportional to the obstacle's thickness in the image. In Fig. 8.1B, the U-disparity map is displayed at the top of the image, clearly showing the presence of obstacles (i.e., pedestrians). A major appeal of the U-disparity map is its capability to provide estimations of obstacles' width and depth [40–42]. By using both U-disparity and V-disparity maps, it becomes possible to discriminate between obstacles and identify ROIs surrounding obstacles, as shown in Fig. 8.1B.

Sometimes, a density map is used in stereovision instead of a V-disparity map for obstacle detection and localization. Basically, a density map is a compressed version of a V-disparity map without the loss of relevant information. A major benefit of this compression is a reduction in the computational cost. After segmenting the V-disparity in numerous cells(Fig. 8.2), the density corresponding to each cell is expressed as follows:

$$\text{Density}_{\text{Cell}} = \Big(\sum^{\text{Cell}} I(i, j) \Big) / (w \cdot h),$$

where $I(i, j)$ represents the intensity of the pixel in row i and column j, and w and h denote the width and height of the cell, respectively.

As shown above, the V-disparity map has good characteristics, and it can be used to detect obstacles in front of the vehicle at any time point by constructing a reference model of obstacle-free scenes. Below, we test different approaches involving deep-learning-based models.

https://t.me/learningnets

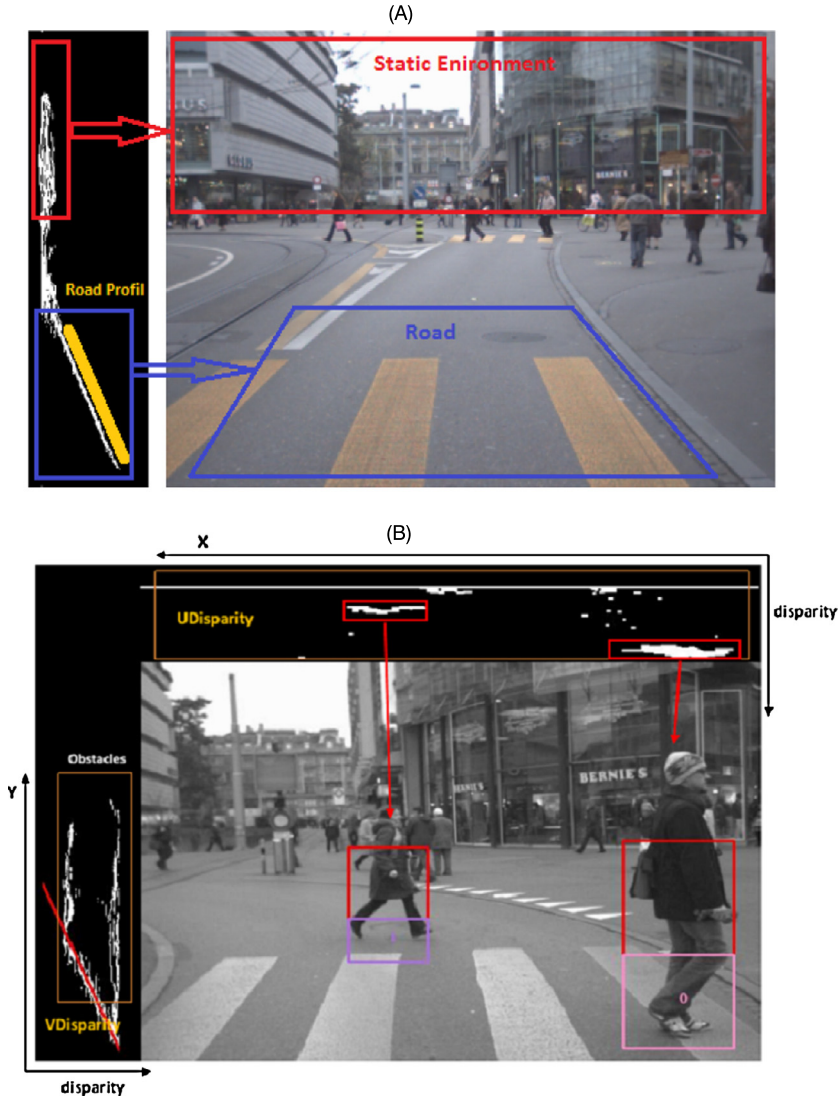


FIGURE 8.1 (A) Example of V-disparity analysis of a free scene. (B) Example of a U–V-disparity map of an urban environment with obstacles.

8.2.1 Deep stacked autoencoder-based KNN approach

This section provides a brief overview of autoencoders, which are used to build deep learning models for obstacle detection. Then, we briefly present a deep-stacked autoencoders (DSA)-based obstacle detection approach.

https://t.me/learningnets

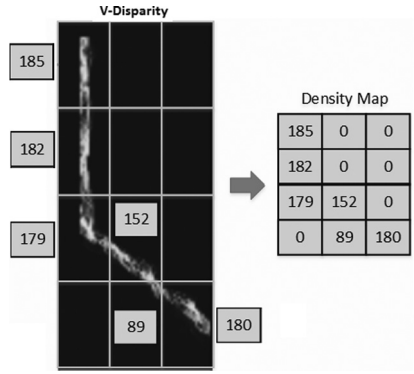


FIGURE 8.2 An illustrative example of the basic principle of density map.

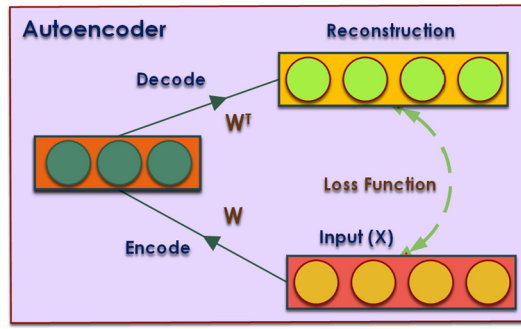


FIGURE 8.3 Diagrammatic illustration of an autoencoder.

8.2.1.1 *Preliminary materials: autoencoders*

An autoencoder is an extended version of a neural network constituted of a visible (input) layer, v, and a hidden layer, h [33]. Autoencoders are extensively used in multivariate processes to resolve problems of dimensionality reduction and feature extraction (see Fig. 8.3). Generally speaking, an autoencoder is trained without labeled data (unsupervised), and the main goal is to reconstruct inputs via two function encoders, $\mathcal{E}$ and $\mathcal{D}$. The encoder function is expressed as $h = \mathcal{E}(v)$, which can be either linear or nonlinear. When $\mathcal{E}$ is nonlinear, the autoencoder will discover and learn more complex data representation than linear PCA [33]. Essentially, the decoder function is denoted by $\widehat{v} = \mathcal{D}(h)$. Its main objective is to reconstruct inputs based on learned features. The aim of the learning process is to minimize the negative log-likelihood of the reconstruction,

$$\text{Reconstruction error} = -\log(P(v|\mathcal{E}(v)), \tag{8.2}$$

where P denotes the probability attributed to the input vector v by the model.

https://t.me/learningnets

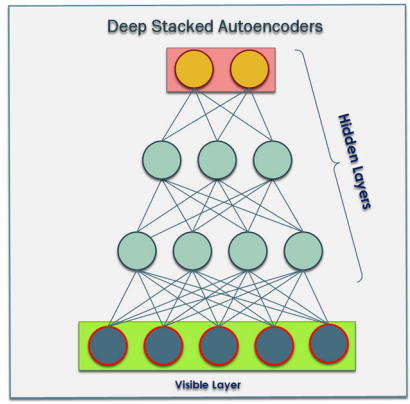


FIGURE 8.4 Diagrammatic illustration of a stacked autoencoder.

A stacked autoencoder is formed of several layers (see Fig. 8.4), each of which is an autoencoder. The outputs of each layer feed the next layer in a greedy layerwise way. The encoding process is implemented by encoding each layer in a forwarding order, and the decoding process is performed in reverse order. Stacked autoencoders have been applied for different purposes, including content-based image retrieval, image denoising [43,44], and medical object recognition [45].

8.2.1.2 The SDA-kNN obstacle detection approach

Several advanced sensors have been designed for concise and effective obstacle detection in autonomous vehicles and robots, including RADAR and LIDAR systems and 3D cameras [46,1]. However, these sensors have a high cost and require continuous maintenance. In addition, it is sometimes challenging to synchronize data from different sources. To overcome these drawbacks, vision-based methods have been designed to automatically detect and localize obstacles [42,41,47]. Such methods primarily use multiple views based on visual sensors that are able to obtain depth information and see 3D components in a scene. Binocular stereovision, which uses two rectified images (left and right) to create a disparity map, has attracted much attention among researchers. This method mimics human eyes by observing one scene from two different viewpoints [42,41,40].

An effective solution to all the above-mentioned challenges is to merge the benefits of deep learning models with nonparametric clustering methods to detect obstacles with stereovision. In this section, we present an approach that merges DSA modeling with the kNN algorithm. Essentially, the DSA model is used to model obstacle-free scenes, and the kNN algorithm is applied to the features extracted using DSA in order to reveal obstacles. This approach is advantageous because DSA is an assumption-free model, it captures relevant

https://t.me/learningnets

features from a V-disparity map, and it has greedy learning properties and the capacity for dimensionality reduction. In addition, the kNN is an unsupervised and nonparametric clustering algorithm that does not require linear or Gaussian distribution of input data.

The DSA-based kNN method detects obstacles as anomalies using a V-disparity map. The V-disparity of images perceived in urban environments without obstacles is relatively stable with weak fluctuations due to measurement noise [34,40], and significant variations in V-disparity can be observed when obstacles are present. The method includes two steps: first, the V-disparity dataset is used to provide unsupervised greedy layerwise training to the deep encoder, and second, kNN is used to reveal potential obstacles. The kNN scheme is trained using features extracted with the DSA model based on an obstacle-free scene. This allows the kNN scheme to separate free and busy scenes in the testing data.

The objective of the monitoring system is to promptly detect any obstacles in front of an autonomous vehicle or mobile robot that can lead to dangerous accidents. In summary, in the DSA-kNN approach, obstacle detection is performed as anomaly detection based on features extracted from the DSA model. The kNN is used to identify deviations between obstacle-free features and newly extracted features. The DSA-kNN obstacle detection scheme is implemented in two stages: feature generation and feature evaluation:

Stage 1. The objective of the first stage is to learn relevant information from the V-disparity map to generate features that are sensitive to the presence of obstacles. To this end, a normalized V-disparity map is used as an input for the DSA model. Then, a reference SDA model is constructed using unsupervised greedy layer-wise training with unlabeled data. This model has four layers. The role of each is to learn complex and hierarchical features, and the output of the ith layer feeds the next $(i+1)$ layer. Additionally, each layer discovers useful features and reduces dimensionality through encoding. Then, the extracted features are utilized for monitoring. The main advantage of DSA is its capacity to learn pertinent information from complexly distributed data and produce low-dimensionality outputs. One of the key purposes in training the DSA model is to minimize reconstruction errors computed through cross-entropy (represented by $\mathcal{L}$ in Eq. (8.3)). Cross-entropy is a commonly used metric for evaluating deep learning models. Optimization is achieved by minimizing cross-entropy by quantifying the mismatch between two probability distributions of the input data and the estimation of the DSA model. The DSA model is selected when the dissimilarity between the two distributions decreases, which means that cross-entropy converges near zero [44]:

$$\mathcal{L}(X, \hat{X}) = -\sum_{i}^{n}\sum_{j}^{m}(\hat{x}_{ij}\log(x_{ij}) + (1-\hat{x}_{ij})\log(1-x_{ij})) \quad (8.3)$$

https://t.me/learningnets

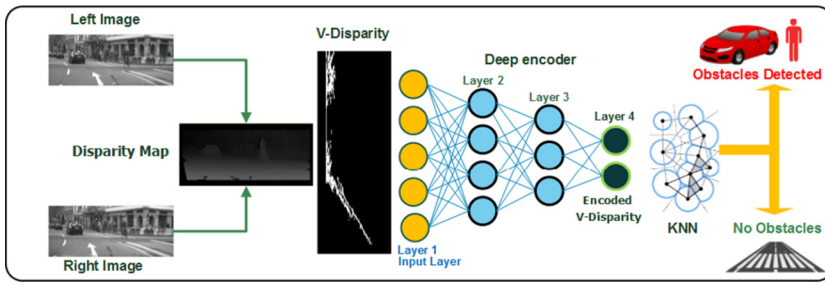


FIGURE 8.5 Illustration of the DSA-based kNN obstacle detection method.

where X is the input $(n \times m)$ normalized V-disparity matrix and $\hat{X}$ is the reconstructed V-disparity matrix via the built model of size $(n \times m)$.

Stage 2. The obstacle detection phase involves evaluating the features extracted through DSA using the kNN scheme, which is an efficient nonparametric algorithm for separating different features (Fig. 8.5). Briefly, the kNN algorithm separates normal features from abnormal features (to identify inliers from outliers) by measuring the distance between the actual observation and the kNNs of obstacle-free data. The kNN scheme are applied by computing all distances, d_j, in elements of the training set, $\mathcal{S}$, using the following equation, $d_j = \text{distance}(x, \mathcal{S}j)$. The kNN distances with large values are used as indicators of obstacles in front of the vehicle. Euclidean distance is often used to measure similarity in kNN-based approaches. In scenes with no obstacles in front of the vehicle, the kNN distance fluctuates near zero, and in scenes with obstacles, the kNN distance significantly diverges from zero. Hence, large kNN distances indicate that there is a significant inconsistency between new observations and the training (obstacle-free) observations. Specifically, for every observation x_i (output of the DSA) in the training data, the kNN distances to its nearest neighbor in the training data, D_i, are determined, based on which the sample distributions of distances can be computed:

$$D_i = \sum_{j=1}^{k} d_{ij}, \tag{8.4}$$

where d_{ij} is the distance from an observation to its jth nearest neighbor. The goal is to find obstacles at any time point within an observed scene using image pairs (left and right). This can be done by creating a detection threshold or control limit that distinguishes between free and busy scenes. We utilize the 3-sigma rule to fix the detection threshold, UCL, based on the kNN distance. In other words, we aim to

https://t.me/learningnets

flag an obstacle in front of the vehicle once the kNN distance passes the detection threshold. For the selected threshold, the false alarm rate was 0.27%. Based on the distribution of D_i, the parametric threshold of the kNN-Shewhart approach is computed as follows:

$$\text{UCL} = \mu_D + 3\sigma_D, \tag{8.5}$$

where μ_D and σ_D are the mean and standard deviation of kNN distances based on obstacle-free training data. An outlier (i.e., obstacle) is identified at the ith time point if the kNN distance, d_j, exceeds the decision threshold,

$$d_j > \text{UCL}. \tag{8.6}$$

8.2.2 Data description

The effectiveness of the previously described algorithms was verified using the Malaga stereovision urban data set (MSVUD) [48] and the Daimler urban segmentation data set (DUSD) [49]. MSVUD consists of 15 subdatasets (extracts) with rich urban scenarios. MSVUD data was collected as a vehicle traveled for more than 20 km in an urban environment, which included a straight path, turns, road junctions, avenue traffic, and a freeway. The resolution is 800×600 pixels. DUSD data, which were gathered in urban traffic, include rectified stereo image pairs with a resolution of 1024×440 pixels [49].

The training dataset consists of two subdatasets of MSVUD. The first subset (called Extract #05 in MSVUD) includes 5,000 pairs of images collected from an avenue loop closure of around 1.7 km. The second subset (called Extract #08 in MSVUD) includes 10,000 pairs of images collected from a long loop closure of around 4.5 km. We also used 500 pairs of images from the DUSD dataset. For model training purposes, we selected datasets known to be devoid of obstacles.

For testing purposes and to verify the performance of the obstacle detection methods, we used two MSVUD datasets (1437 pairs of images) [48]: FREE-DST (3563 pairs of images) and the BUSY-DST datasets. The FREE-DST consists of images of free roads, while the BUSY-DST dataset consists of images of busy scenes with obstacles (e.g., vehicles, motorbikes, and pedestrians). These two datasets were randomly selected from extracts #10 and #12 of MSVUD.

8.2.3 Results and discussion

The DSA-kNN method was implemented into two steps. In the first step, we constructed the DSA model and established the detection threshold of the kNN detector based on training data with no obstacles. In the second step, we used the kNN detection threshold to detect obstacles in the unseen (remaining) data.

https://t.me/learningnets

8.2.4 Model trained using data with no obstacles

Data believed to be devoid of obstacles were used to construct a reference DSA model. Fig. 8.6 displays four examples of scenes with no obstacles and their corresponding V-disparity maps. The inclined line in the V-disparity map represents the road profile, while the vertical line in the low V-disparity map represents the static environment (Fig. 8.6). These scenes do not include obstacles, as indicated by the free inclined line, and the vehicle is relatively far from the static environment, as indicated by the settled vertical lines with low V-disparity.



FIGURE 8.6 Examples of four scenes with no obstacles: (right) the input image; (left) its corresponding V-disparity map.

Fig. 8.7 shows the evolution of the loss function (cross-entropy) in the function of the number of epochs. An epoch is one forward and backward pass of the training data over the layers of the deep learning model. The learning rate of the DSA model was set at 0.01. As can be seen in Fig. 8.7, that model loss converges at a relatively fast rate and reaches its minimum, which is close to zero, after 120 epochs. Based on this, we conclude that the DSA is able to model and reconstruct a V-disparity map with small errors. As discussed earlier, DSA has a good capacity for learning complex data without assuming anything regarding the underlying input data. Thus, this model will be adopted to detect obstacles in urban scenes.

The DSA we used has four layers. The V-disparity map is used as an input for the first, or input, layer, which is comprised of 600×256 neurons. The second layer, called the first hidden layer, is the first layer used to learn important features and reduce the dimensionality of the input data by 75% ($(600 \times 256)/4$). Then, the output data from the first hidden layer is passed to the second hidden layer, which contains $(600 \times 256)/64$ neurons, in order to extract the remaining information and further reduce dimensionality. The fourth layer, called the

https://t.me/learningnets

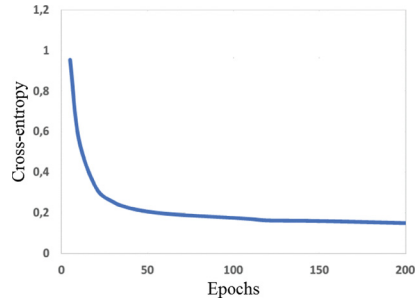


FIGURE 8.7 Line plot of cross-entropy error loss over training epochs.

output layer, contains 1024 neurons, representing the dimension of the extracted features.

The process can handle 12 frames per second using a current PC. The PC we used is equipped with an Intel i7 CPU and Intel Streaming SIMD Extensions technology, allowing for real-time implementation.

Next, the DSA model was used with the kNN algorithm to reveal obstacles in an urban environment. To illustrate the efficiency and effectiveness of DSA-kNN for extracting relevant features from data and distinguishing free scenes from busy scenes, we tested the DSA-kNN approach using training data sets of different sizes. Fig. 8.8 depicts the outcome of the DSA-kNN approach when using differently sized samples of FREE-DST data. Inliers are considered true positives and outliers are false positives. It is clear from Fig. 8.8 that the accuracy of the DSA-kNN approach is improved when the size of the training samples is increased. The best performance was achieved with 5,000 training samples.

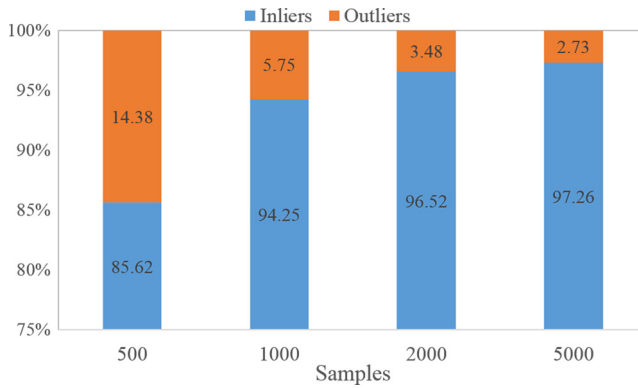


FIGURE 8.8 Performance of the proposed DSA-KNN using the FREE-DST dataset.

https://t.me/learningnets

8.2.5 Evaluation of performance for busy scenes

We assessed the performance of the DSA-based kNN obstacle detection approach based on its ability to detect obstacles in busy scenes in an urban environment. Fig. 8.9 illustrates two examples of busy scenes from the BUSY-DST dataset. In the V-disparity maps in this figure, there are vertical clouds of points settled on the road profile, indicating the presence of obstacles. This characteristic of V-disparity maps could be useful for obstacle detection. This confirms that the V-disparity map is a sensitive indicator of obstacle detection, as discussed earlier.

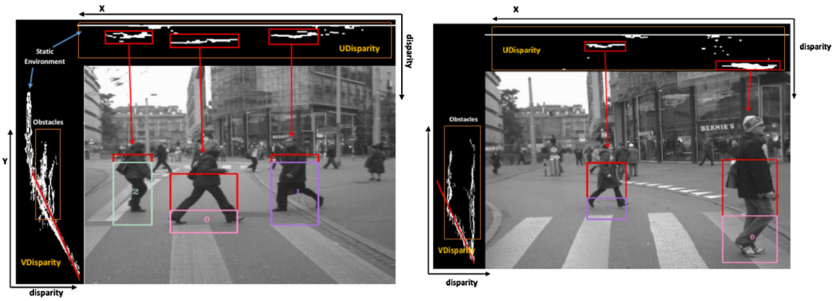


FIGURE 8.9 Examples of scenes with obstacles with V-disparity and U-disparity maps.

We also compared the performance of the DSA-based kNN approach to that of DSA- and deep belief network (DBN)-based clustering algorithms (i.e., kNN, KM, MS, EM, BIRCH, SC, AG, and AP) using the same parameters for the clustering schemes (Table 8.1). The DBN model had 600×256 neurons in the input layer and 1024 neurons in the output layer. For more details on DBN, see [26].

Similar to DSA-based methods, in DBN-based methods, the DBN model is used to extract relevant information from an input V-disparity map and then clustering schemes are applied to the DBN's features to detect obstacles (Fig. 8.10). In order to select the most effective approach for detecting obstacles, we used the

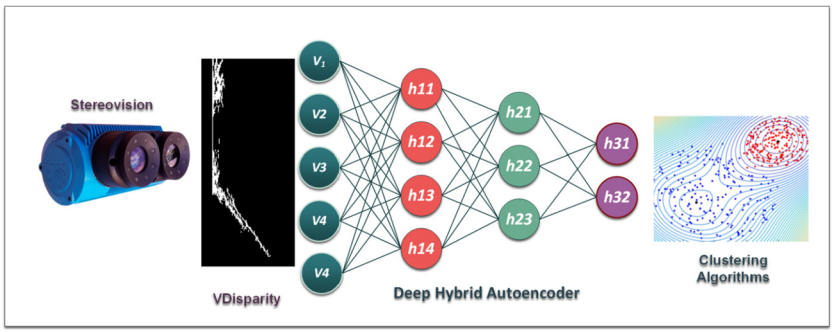


FIGURE 8.10 DBN clustering obstacle detector.

https://t.me/learningnets

TABLE 8.1 Values of parameters used in the studied schemes.

Models	Parameter	Value
KNN	weights algorithm metric leaf_size	uniform BallTree minkowski 30
Autoencoder	Learning rate Training epochs	0.01 100
MS	bandwidth	0,44
AP	damping iteration convergence	0.5 200 15
AG	affinity linkage	euclidean ward
Birch	threshold branch	0.5 50
KM	init init iteration	k-means++ 10 300
SC	affinity gamma	rbf 1.0
EM	covarianceType covar iteration	full le-06 100
Operating area	Disparity Min Disparity Max	32 (pixels) 64 (pixels)

following evaluation metrics: true positive rate (TPR), false-positive rate (FPR), precision, accuracy, area under curve (AUC), and F-measure. The outcomes of the DSA-based and DBN-based obstacle detection methods for the MSVUD and DUSD data sets are displayed in Figs. 8.11A and 8.11B, respectively.

The results shown in Fig. 8.11A indicate that the DSA-kNN method achieved the highest AUC (0.91), followed by DSA-AG (AUC = 0.77) and DSA-EM (AUC = 0.67). The remaining DSA-based clustering methods did not achieve suitable results. Thus, we can conclude that obstacle detection is significantly improved when using the DSA-kNN method.

As shown in Figs. 8.11B and 8.12, DSA-kNN achieves clearer separation between free scenes and scenes with obstacles compared to DBN-kNN. The AUCs achieved by DSA-kNN and DBN-kNN are 0.91 and 0.86, respectively. Based on these results, DSA-kNN is better than DSA-based and DBN-based clustering algorithms for obstacle detection.

https://t.me/learningnets

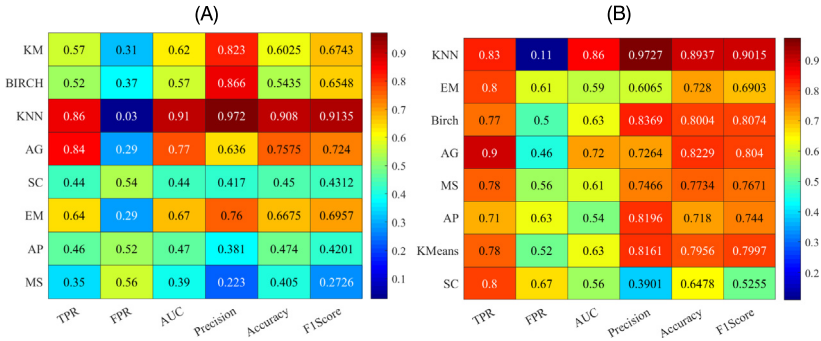


FIGURE 8.11 Heatmap depicting the detection performance of DSA-based clustering algorithms when applied to MSVUD and DUSD datasets. (A) DSA-based obstacle detection schemes; (B) DBN-based obstacle detection schemes.

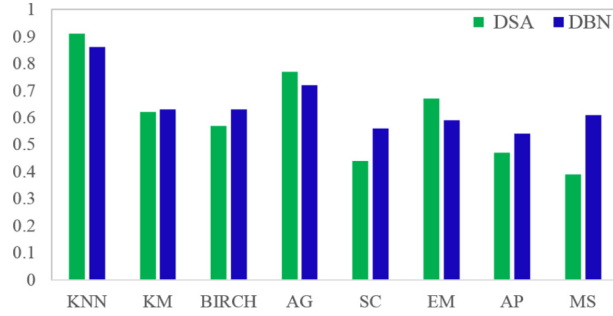


FIGURE 8.12 Comparison of the performance of the DSA-kNN algorithm and other algorithms in terms of AUC.

The major reason is due to the great feature extraction ability of DSA and the extended capacity of the kNN scheme to uncover and separate normal form abnormal features. Also, kNN is nonparametric and an assumption-free algorithm that skips assumption on data convexity (cluster shape) as the case when using the EM algorithm. Another interesting observation is that there is a priori information about the number of clusters when applying kNN and it is insensitive to data ordering as in the BIRCH scheme. Additionally, kNN is flexible and does not require the concept of center (centroid), it can deal with high multivariate data, and it possesses robustness to noise measurements, which is not the case of some schemes including agglomerative and k-means.

8.2.6 Obstacle detection using the Bahnhof dataset

Then, we evaluated the performance of the previously presented obstacle detection schemes using Bahnhof data sets collected from busy inner-city scenes, focusing only on pedestrian tracking-by-detection [51]. It is designed to address the multiperson detection and tracking problem in crowded pedestrian areas us-

https://t.me/learningnets

ing a stereo rig placed on a mobile platform. The data consist of 800 stereo image pairs. We selected 520 scenes with no obstacles and 280 scenes with obstacles. The goal was to reveal obstacles at any time point within the image pairs by constructing a deep-learning-based model that distinguishes between obstacle-free and busy scenes. Specifically, we aimed to use the data to check the performance of DSA- and DBN-based clustering methods for detecting obstacles (i.e., pedestrians) in busy scenes. We first used the data with no obstacles (i.e., pedestrians) to build the DSA- and DBN-based clustering methods. Then, the constructed models were applied to the testing data (i.e., 280 scenes with obstacles) for obstacle detection. The extracted features from deep learning models (i.e., DSA and DBN) served as input for the clustering schemes during the fault detection task. Fig. 8.13A–B illustrates the outcome of directly applying DSA- and DBN-based clustering methods to the Bahnhof testing data. The columns in the heatmaps represent evaluation metrics, and the rows represent the applied methods. Fig. 8.14 displays the AUC values achieved by the DSA- and DBN-based obstacle detection methods when applied to the testing data.

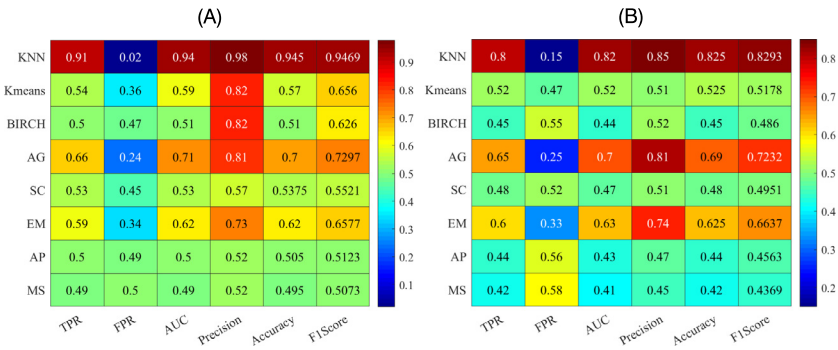


FIGURE 8.13 Detection performances of (A) DSA-based, and (B) DBN-based clustering schemes when applied to Bahnhof dataset: (A) DSA-based obstacle detection schemes; (B) DBN-based obstacle detection schemes.

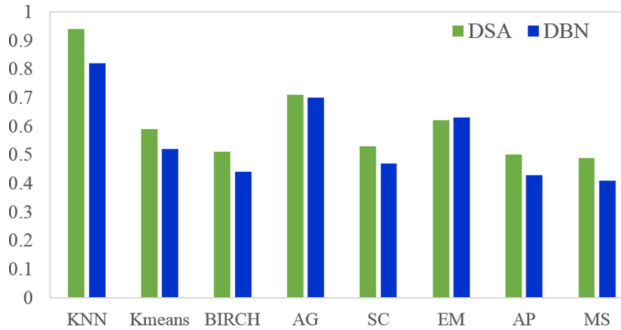


FIGURE 8.14 Comparison of AUC's DSA and DBN-based clustering methods.

https://t.me/learningnets

The results show that the DSA-based clustering algorithms have higher detection performance than the DBN-based clustering algorithms. In other words, the features extracted using DSA based on the V-disparity map are more informative and sensitive to the presence of obstacles than the DBN model. This confirms the results obtained based on MSVUD and DUSD data sets. Furthermore, compared with the other types of machine learning, DSA-kNN has higher detection performance and achieved an AUC of 0.94. DBN-kNN achieved an AUC of 0.82. The obtained results demonstrate that DSA-kNN has a good capacity to detect obstacles in urban environments and crowded pedestrian zones, mainly due to the capability of DSA to extract relevant features from the input V-disparity data, which are sensitive to obstacles. In addition, the DSA is an assumption-free model that can very flexibly extract linear and nonlinear features from data using greedy learning properties, and it can reduce the dimensionality of input data. Interestingly, applying the kNN lazy learning procedure to the DSA-kNN method allowed us to handle nonlinear features, leading to greater performance than the other DSA-based binary clustering methods. Thus, overall, the combined DSA-kNN detection method outperformed the other alternatives.

In summary, accurately detecting obstacles in an urban environment is indispensable for obstacle avoidance in several applications related to autonomous vehicles and mobile robots. We presented an unsupervised deep learning method for obstacle detection that integrated DSA as a modeling framework with the kNN detection scheme. V-disparity maps computed using the stereo image pairs were used as inputs for the deep learning model. This approach exploits the sensitivity of V-disparity maps to the presence of obstacles in order to improve obstacle detection accuracy. New features are constructed and learned by DSA to lower the dimensionality of the space and improve detection performance. In addition, kNN is used to identify abnormal features and flag potential obstacles in front of the autonomous vehicle. The experimental results were obtained with three publicly available datasets: MSVUD, DUSD, and the Bahnhof data set. Also, the DSA- and DBN-based binary clustering methods were compared. The verification results reveal that DSA-kNN is a promising approach to reliable obstacle detection in road environments. Since it achieved acceptable results when faced with real-time constraints (i.e., processing at least 10 frames per second using a PC), it is expected that the performance could be further improved by using a GPU and parallelization for real-time operation.

In this study, obstacle detection in road environments and crowded pedestrian zones was performed under good weather conditions. However, changes in weather conditions and illumination could affect the quality of the collected stereovision images, reducing obstacle detection performance. Therefore, in future work, we will investigate the performance of the aforementioned methods under poor weather conditions and work to develop a stereovision-based obstacle detection method that effectively detects obstacles under such circumstances.

https://t.me/learningnets

8.3 Detecting abnormal ozone measurements using deep learning

8.3.1 Introduction

In recent years, great attention has been paid to air quality monitoring worldwide, particularly after the 2015 United Nations Climate Change Conference (COP 21) [50,51]. Ozone (O_3) is a critical air pollutant that has a negative impact on human health [50,51]. It is created by a chemical reaction of volatile organic compounds (VOCs) and nitrogen [50,51]. High concentrations of ozone, which are usually observed in the summer under high temperatures and in bright sunlight conditions, can lead to an air quality issue called photochemical smog [52–54]. High levels of ozone are frequently reported in urban zones, particularly in industrial countries, such as China and France [55,56]. Reliable detection of ozone pollution is crucial for informing and alerting people of the need to avoid exposure to severe pollution, protecting public health, and managing air quality [57,58].

Over the years, several methods for detecting atypical ozone pollution have been developed. In [59], the multivariate Seasonal AutoRegressive Moving Average with eXogenous variable (SARMAX) method was used with a constrained generalized likelihood (CGLR) approach for detecting abnormal ozone levels, which may be caused by sensor faults or ozone pollution. SARMAX was used as a reference model to describe the ambient temperature and generate the residuals that were used by the CGLR approach for identifying abnormal measurements. Ozone measurements from the Upper Normandy region of France were used for validation. However, the SARMAX is a linear model and therefore was not appropriate to describe nonlinear features of ozone data. In [59,56], a statistical method for detecting abnormality in ozone concentrations was proposed. This method is based on the joint use of principal component analysis (PCA) and the multivariate exponentially weighted moving average (MEWMA). MEWMA was used to monitor the principal components with the smallest variances (i.e., residuals). The sensitivity of MEWMA to incipient changes was investigated to improve the detection capability of the PCA-based approach. Since the PCA is a static model, it failed to describe dynamic features in data. Additionally, the MEWMA scheme was designed based on the assumption of Gaussianity, which limits the efficiency of this approach for detecting non-Gaussian data. The method described in [60] used stochastic modeling and a generalized likelihood ratio (GLR) test to detect biases in the sensors used to measure ozone concentrations and, in doing so, to improve the quality of the gathered ozone measurements. To obtain a reasonable ozone model, nitrogen dioxide concentrations and temperature values were used as input variables. The GLR detection threshold was determined based on Monte Carlo simulations. This kind of algorithm could be implemented to further improve the maintenance of air pollution monitoring sites by detecting faulty instruments, but it is not designed for promptly detecting abnormal ozone pollution. Other works focused on assess-

https://t.me/learningnets

ing data quality to minimize the costs of managing ozone air-quality monitoring networks [61–63]. For instance, In [63], an approach using a winner filter and hypothesis testing was introduced to detect sensor malfunctions in air quality monitoring networks. Daily ozone measurements from the Houston air quality monitoring network were used for validation. The proposed method involves applying individual thresholds to every sensor so that the false alarm rate can be approximately fixed within a network. Based on the analysis performed in [63], studying real-time detection of abnormal ozone measurements is important for not only checking the accuracy of monitoring sensor networks and checking the quality of collected data but also for alerting the population when ozone pollution exceeds the warning limits.

Real-time detection of atypical ozone measurements can prevent further damage to sensor networks and offer relevant information to people in order to improve human safety and avoid undesirable consequences. Ozone pollution can be detected using model-based or data-based monitoring techniques. Designing an accurate analytical model of ozone variation is not always feasible and is time-consuming due to the complex process through which ozone is produced in the troposphere and high measurement uncertainty [64,65]. In many studies, when an analytical model describing ozone variations was not available, data-driven methods were applied to predict ozone pollution. These methods include time-series models (e.g., the autoregressive moving average, or ARMA) [66,67]; multiple linear regression methods [60, 69]; and multivariate latent variable regression methods, such as PCA, partial least squares, and principal component regression [59,68]. Recently, modeling and predicting air pollutant concentrations using machine learning techniques have become popular research areas. Several shallow learning models, including support vector regression (SVR) [69,70], fuzzy logic modeling [71], classification and regression trees (CART) [72], and artificial neural networks (ANNs) [66,70], have been applied to improve the quality of ozone measurement predictions due to their ability to capture nonlinear features. However, as these methods have a one- or two-layer structure, they are limited in capturing relevant features from complex and high-dimensionality data [73,74].

Due to the limitations of the above-mentioned shallow methods, deep learning methods such as stacked autoencoders [75] and DBNs have attracted increasing attention in recent years, becoming some of the most widely used methods for feature extraction and prediction. Unlike shallow methods, deep learning methods generally lead to an improved abstraction of the original data for modeling and prediction. A major advantage of deep learning models is their ability to learn the desired features from complex processes layer by layer. Furthermore, they are flexible, simple, and assumption-free. For these reasons, when extracting important characteristics, it is imperative to use deep structures [73, 74]. Successful large-scale applications of deep learning have improved traffic management [76,77], health informatics [78], pattern recognition [79], detection of cerebral microbleed voxels [80], and air quality modeling [81].

https://t.me/learningnets

As the concentration of ozone pollutants is increasing in industrial countries, such as those in Europe, and is frequently surpassing safe limits, particularly in the summer, it is crucial to monitor these pollutants to avoid harmful effects on human health. Shallow methods achieve unsatisfactory accuracy when used to predict pollution [64,65]. Therefore, as discussed above, deep learning models are essential for modeling and anomaly detection purposes, as they can effectively extract useful features from multivariate processes. To overcome the limitations of shallow methods regarding the detection of abnormal ozone measurements, we suggest using a deep learning model combined with an unsupervised binary detector. In our study, DBNs, which are efficient for dimensionality reduction and feature extraction, are used to model multivariate time series ozone measurements. This model is trained using greedy layerwise pretraining, and it is employed to learn features of the ozone level. To detect anomalies, the DBN model is coupled with a one-class support vector machine (OCSVM). This method benefits from both DBN's capability to extract features from high-dimensionality data and OCSVM's capacity to discriminate between normal and atypical features. OCSVM is a preferred algorithm for anomaly detection due to its flexibility to discriminate between linear and nonlinear features without constraints related to the distribution of data. In the detection methodology applied in this case study, ozone concentration measurements were collected from one monitoring site of the Isère monitoring network. Then, we compared the performance of DBN-based methods to DSA- and RBM-based clustering methods.

Section 8.3.2 presents a brief description of the ozone data we used. Section 8.3.3 introduces the main idea underlying the DBN-OCSVM detection approach. In Sect. 8.3.4, we apply and illustrate the used methodology using real ozone datasets. Then, we provide some concluding remarks.

8.3.2 Data description

The ozone data considered in this study was collected from the Isère region of France. Fig. 8.15 shows the geographical location of the studied site. The air quality network in this urban region is managed by the Atmo Auvergne-Rhône-Alpes Association. Ozone data are measured every hour by 14 stations: eight urban stations, five peri-urban stations, and one rural station (Table 8.2). The stations are labeled S1–S14, as shown in Table 8.2. The spatial distribution of the measurement stations is shown in Fig. 8.16.

There are two kinds of anomalies in ozone data, true ozone pollution and abnormalities related to sensor defects (called false anomalies). There is a clear distinction between these two types of abnormalities. True ozone pollution results from chemical reactions in the atmosphere under certain conditions, such as sunny days with humid air conditions and high temperatures [56]. This type of ozone pollution is characterized by a progressive increase in the concentration of ozone within a few hours (Fig. 8.17A). On the other hand, false anomalies

https://t.me/learningnets

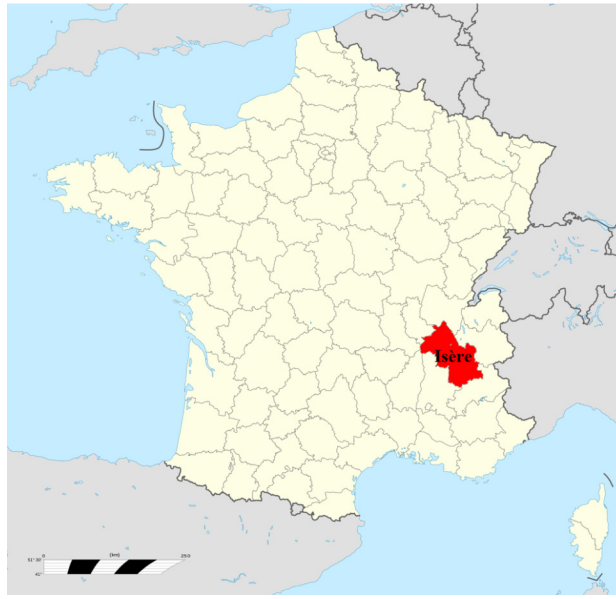


FIGURE 8.15 Geographic location of the studied site.

TABLE 8.2 Investigated ozone measurement sensors in the Isère region.

Network	Type	Label
Sud grenoblois/Champ sur Drac	Peri-urban	S1
Fontaine les Balmes	Urban	S2
Voiron Urbain	Urban	S3
Saint-Martin d'Heres	Urban	S4
Grenoble les Frenes	Urban	S5
Sud grenoblois/Vif	Peri-urban	S6
Est grenoblois/Grésivaudan	Peri-urban	S7
Grenoble Caserne de Bonne	Urban	S8
Roussillon	Urban	S9
Les Roches de Condrieu ZI	Peri-urban	S10
Vienne Centre	Urban	S11
Sud roussillonnais/Sablons	Peri-urban	S12
Bourgoin-Jallieu	Urban	S13
Plateau de Bonnevaux	Rural	S14

due to sensor anomalies are mainly generated by biases in the measurement sensors. These anomalies are characterized by an unusually high level of ozone concentration in a very short time period (i.e., less than an hour; Fig. 8.17B).

https://t.me/learningnets

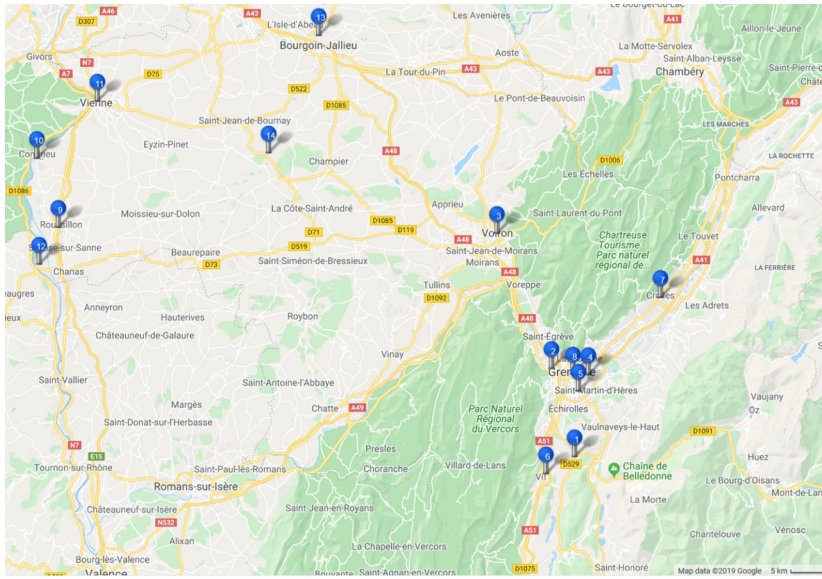


FIGURE 8.16 Spatial distribution of ozone measurement stations.

The goal of this study is to apply a deep-learning-based monitoring algorithm to detect abnormal ozone measurements and malfunctions in sensor networks. The monitoring technique adopted in this study is a DBN-based OCSVM technique, which is briefly introduced in the next section.

8.3.3 Ozone monitoring based on deep learning approaches

As discussed previously, in cases in which an analytical model is not available to describe process variations, data-driven models can be applied. In this study, we used unsupervised deep-learning-based approaches to monitor ozone pollution. In this section, we describe the structure of the proposed DBN-OCSVM monitoring method (Fig. 8.18). DBN is employed to capture the relevant features of ozone concentrations, and OCSVM is applied to detect anomalies. This monitoring method is nonparametric because it does not require specification related to the distribution of process data, making it able to very flexibly handle ozone data.

As mentioned above, DBN models are assumption-free, and they are able to extract important information from high-dimensionality data and reveal the nonlinearity of processes [82]. In the DBN model applied in this study, which contains five layers of RBMs, learning is based on greedy layerwise pretraining and the outputs of each layer are employed as inputs by the next layer [83]. This means that every layer of the DBN is trained independently. The data used to train the first layer are gathered from sensors. It is essential to use deep structures

https://t.me/learningnets

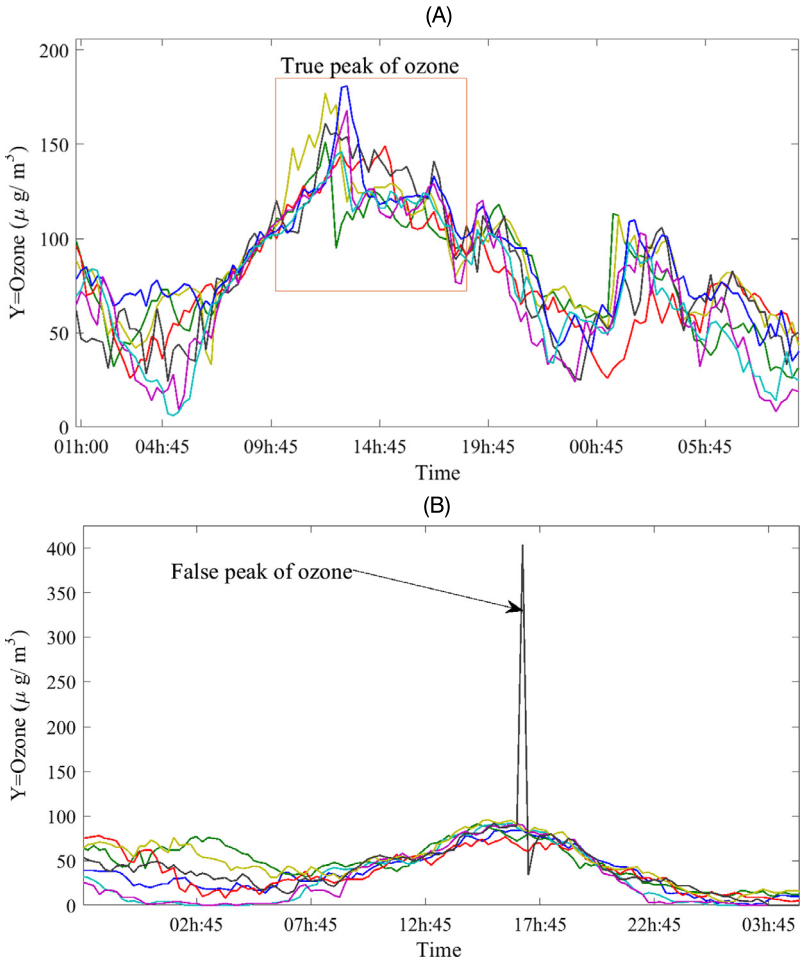


FIGURE 8.17 (A) Atypical ozone pollution and (B) high ozone concentration due to sensor failure.

to efficiently represent data. Then, more hidden units are stacked to the input layer to enhance the flexibility and efficiency of the DBN model. In the first step of feature extraction using the DBN model, input data are passed through the first (i.e., visible) layer, producing the first features. In the next step, the same operation is performed, passing the previously extracted features through the second layer to obtain new features. This process is repeated for all subsequent layers; each layer learns from the input data and produces new extracted features that to be used by the next layer. This method, called greedy layerwise training, was introduced to construct DBN models in [84]. In this study, the OCSVM was trained using the features generated by the last layer of the DBN model. Then, the hybrid model was used for anomaly detection in new testing data.

https://t.me/learningnets

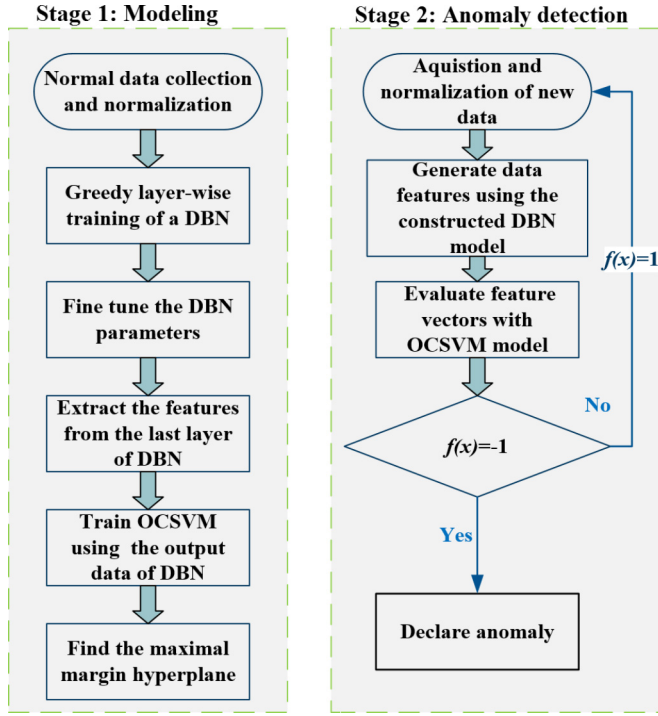


FIGURE 8.18 Hybrid DBN-OCSVM monitoring framework.

8.3.3.1 *Results and discussion*

In this section, we first present a preliminary analysis of the considered ozone dataset. Then, the performance of the deep-learning-based methods for detecting abnormal ozone measurements is evaluated. Finally, we compare the detection results of the DBN-OCSVM approach to those of the standalone clustering methods and the DSA-based clustering methods.

The data chosen to develop the DBN model is anomaly-free, and it was collected from January 1 to March 4, 2015 (see Fig. 8.19). A heatmap of the correlation matrix of the ozone data is presented in Fig. 8.20. As can be seen in the figure, data from the majority of stations are highly cross-correlated. Additionally, there are three clear groups: stations S1–S8, stations S9–S13, and station S14. The data within the first and second groups shows high cross-correlation because they are in close proximity to each other (Fig. 8.16). The data from station S14 have a relatively low correlation with data from the other stations. This is because the data from station S14 were collected from a rural zone, and the variation of ozone in rural zones differs from the variation in urban and peri-urban zones. Specifically, in rural zones, the level of ozone is higher at night because of the low presence of ozone destroyers.

https://t.me/learningnets

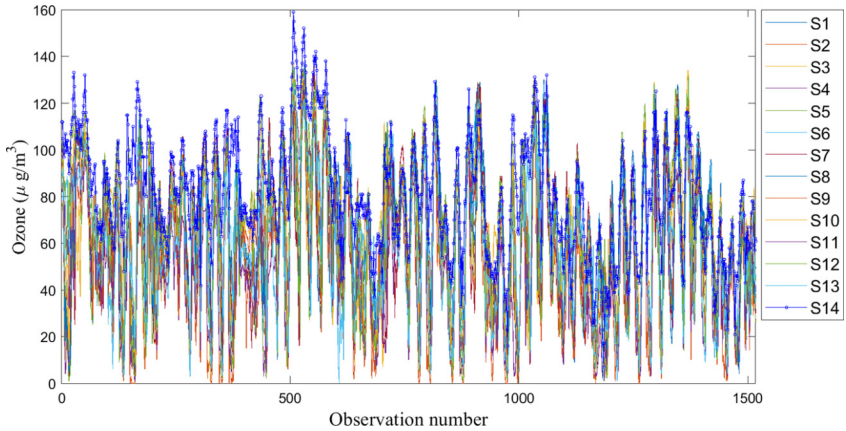


FIGURE 8.19 Hourly concentrations of O_3 (mg/m^3) without anomalies.

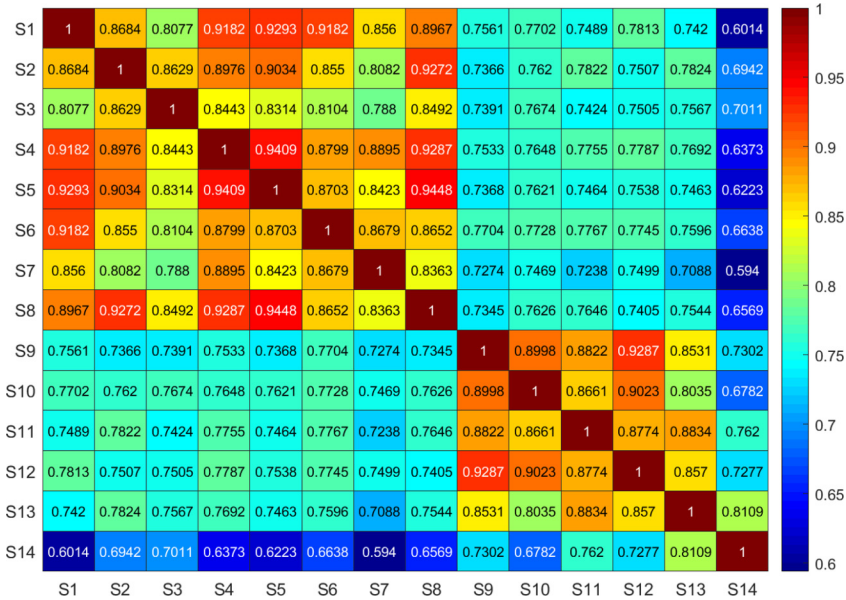


FIGURE 8.20 Heatmap of the correlation matrix of the O_3 training measurement.

Fig. 8.21 shows a boxplot of the training data set by station. Based on the analysis of this boxplot, we can confirm that the behavior of the ozone concentrations in the rural zone (S14) is different than that of the ozone concentrations in urban and peri-urban zones (S1–S13). Moreover, the distribution of data from all stations is almost symmetrical.

Fig. 8.22 illustrates the autocorrelation function (ACF) plots of the ozone concentrations at each location. Fig. 8.22 shows the distribution of the hourly

https://t.me/learningnets

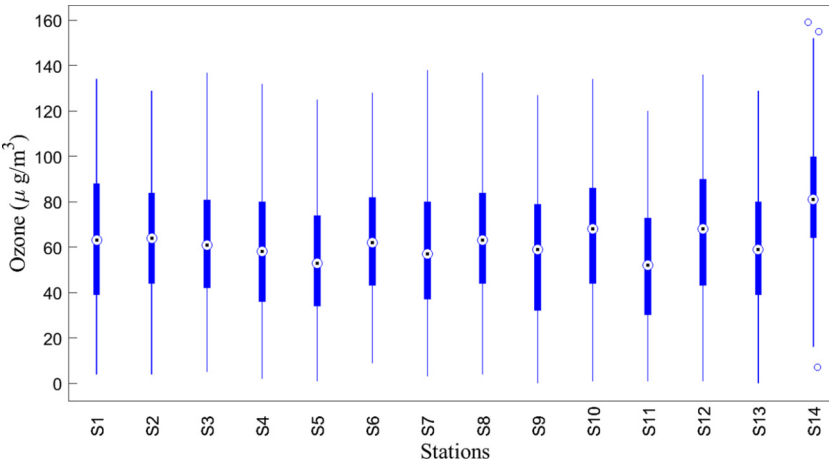


FIGURE 8.21 Boxplot graphs of the O_3 training measurements for all the 14 stations.

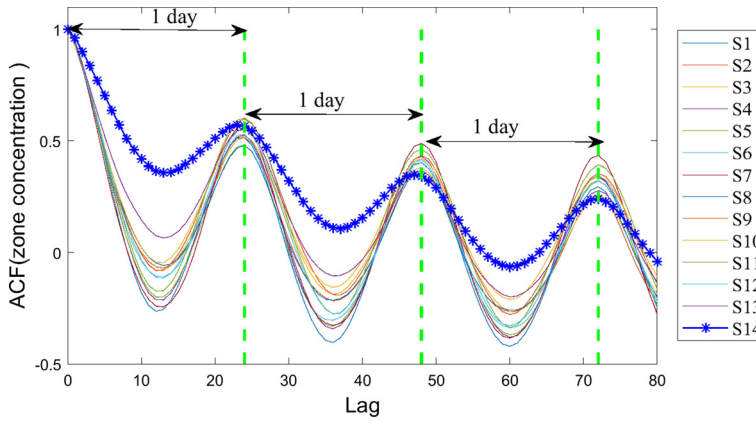


FIGURE 8.22 ACF of O_3 training measurements by stations.

ozone variations for each station. As expected, the hourly mean shows a bell-shaped distribution with a peak around noon, indicating a diurnal ozone cycle resulting from the diurnal temperature cycle. A cycle is determined by the time distance between two respective maxima in the ACF. In this ozone data set, the cycle is one day. There is a clear cycle in the variation of ozone concentrations, with maximum values late at night and minimum values around noon (Fig. 8.17). The ozone level starts to increase after sunrise and reaches its maximum by the afternoon. This is mainly due to the photochemical formation of ozone due to the oxidation of volatile organic compounds (VOC) with a sufficient quantity of NOx under sunlight conditions. Particularly high ozone levels can be observed in the summer period.

https://t.me/learningnets

In this study, the ozone data set at the training stage was collected from January 1 to March 4, 2015. This training data set is known to be devoid of anomalies. The DBN model was trained based on the manually selected parameters presented in Table 8.3. During the training process, the hyperplane of the OCSVM was established based on the features extracted from the DBN model. OCSVM was used with a radial basis function (RBF) kernel with parameter values of $\gamma = 0.1$ and $\nu = 0.001$.

TABLE 8.3 Selected parameters for the trained DBN model.

Models	Parameter	Value
DBN	Layers	3
	Units	17
	Learning rate	0.001
	Epochs	1000

To identify a satisfactory model for the ozone measurements, the loss between the input ozone measurements and the model predictions was computed within a fixed number of batches. For this purpose, we used the cross-entropy error function, which is commonly used to verify the quality of the model predictions so that the weights of the parameters for the deep learning model can be updated to reduce loss in the next evaluation. Generally, the model is trained until the cross-entropy error converges near zero. Fig. 8.23 displays the cross-entropy loss over the 180 training epochs, demonstrating that the cross-entropy exhibited good convergence.

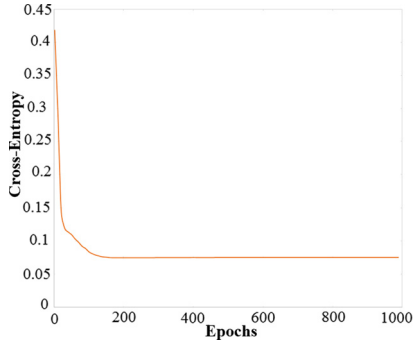


FIGURE 8.23 Line plot of cross-entropy error loss over the training epochs.

Next, we will use the previously identified DBN model with the OCSVM to detect anomalies in ozone measurements, which may result from photochemical ozone pollution or malfunctions in the sensor network.

https://t.me/learningnets

8.3.4 Detection results

8.3.4.1 Sensor anomaly detection: false anomalies

In the following section, we assess the ability of the DBN-OCSVM method to detect malfunctions and bias in measurement instruments (i.e., faults in sensor readings), which may be due to sensor failure or calibration problems. Sensor faults lead to unusually high ozone levels (150 to over $\mu g/m^3$) measured during the night and outside the summer season, which are not typically the peaks for photochemically produced ozone.

To test the performance of the previously constructed DBN-SVM detection method, three types of sensor faults are considered: bias or fault in a single sensor, biases in multiple sensors, and intermittent faults. This method, which uses DBN with three layers, is compared to RBM; DBN2 (which features two hidden layers); DSA-based OCSVM; and the DBN-based expectation-maximization (EM) [85], BIRCH [86], and *k*-means [87] methods. The parameters selected for the considered methods are presented in Table 8.4.

TABLE 8.4 Values of the selected parameters for the considered methods.

Models	Parameter	Value
Autoencoder	weights	uniform
	Learning rate	0.01
	Training epochs	100
BIRCH	branch	50
k-means	init	k-means++
	init	10
	iteration	300
EM	covarianceType	full
	covar	1e–06
	iteration	100

8.3.4.1.1 Case A: single abrupt fault

In the first scenario, we focused on assessing the performance of the DBN-OCSVM method in the presence of a bias sensor fault due to a degradation of the measurement instrument. We first introduced different levels of bias (between 5% and 100% of the total variation found in the raw data of S_2 measurements). Then, we applied the DBN-OCSVM scheme and calculated AUC for different levels of bias. Every phase of the DBN-OCSVM method provided complementary information to enable fault detection. This monitoring method is beneficial due to its desirable properties and the possibility of adapting it to more complex processes. The AUC values achieved from the DBN3-, DBN2-, RBM-, and DSA-based OCSVM methods and OCSVM alone with different levels of bias are displayed in Fig. 8.24.

https://t.me/learningnets

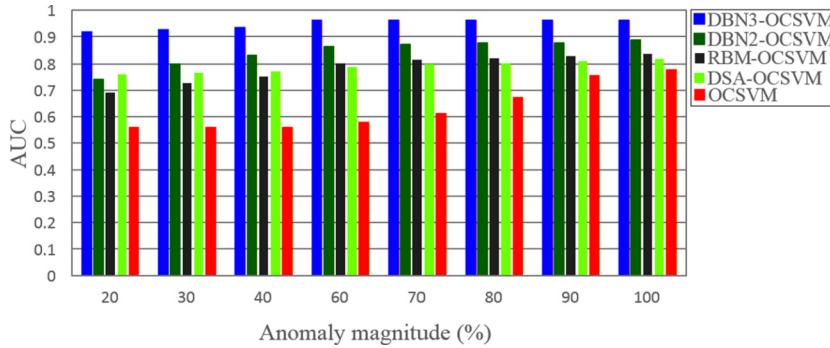


FIGURE 8.24 Comparison of the performance of DBN-OCSVM with other algorithms in terms of AUC at different levels of bias (Case A).

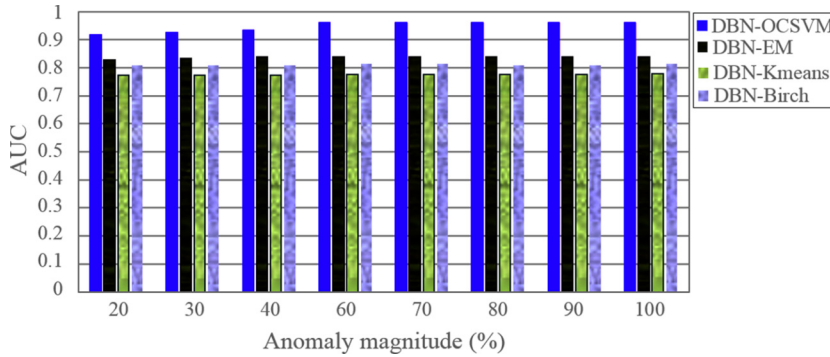


FIGURE 8.25 Comparison of the performance of the DBN-based detection algorithms in terms of AUC at different levels of bias (Case A).

This method outperformed the RBM-, DBN2-, and DSA-OCSVM methods, as well as the shallow OCSVM algorithm, which was applied directly to the ozone measurements without including the feature extraction steps. The results demonstrate that the method with a deeper network (DBN3-OCSVM) achieved better detection results than RBM-OCSVM or DBN2-OCSVM. This demonstrates the benefits of deep learning methods over shallow OCSVM, such as the ability to reveal information that is more useful for detecting abnormalities in ozone data. In addition, it was found that the DBN model extracted more informative features for anomaly detection than the DSA model.

The second comparison was conducted between DBN-OCSVM and the DBN-based EM, *k*-means, and BIRCH algorithms. The results are presented in Fig. 8.25. In this experiment, the clustering algorithms use the features extracted from the last layer of the DBN model. It was found that combining DBN with OCSVM resulted in a significantly greater detection ability than when DBN was combined with the other clustering algorithm (i.e., EM, *k*-means, and BIRCH). OCSVM is an assumption-free algorithm that, unlike the EM algorithm, does

https://t.me/learningnets

not assume data convexity. It maps the input ozone data to higher-dimensional space using a kernel function to separate normal and abnormal features. Moreover, OCSVM is not sensitive to the rank of data records, unlike the BIRCH algorithm. Further, it is easy to implement and requires only a training data set that is devoid of anomalies. No labeling is required in the training phase. Overall, DBN-OCSVM showed a better ability to identify abrupt changes in ozone time-series data than the other considered algorithms.

8.3.4.1.2 Case B: multiple abrupt faults

Case B was used to verify the ability of DBN-OCSVM to detect multiple sensor faults. To this end, a bias fault was incorporated into the ozone measurements of sensors S_3 and S_6 with a time interval [240, 300] between the measurements. The detection results of the DBN3-, DBN2-, RBM-, and DSA-based OCSVM algorithms and the shallow OCSVM algorithms (in terms of AUC values at different fault levels) are shown in Fig. 8.26. The results confirm that it is crucial to use deep learning models (e.g., DBN3) to effectively learn the features of ozone data and improve anomaly detection. DBN-OCSVM successfully distinguished between normal data and faulty data, achieving an AUC of 0.916 when the magnitude of the fault was relatively small (20%). As expected, DBN3-OCSVM detected faults better than the other considered schemes. The shallow OCSVM algorithm was not effective for sensing small changes (AUC = 0.559 in the presence of a fault with a magnitude of 20%). Fig. 8.27 compares the detection performance of the DBN-OCSVM, EM, k-means, and BIRCH algorithms. The results indicate the superiority of DBN-OCSVM for detecting multiple anomalies in ozone data.

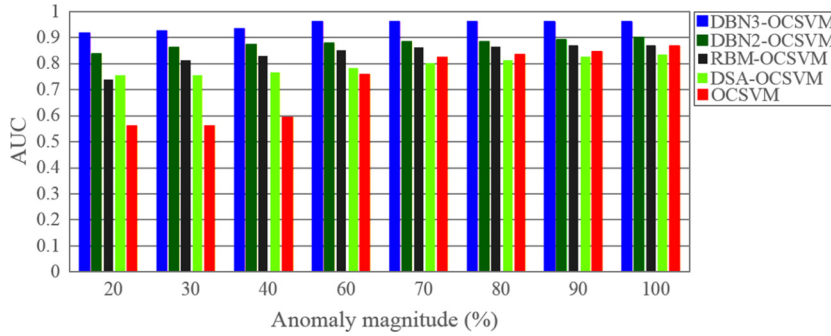


FIGURE 8.26 Comparison of the performance of the DBN-OCSVM algorithm with other algorithms in terms of AUC at different fault levels (Case B).

8.3.4.1.3 Case C: intermittent faults

Case C involved more subtle sensor malfunctions, such as intermittent sensor faults, that repeatedly appear and disappear. Intermittent shifts were incorporated in the ozone measurements of sensor S5 at time intervals of [410, 440] and

https://t.me/learningnets

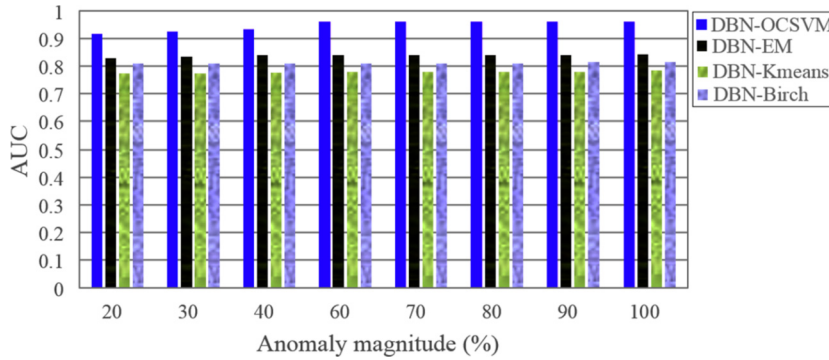


FIGURE 8.27 Comparison of the performance of the DBN-based detection algorithms in terms of AUC at different fault levels (Case B).

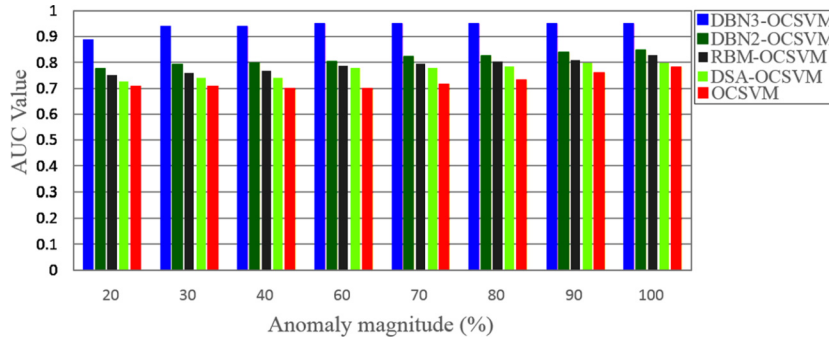


FIGURE 8.28 Comparison of the performance of the DBN-OCSVM algorithm with other algorithms in terms of AUC at different fault levels (Case C).

[502, 520]. Fig. 8.28 depicts the detection performance of the five algorithms (in terms of AUC) as a function of fault level. The results demonstrate that DBN-OCSVM has a good ability to detect intermittent sensor bias. Also, the results confirm that combining the deeper learning model (DBN3) with OCSVM led to better performance than the RBM-, DBN2-, and DSA-based OCSVM algorithms. The results depicted in Fig. 8.29 also confirm that DBN-OCSVM outperforms the DBN-based EM, k-means, and BIRCH algorithms.

8.3.4.2 Conclusion

In this section, we discussed some unsupervised deep-learning-based anomaly detection methods for monitoring ozone measurements. In particular, we focused on the hybrid DBN-based OCSVM monitoring approach to detect abnormalities in ozone measurements. This approach makes a decision about ozone pollution based on observed ozone data collected from the air quality sensor network without the need for prior labeling of data. It is simple to build and convenient to use. The DBN model has a strong ability to learn pertinent fea-

https://t.me/learningnets

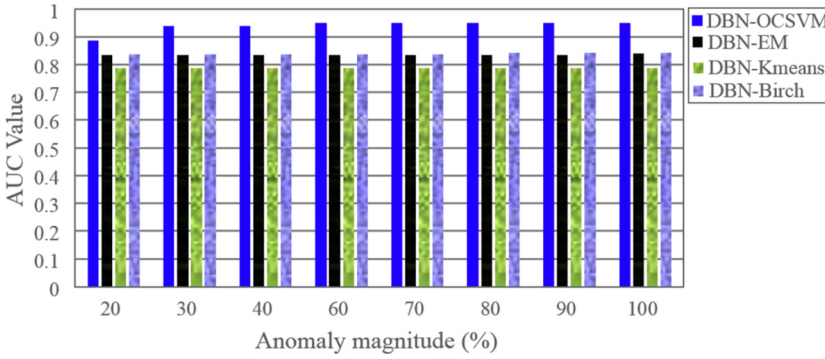


FIGURE 8.29 Comparison of the performance comparison of the DBN-based detection algorithms in terms of AUC at different fault levels (Case C).

tures from data without specifying the data distribution, and OCSVM is an assumption-free algorithm that evaluates the input features obtained from the DBN model using only actual input data. For these reasons, OCSVM and DBN are especially popular in the literature. We verified the performance of the DBN-OCSVM approach using ozone data from a regional air quality network in France. The results indicate that it is able to accurately detect different types of sensor faults and that it has better detection performance than RBM and the DSA-based k-means, EM, and BIRCH algorithms.

8.4 Monitoring of a wastewater treatment plant using deep learning

8.4.1 Introduction

Treating wastewater by removing pollutants is crucial for ensuring the health of communities and the environment. Treated wastewater can be safely discharged or reused for cleaning, irrigation, and industrial purposes [88]. Practically, it is more advantageous to recycle and reuse treated wastewater instead of discharging it [89]. In water-stressed countries, reusing wastewater is vital, as it is a substantial source of water and is associated with a lower cost than desalting seawater [90,91]. Recently, monitoring WWTPs using data-driven methods, such as computational intelligence and machine learning techniques, has gained attention among researchers and engineers.

To guarantee the effective running of WWTPs, several important variables, including pH, dissolved oxygen, and nitrogen, must be continuously monitored [92,93]. To ensure efficient monitoring, a good model that describes the dynamics and variability of WWTP data is indispensable. However, as the WWTP process is complex and uncertain due to changes in physical features, developing a model that produces reliable and accurate descriptions of variability in WWTP data is very challenging. Many methods for understanding

https://t.me/learningnets

wastewater processes, including ASM1, ASM2, first principals can describe linear and nonlinear variation and reveal important process features, but it is challenging to construct a realistic model because it requires prior parameters for calibration and incurs a great time cost, especially for complex processes such as WWTPs [94–96]. For instance, in the ASM1 model, there are 13 nonlinear differential equations involving 19 parameters that are not easy to predict [97]. Generally speaking, analytical models based on first principals could describe linear and nonlinear variation, reveal important process features, but there is a major challenge in constructing a realistic model because it demands prior parameters for calibration and time-costly in particular for complex processes such as WWTPs. The computational complexity of these models constitutes an obstacle to the simulation and design process [98].

Although WWTPs utilize advanced technologies, they are exposed to anomalies and failures that limit their capability and productivity. Misdetected anomalies in WWTPs could seriously affect the health of a WWTP and may result in safety issues. Therefore, accurately detecting anomalies is required to ensure normal operating conditions and achieve the desired performance [99–101]. Much prior research has focused on developing improved anomaly detection methods for monitoring WWTPs [102,103]. Recently, monitoring WWTPs using data-driven methods, such as computational intelligence and machine learning, has gained special attention from researchers and engineers. In data-driven methods, measurements from the WWTPs devoid from anomalies are used to construct an empirical model that is used to verify new measurements.

In the literature, several data-based anomaly detection methods are designed using dimensionality reduction techniques, such as In the literature, several data-based anomaly detection methods have been designed using dimensionality reduction techniques, such as PCA [104], PLS [105], linear discriminant analysis (LDA) [106] and locally linear embedding (LLE) [93]. Kernel extensions of some linear dimensionality reduction algorithms, such as kernel PCA and kernel PLS, have been considered for nonlinear process monitoring because of their ability to extract linear and nonlinear features from nonlinear processes like WWTPs [93,107–112]. For instance, in [102] a method combining PCA and multiple regression was introduced for modeling a WWTP. In addition, the method described in [104] integrated PCA with kNN to monitor influent measurements in a coastal municipal WWTP. In [113], a PLS-based method was applied to monitor effluent quality and filamentous bulking in an activated sludge process. In recent years, numerous machine learning methods have been designed to model and monitor WWTPs [114,115,103,116]. For example, the method described in [103] coupled an ANN with neural fuzzy models to monitor WWTPs. Also, in [116], a neural ANN model was used to monitor an anaerobic WWTP. In [117], a hybrid method using multiple linear regression and ANNs was developed to predict the influence of biochemical oxygen demand, which is costly and challenging to measure. In [118], kernel PCA was

https://t.me/learningnets

coupled with a one-class SVM algorithm containing several kernels to examine a WWTP with influent characteristics (ICs). In [114], a method combining an optimization forecast component model and SVM algorithm was proposed for the detection and diagnosis of faults in WWTPs. However, the above-mentioned data-based methods (both linear and nonlinear) involve shallow learning frameworks.

Recently, numerous deep-learning-based methods with a high learning capability have been developed to serve as advanced versions of neural networks for modeling and monitoring the operating conditions of WWTPs [119–123]. Generally speaking, deep learning models have shown promising results when applied to a variety of fields because of their ability to learn how to effectively represent data. In addition, these models proved efficacious for automatically extracting relevant features from input data. The method described in [123] was based on a recurrent neural network (RNN) and was used to predict TSS, BOD, and NO3 based on time series data. However, this study was based on data with only one source of inlet water with stable quality. In [120], a deep learning model based on a CNN was applied to model membrane fouling during nanofiltration and reverse osmosis filtration using image data obtained from optical coherence tomography. In [121], a hybrid deep learning prediction model combining CNN and LSTM (called CNN-LSTM) was applied to predict dynamic chemical oxygen demand (COD), which is an important indicator of the health of WWTPs. The CNN-LSTM model showed an improved prediction ability compared to the CNN or LSTM models alone when applied to data from an urban WWTP. In [119], an unsupervised deep learning method combining the benefits of the RNN, RBM, and OCSVM methods was proposed for detecting abnormal ICs in a WWTP. This unsupervised deep learning approach achieved good detection performance when applied to data from a coastal WWTP.

In this case study, we present an unsupervised deep learning model to monitor the health status of a WWTP. This model uses the strong learning ability of DBNs to learn important features of data from WWTPs and applies OCSVM to distinguish between normal and abnormal features. We applied the model to data collected from a decentralized WWTP in Golden, CO, USA. The following section briefly describes the methodology used in this study. Section 8.4.3 presents the results of the DBN-OCSVM method and compares them with the results of the kNN-EWMA and k-means schemes.

8.4.2 Proposed DBN-based kNN, OCSVM, and k-means algorithms

This section describes the DBN-OCSVM model, which was used to detect anomalies in a decentralized WWTP. The DBN model has a great learning capacity because it possesses greedy learning features and the ability to reduce dimensionality. It was coupled with the OCSVM algorithm to detect faults in multivariate data (Fig. 8.30). The approach was implemented in two steps. First, a DBN model possessing four layers was trained based on a sample data set

https://t.me/learningnets

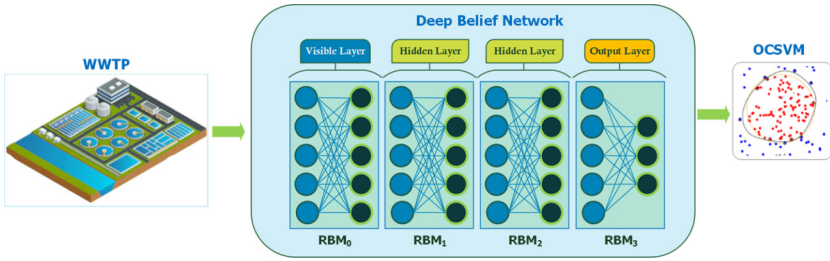


FIGURE 8.30 Framework of the hybrid DBN-OCSVM monitoring scheme.

collected from the WWTP under normal conditions. Then, the OCSVM was applied to the features extracted by the DBN model. The approach was performed in an unsupervised way without any data labeling.

As mentioned above, it is necessary to use a deep learning model to extract important features of data. In this study, we investigate the application of the DBN-OCSVM approach to detect anomalies in multivariate data collected from a WWTP.

8.4.3 Real data application: monitoring a decentralized wastewater treatment plant in Golden, CO, USA

Data gathered from a decentralized WWTP facility in Golden, CO, USA, was used to verify the detection capability of the considered algorithms. The WTTP facility treats wastewater gathered from 400 units of a student housing complex at Mines Park via a sewer diversion [124]. The process generates effluent that can be used for landscape irrigation [124,125]. Small-scale decentralized facilities are gaining more attention, and it is expected that they will be common in the future since the treated wastewater can be locally reused. Practically, since the quality and quantity of influent can highly fluctuate, decentralized processes need an efficient and accurate monitoring system to identify changes or malfunctions early.

The data considered in this study include the ten-minute averages of 28 variables gathered from April 10 to May 10, 2010. This dataset contains real anomalies in both pH and salinity that required the system to be shut down for a period of two months to recover. The data used for monitoring purposes contains seven selected variables: membrane bioreactor (MBR) permeate pressure, MBR dissolved oxygen (DO), permeate turbidity, return activated sludge (RAS) DO content, RAS pH, RAS total suspended solids (TSS), and permeate tank conductivity (Fig. 8.31). Variable selection was based on expert recommendations. The selected variables were spread across the system and had the ability to provide relevant information about the operating state of the examined WWTP. For instance, RAS DO content, RAS pH and RAS TSS provide information about the water circulating in the first set of tanks. pH is a potentially useful indicator

https://t.me/learningnets

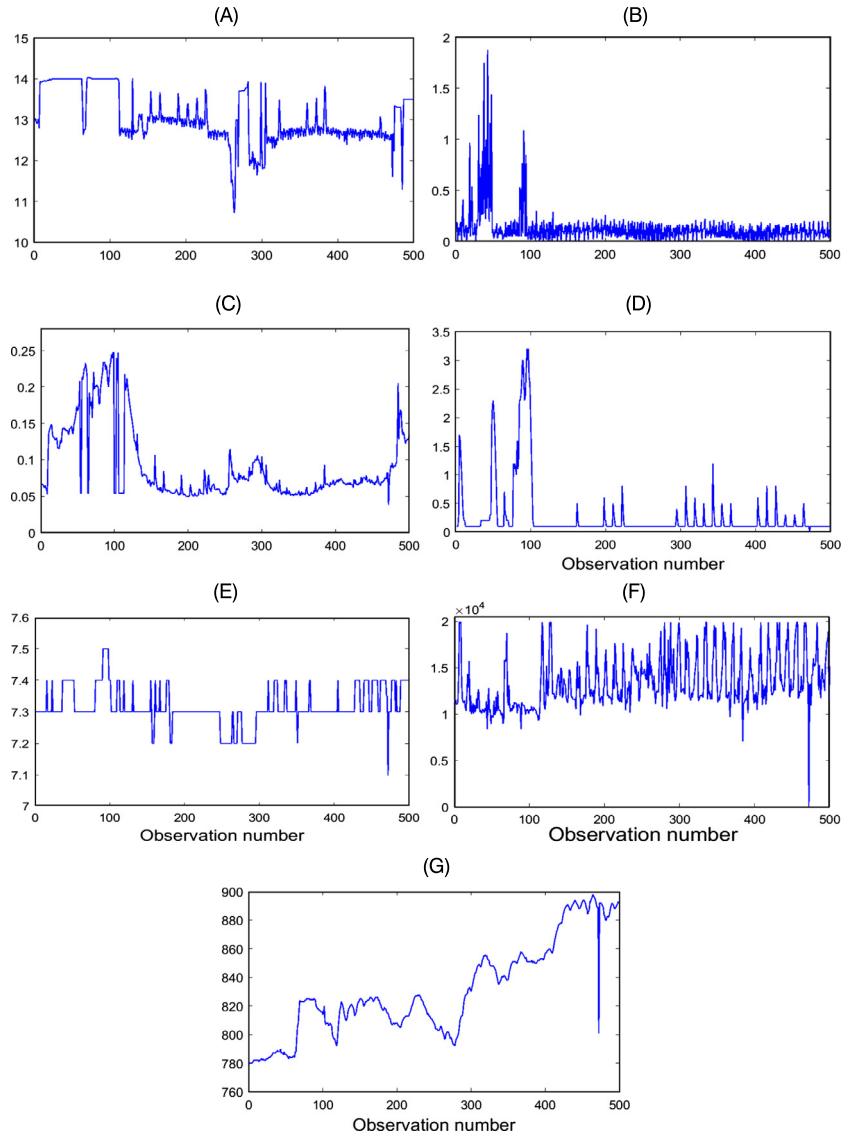


FIGURE 8.31 Training data sets that are devoid of anomalies. (A) MBR 2 permeate pressure; (B) BIO 2 dissolved oxygen; (C) Permeate turbidity; (D) Return activated sludge dissolved oxygen content; (E) Return activated sludge pH; (F) Total suspended solids in return activated sludge; (G) Permeate tank conductivity.

because variation within a certain range is required by biological organisms. In addition, DO and TSS provide pertinent information about the health of biological organisms that are indispensable for transforming ammonia into nitrogen.

https://t.me/learningnets

The MBR permeate pressure, permeate turbidity, and permeate tank conductivity are directly related to the second half of the process, in which water is spread through a membrane. Transmembrane pressure is vital for verifying the health status of the membrane, while turbidity and conductivity can be useful for checking the quality of the water issued out of the bioreactors. Finally, the level of MBR DO inside the bioreactor tank is a very important variable that should be monitored due to its impact on health status, treatment efficiency, and operating expenses [125].

The monitored system possesses two MBRs that operate in a similar manner. In this study, we inspected the second one. More details about the system are presented in [126]. Here, we used 500 observations devoid of anomalies to train the DBN-based methods (Fig. 8.31).

Table 8.5 summarizes the descriptive statistics of the training data. The mean provides an indication of the location of the distribution of every variable, while the standard deviations, extremes, and quartiles provide useful information about the spread in the data. We used skewness and kurtosis to quantify the symmetry and flatness of the data distribution. Generally speaking, the kurtosis value for Gaussian data is 3. Values larger than 3 indicate a highly non-Gaussian (i.e., super-Gaussian) distribution, while values smaller than 3 indicate a flatter (i.e., sub-Gaussian) distribution. Skewness quantifies the symmetry of the distribution in comparison to the sample mean. Symmetric distributions, like Gaussian, are characterized by the skewness of zero. Distributions that are skewed to the right are characterized by positive skewness, while distributions that are skewed to the left are characterized by negative skewness. As shown in Table 8.5, the training data considered in this study are non-Gaussian (i.e., kurtosis values deviated from 3 and the skewness values deviated from zero).

A heatmap of the correlation matrix of the training data is depicted in Fig. 8.32. As shown, the data from the majority of the training variables are weakly cross-correlated, except for the permeate variables, which show a moderate correlation.

As we explained previously, this WWTP data is multivariate, non-Gaussian, and very dynamic. Thus, developing a satisfactory analytical model is not easy. We chose a data-based method based on the DBN model and OCSVM detection algorithm for modeling and detecting anomalies in the WWTP data. One of the reasons we used this approach is the efficiency of the DBN model for learning and discovering relevant features in multivariate data. Once the DBN model is trained, it can be used to extract features from new WWTP data. These features serve as the inputs for the OCSVM algorithm, which can then reveal anomalies in the MBR system that may reduce system performance. The algorithm provides good separation of normal and abnormal features by constructing a hyperplane for discrimination during the training phase. We used the DBN-based kNN and k-means algorithms as benchmarks for comparison. Table 8.3 summarizes the selected parameters for the DBN-based methods.

https://t.me/learningnets

TABLE 8.5 Statistical features of the training data set.

	Mean	STD	Min	Q1	Median	3Q	Max	Skewness	Kurtosis
X1	13.029	0.629	10.722	12.654	12.802	13.515	14.031	0.049	2.953
X2	0.132	0.189	0.000	0.043	0.100	0.160	1.880	5.378	39.142
X3	0.092	0.051	0.038	0.058	0.069	0.107	0.248	1.523	4.212
X4	0.271	0.528	0	0.100	0.100	0.100	3.200	3.751	16.893
X5	7.315	0.060	7.097	7.300	7.300	7.300	7.500	0.324	4.009
X6	13362.718	2837.318	0	11414.500	12510	14966	19925	0.631	3.748
X7	831.34	33.320	779.5	808	824	852	898	0.444	2.211

https://t.me/learningnets

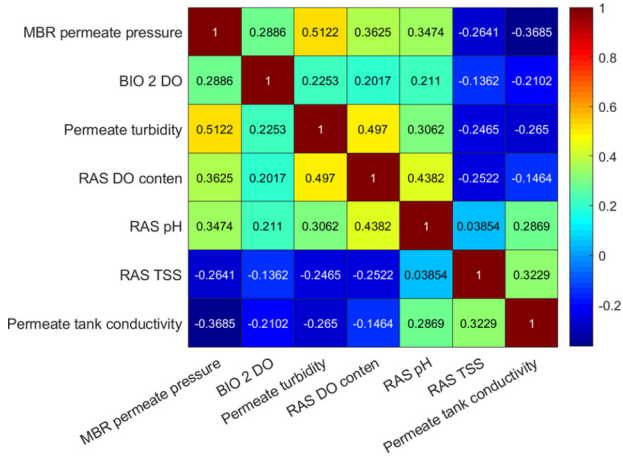


FIGURE 8.32 Heatmap of the correlation matrix of the training data.

As discussed above, the extraction of relevant features from the WWTP data was managed by the DBN model. We used the learned parameters from the training data to extract features from the testing data. Then, the OCSVM algorithm was used to separate normal and abnormal features using the parameters learned during the training phase. We also combined DBN with two common unsupervised clustering algorithms, kNN and k-means, for comparison purposes. The detection threshold in the DBN-kNN algorithm was fixed by applying an exponentially weighted moving average (EWMA) to the kNN distances. The EWMA was chosen for its ability to sense small deviations in time-series data [127]. Fig. 8.33A–C depicts the detection results of the three anomaly detection methods. All three methods were able to identify significant anomalies. As expected, the number of detections (i.e., alarms) was higher when using the OCSVM and kNN-EWMA algorithms compared to the k-means algorithm due to the flexibility and sensitivity of OCSVM and kNN-EWMA. The first alarm was triggered by the three schemes on April 18th, indicating that an abnormal event was taking place. This supports the outcomes of [93], in which the authors used all 28 variables of the process for monitoring WWTP using several data-based methods. They used other feature learning algorithms intended to reduce dimensionality, such as the static, dynamic, adaptive, and adaptive-dynamic versions of PCA, KPCA, and LLE. A T^2 chart based on parametric and nonparametric detection thresholds was used for detection. However, all the methods described in [93], both linear and nonlinear, have a shallow learning framework. These shallow methods triggered an alarm at 1:40 p.m. on April 21st and again on April 22nd [93]. The DBN-based methods detected an abnormal event earlier, on April 18th. The operator did not detect this anomaly until April 24th. Practically, detecting this anomaly at the time as an operator or earlier is crucial for avoiding the progression of the fault and serious degradation

https://t.me/learningnets

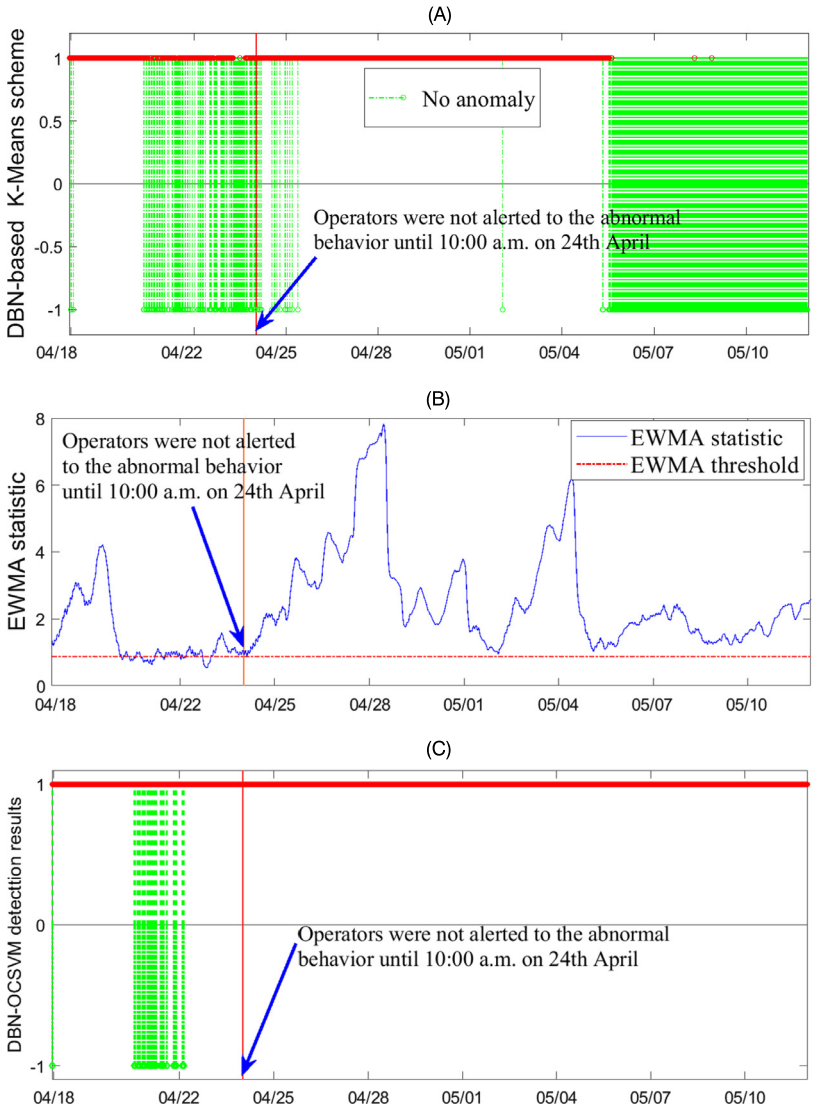


FIGURE 8.33 Monitoring results from the DBN-k-means approach (A), the DBN-kNN-EWMA approach (B), and the DBN-OCSVM approach (C) for a pH data set.

of the system. This fault caused the WWTP to shut down for two months for reparations. The integration of the greedy learning features of the DBN model and the flexibility of the OCSVM algorithm resulted in a remarkable capacity for fault detection in WWTP.

In summary, deep learning has recently received much attention from researchers because of its widespread application. In the previous sections, we

https://t.me/learningnets

presented an unsupervised deep learning approach for monitoring multivariate process variables. In this approach, the DBN model is coupled with the OCSVM algorithm to monitor the operation of a WWTP. The DBN model is used to learn complex features from the WWTP data, and the OCSVM algorithm is used to achieve good separation of normal and abnormal features. The DBN-OCSVM method showed a good detection ability when applied to real data from a decentralized WWTP in the USA.

8.4.4 Conclusion

In this chapter, we described successful applications of deep-learning-based methods for anomaly detection. In particular, we focused on utilizing the advantages of deep learning models to learn features and efficiently represent data in order to further improve anomaly detection. We showed that deep learning models, as opposed to shallow models, are required for efficient data representation. Three applications were used to demonstrate the benefits of deep learning models for enhancing anomaly detection performance. We merged shallow learning approaches with desirable properties, such as one-class SVM, kNN, and unsupervised deep learning, with more sophisticated and efficient monitoring techniques, applying the developed approaches to several processes: detecting obstacles in driving environments for autonomous vehicles; monitoring ozone pollution; and monitoring the operating conditions of wastewater treatment plants in Golden, CO, USA.

References

[1] A. Asvadi, C. Premebida, P. Peixoto, U. Nunes, 3D Lidar-based static and moving obstacle detection in driving environments: an approach based on voxels and multi-region ground planes, Robotics and Autonomous Systems 83 (2016) 299–311.
[2] C. Del, S. Skaar, A. Cardenas, L. Fehr, A sonar approach to obstacle detection for a vision-based autonomous wheelchair, Robotics and Autonomous Systems 54 (12) (2006) 967–981.
[3] P. Fleischmann, K. Berns, A stereo vision based obstacle detection system for agricultural applications, in: Field and Service Robotics, Springer, 2016, pp. 217–231.
[4] J. Leng, Y. Liu, D. Du, T. Zhang, P. Quan, Robust obstacle detection and recognition for driver assistance systems, IEEE Transactions on Intelligent Transportation Systems 21 (4) (2020) 1560–1571.
[5] M. Fathollahi, R. Kasturi, Autonomous driving challenge: to infer the property of a dynamic object based on its motion pattern, in: European Conference on Computer Vision, Springer, 2016, pp. 40–46.
[6] N. Fakhfakh, D. Gruyer, D. Aubert, Weighted V-disparity approach for obstacles localization in highway environments, in: Intelligent Vehicles Symposium (IV), 2013, IEEE, IEEE, 2013, pp. 1271–1278.
[7] H. Sun, H. Zou, S. Zhou, C. Wang, N. El-Sheimy, Surrounding moving obstacle detection for autonomous driving using stereo vision, International Journal of Advanced Robotic Systems 10 (6) (2013) 261.
[8] N. Appiah, N. Bandaru, Obstacle detection using stereo vision for self-driving cars, 2015.
[9] L. Matthies, R. Brockers, Y. Kuwata, S. Weiss, Stereo vision-based obstacle avoidance for micro air vehicles using disparity space, in: 2014 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2014, pp. 3242–3249.

https://t.me/learningnets

[10] X. Zhang, Y. Song, Y. Yang, H. Pan, Stereo vision based autonomous robot calibration, Robotics and Autonomous Systems 93 (2017) 43–51.
[11] J. Hernandez-Aceituno, R. Arnay, J. Toledo, L. Acosta, Using Kinect on an autonomous vehicle for outdoors obstacle detection, IEEE Sensors Journal 16 (10) (2016) 3603–3610.
[12] M.K. Habib, Fiber-grating-based vision system for real-time tracking, monitoring, and obstacle detection, IEEE Sensors Journal 7 (1) (2006) 105–121.
[13] Y.N.K. Yamaguchi, T. Kato, Moving obstacle detection using monocular vision, in: Intelligent Vehicles Symposium, IEEE, 2006, pp. 288–293.
[14] D. Pfeiffer, U. Franke, Efficient representation of traffic scenes by means of dynamic stixels, in: 2010 IEEE Intelligent Vehicles Symposium, IEEE, 2010, pp. 217–224.
[15] N. Morales, A. Morell, J. Toledo, L. Acosta, Fast object motion estimation based on dynamic stixels, Sensors 16 (8) (2016) 1182.
[16] I. Nadav, E. Katz, Off-road path and obstacle detection using monocular camera, in: 2016 IEEE International Conference on the Science of Electrical Engineering (ICSEE), IEEE, 2016, pp. 1–5.
[17] J. Woo, N. Kim, Vision based obstacle detection and collision risk estimation of an unmanned surface vehicle, in: 2016 13th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI), IEEE, 2016, pp. 461–465.
[18] A. Burlacu, S. Bostaca, I. Hector, P. Herghelegiu, G. Ivanica, A. Moldoveanul, S. Caraiman, Obstacle detection in stereo sequences using multiple representations of the disparity map, in: 2016 20th International Conference on System Theory, Control and Computing (ICSTCC), IEEE, 2016, pp. 854–859.
[19] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: Computer Vision and Pattern Recognition, 2005, IEEE, 2005, pp. 886–893.
[20] C. Demonceaux, A. Potelle, D. Kachi-Akkouche, Obstacle detection in a road scene based on motion analysis, IEEE Transactions on Vehicular Technology 53 (6) (2004) 1649–1656.
[21] F. Oniga, S. Nedevschi, Processing dense stereo data using elevation maps: road surface, traffic isle, and obstacle detection, IEEE Transactions on Vehicular Technology 59 (3) (2009) 1172–1182.
[22] L. Nalpantidis, D. Kragic, I. Kostavelis, A. Gasteratos, Theta-disparity: an efficient representation of the 3D scene structure, in: Intelligent Autonomous Systems, vol. 13, Springer, 2016, pp. 795–806.
[23] A. Dairi, F. Harrou, M. Senouci, Y. Sun, Unsupervised obstacle detection in driving environments using deep-learning-based stereovision, Robotics and Autonomous Systems 100 (2018) 287–301.
[24] G. Qi, H. Wang, M. Haner, C. Weng, S. Chen, Z. Zhu, Convolutional neural network based detection and judgement of environmental obstacle in vehicle operation, CAAI Transactions on Intelligence Technology 4 (2) (2019) 80–91.
[25] A. Dairi, F. Harrou, Y. Sun, M. Senouci, Obstacle detection for intelligent transportation systems using deep stacked autoencoder and *k*-nearest neighbor scheme, IEEE Sensors Journal 18 (12) (2018) 5122–5132.
[26] M. Cornacchia, B. Kakillioglu, Y. Zheng, S. Velipasalar, Deep learning-based obstacle detection and classification with portable uncalibrated patterned light, IEEE Sensors Journal 18 (20) (2018) 8416–8425.
[27] S. Ramos, S. Gehrig, P. Pinggera, U. Franke, C. Rother, Detecting unexpected obstacles for self-driving cars: fusing deep learning and geometric modeling, in: 2017 IEEE Intelligent Vehicles Symposium (IV), IEEE, 2017, pp. 1025–1032.
[28] D. Petković, A.S. Danesh, M. Dadkhah, N. Misaghian, S. Shamshirband, E. Zalnezhad, N.D. Pavlović, Adaptive control algorithm of flexible robotic gripper by extreme learning machine, Robotics and Computer-Integrated Manufacturing 37 (2016) 170–178.
[29] P. Dollar, C. Wojek, B. Schiele, P. Perona, Pedestrian detection: an evaluation of the state of the art, IEEE Transactions on Pattern Analysis and Machine Intelligence 34 (4) (2011) 743–761.

https://t.me/learningnets

[30] G. Hinton, Learning multiple layers of representation, Trends in Cognitive Sciences 11 (10) (2007) 428–434.
[31] Y. Bengio, Y. LeCun, et al., Scaling learning algorithms towards AI, in: Large Scale Kernel Machines, vol. 34(5), 2007, pp. 1–41.
[32] V.D. Nguyen, H. Van Nguyen, D.T. Tran, S.J. Lee, J.W. Jeon, Learning framework for robust obstacle detection, recognition, and tracking, IEEE Transactions on Intelligent Transportation Systems 18 (6) (2016) 1633–1646.
[33] Y. Bengio, et al., Learning deep architectures for AI, Foundations and Trends® in Machine Learning 2 (1) (2009) 1–127.
[34] R. Labayrade, D. Aubert, In-vehicle obstacles detection and characterization by stereovision, in: Proceedings of the 1st International Workshop on In-Vehicle Cognitive Computer Vision Systems, Graz, Austria, 2003.
[35] C. Georgoulas, L. Kotoulas, G.C. Sirakoulis, I. Andreadis, A. Gasteratos, Real-time disparity map computation module, Microprocessors and Microsystems 32 (3) (2008) 159–170.
[36] R.A. Hamzah, H. Ibrahim, Literature survey on stereo vision disparity map algorithms, Journal of Sensors 2016 (2016).
[37] S.H. Lee, S. Sharma, Real-time disparity estimation algorithm for stereo camera systems, IEEE Transactions on Consumer Electronics 57 (3) (2011) 1018–1026.
[38] B.J. Tippetts, D.-J. Lee, J.K. Archibald, K.D. Lillywhite, Dense disparity real-time stereo vision algorithm for resource-limited systems, IEEE Transactions on Circuits and Systems for Video Technology 21 (10) (2011) 1547–1555.
[39] N. Soquet, M. Perrollaz, R. Labayrade, D. Aubert, Free space estimation for autonomous navigation, in: International Conference on Computer Vision Systems: Proceedings, 2007.
[40] Z. Hu, K. Uchimura, U-V-disparity: an efficient algorithm for stereovision based scene analysis, in: Intelligent Vehicles Symposium, Proceedings, 2005, IEEE, IEEE, 2005, pp. 48–54.
[41] N. Fakhfakh, D. Gruyer, D. Aubert, Weighted V-disparity approach for obstacles localization in highway environments, in: 2013 IEEE Intelligent Vehicles Symposium (IV), IEEE, 2013, pp. 1271–1278.
[42] R. Labayrade, D. Aubert, J.-P. Tarel, Real time obstacle detection in stereovision on non flat road geometry through 'V-disparity' representation, in: Intelligent Vehicle Symposium, 2002, IEEE, vol. 2, IEEE, 2002, pp. 646–651.
[43] P. Vincent, H. Larochelle, Y. Bengio, P.-A. Manzagol, Extracting and composing robust features with denoising autoencoders, in: Proceedings of the 25th International Conference on Machine Learning, ACM, 2008, pp. 1096–1103.
[44] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.-A. Manzagol, Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion, Journal of Machine Learning Research 11 (2010) 3371–3408.
[45] H.-C. Shin, M.R. Orton, D.J. Collins, S.J. Doran, M.O. Leach, Stacked autoencoders for unsupervised feature learning and multiple organ detection in a pilot study using 4D patient data, IEEE Transactions on Pattern Analysis and Machine Intelligence 35 (8) (2013) 1930–1943.
[46] N. Appiah, N. Bandaru, Obstacle detection using stereo vision for self-driving cars, in: IEEE Intelligent Vehicles Symposium (IV), 2011, pp. 926–932.
[47] H. Yoo, J. Son, B. Ham, K. Sohn, Real-time rear obstacle detection using reliable disparity for driver assistance, Expert Systems with Applications 56 (2016) 186–196.
[48] J.-L. Blanco, F.-A. Moreno, J. González-Jiménez, The Málaga urban dataset: high-rate stereo and lidars in a realistic urban scenario, The International Journal of Robotics Research 33 (2) (2014) 207–214. [Online]. Available: http://www.mrpt.org/MalagaUrbanDataset.
[49] T. Scharwächter, M. Enzweiler, U. Franke, S. Roth, Stixmantics: a medium-level model for real-time semantic scene understanding, in: European Conference on Computer Vision, Springer, 2014, pp. 533–548.
[50] M.P. Rissanen, T. Kurtén, M. Sipila, J.A. Thornton, J. Kangasluoma, N. Sarnela, H. Junninen, S. Jørgensen, S. Schallhart, M.K. Kajos, et al., The formation of highly oxidized multifunctional products in the ozonolysis of cyclohexene, Journal of the American Chemical Society 136 (44) (2014) 15596–15606.

https://t.me/learningnets

[51] A. Nawahda, An assessment of adding value of traffic information and other attributes as part of its classifiers in a data mining tool set for predicting surface ozone levels, Process Safety and Environmental Protection 99 (2016) 149–158.

[52] S.A. Abdul-Wahab, C.S. Bakheit, S.M. Al-Alawi, Principal component and multiple regression analysis in modelling of ground-level ozone and factors affecting its concentrations, Environmental Modelling & Software 20 (10) (2005) 1263–1271.

[53] C. Vlachokostas, S. Nastis, C. Achillas, K. Kalogeropoulos, I. Karmiris, N. Moussiopoulos, E. Chourdakis, G. Banias, N. Limperi, Economic damages of ozone air pollution to crops using combined air quality and GIS modelling, Atmospheric Environment 44 (28) (2010) 3352–3361.

[54] C. Dueñas, M. Fernández, S. Canete, J. Carretero, E. Liger, Analyses of ozone in urban and rural sites in Málaga (Spain), Chemosphere 56 (6) (2004) 631–639.

[55] C. Xing, C. Liu, S. Wang, K.L. Chan, Y. Gao, X. Huang, W. Su, C. Zhang, Y. Dong, G. Fan, et al., Observations of the vertical distributions of summertime atmospheric pollutants and the corresponding ozone production in Shanghai, China, Atmospheric Chemistry and Physics 17 (23) (2017).

[56] F. Harrou, F. Kadri, S. Khadraoui, Y. Sun, Ozone measurements monitoring using data-based approach, Process Safety and Environmental Protection 100 (2016) 220–231.

[57] N. Castell, F.R. Dauge, P. Schneider, M. Vogt, U. Lerner, B. Fishbain, D. Broday, A. Bartonova, Can commercial low-cost sensor platforms contribute to air quality monitoring and exposure estimates?, Environment International 99 (2017) 293–302.

[58] F. Harrou, M. Nounou, H. Nounou, Statistical detection of abnormal ozone levels using principal component analysis, International Journal of Engineering and Technology 12 (6) (2012) 54–59.

[59] F. Harrou, L. Fillatre, M. Bobbia, I. Nikiforov, Statistical detection of abnormal ozone measurements based on constrained generalized likelihood ratio test, in: Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on, IEEE, 2013, pp. 4997–5002.

[60] S. Bordignon, M. Scagliarini, Monitoring algorithms for detecting changes in the ozone concentrations, Environmetrics, The Official Journal of the International Environmetrics Society 11 (2) (2000) 125–137.

[61] G. Miskell, J. Salmond, M. Alavi-Shoshtari, M. Bart, B. Ainslie, S. Grange, I.G. McKendry, G.S. Henshaw, D.E. Williams, Data verification tools for minimizing management costs of dense air-quality monitoring networks, Environmental Science & Technology 50 (2) (2015) 835–846.

[62] M. Alavi-Shoshtari, J.A. Salmond, C.D. Giurcăneanu, G. Miskell, L. Weissert, D.E. Williams, Automated data scanning for dense networks of low-cost air quality instruments: detection and differentiation of instrumental error and local to regional scale environmental abnormalities, Environmental Modelling & Software 101 (2018) 34–50.

[63] M. Alavi-Shoshtari, D.E. Williams, J.A. Salmond, J.P. Kaipio, Detection of malfunctions in sensor networks, EnvironMetrics 24 (4) (2013) 227–236.

[64] J.H. Seinfeld, S.N. Pandis, Atmospheric Chemistry and Physics: from Air Pollution to Climate Change, John Wiley & Sons, 2016.

[65] B. Özbay, G.A. Keskin, Ş.Ç. Doğruparmak, S. Ayberk, Multivariate methods for ground-level ozone modeling, Atmospheric Research 102 (1–2) (2011) 57–65.

[66] K. Moustris, P. Nastos, I. Larissi, A. Paliatsos, Application of multiple linear regression models and artificial neural networks on the surface ozone forecast in the greater Athens area, Greece, Advances in Meteorology 2012 (2012).

[67] N.R. Awang, N.A. Ramli, A.S. Yahaya, M. Elbayoumi, Multivariate methods to predict ground level ozone during daytime, nighttime, and critical conversion time in urban areas, Atmospheric Pollution Research 6 (5) (2015) 726–734.

[68] A. Lengyel, K. Héberger, L. Paksy, O. Bánhidi, R. Rajkó, Prediction of ozone concentration in ambient air using multivariate methods, Chemosphere 57 (8) (2004) 889–896.

https://t.me/learningnets

[69] E. Ortiz-García, S. Salcedo-Sanz, Á. Pérez-Bellido, J. Portilla-Figueras, L. Prieto, Prediction of hourly O_3 concentrations using support vector regression algorithms, Atmospheric Environment 44 (35) (2010) 4481–4488.
[70] P. Hájek, V. Olej, Ozone prediction on the basis of neural networks, support vector regression and methods with uncertainty, Ecological Informatics 12 (2012) 31–42.
[71] R. Mintz, B.R. Young, W.Y. Svrcek, Fuzzy logic modeling of surface ozone concentrations, Computers & Chemical Engineering 29 (10) (2005) 2049–2059.
[72] W.R. Burrows, M. Benjamin, S. Beauchamp, E.R. Lord, D. McCollor, B. Thomson, CART decision-tree statistical analysis and prediction of summer season maximum surface ozone for the Vancouver, Montreal, and Atlantic regions of Canada, Journal of Applied Meteorology 34 (8) (1995) 1848–1862.
[73] Y. Bengio, O. Delalleau, On the expressive power of deep architectures, in: International Conference on Algorithmic Learning Theory, Springer, 2011, pp. 18–36.
[74] N. Cohen, O. Sharir, A. Shashua, On the expressive power of deep learning: a tensor analysis, in: Conference on Learning Theory, 2016, pp. 698–728.
[75] Y. Bai, Y. Li, B. Zeng, C. Li, J. Zhang, Hourly PM2.5 concentration forecast using stacked autoencoder model with emphasis on seasonality, Journal of Cleaner Production 224 (2019) 739–750.
[76] A. Koesdwiady, R. Soua, F. Karray, Improving traffic flow prediction with weather information in connected cars: a deep learning approach, IEEE Transactions on Vehicular Technology 65 (12) (2016) 9508–9517.
[77] Y. Jia, J. Wu, Y. Du, Traffic speed prediction using deep learning method, in: Intelligent Transportation Systems (ITSC), 2016 IEEE 19th International Conference on, IEEE, 2016, pp. 1217–1222.
[78] D. Ravì, C. Wong, F. Deligianni, M. Berthelot, J. Andreu-Perez, B. Lo, G.-Z. Yang, Deep learning for health informatics, IEEE Journal of Biomedical and Health Informatics 21 (1) (2017) 4–21.
[79] J. Wang, X. Zhang, Q. Gao, H. Yue, H. Wang, Device-free wireless localization and activity recognition: a deep learning approach, IEEE Transactions on Vehicular Technology 66 (7) (2017) 6258–6267.
[80] Y.-D. Zhang, Y. Zhang, X.-X. Hou, H. Chen, S.-H. Wang, Seven-layer deep neural network based on sparse autoencoder for voxelwise detection of cerebral microbleed, Multimedia Tools and Applications 77 (9) (2018) 10521–10538.
[81] F. Harrou, A. Dairi, Y. Sun, F. Kadri, Detecting abnormal ozone measurements with a deep learning-based strategy, IEEE Sensors Journal 18 (17) (2018) 7222–7232.
[82] Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle, Greedy layer-wise training of deep networks, in: Advances in Neural Information Processing Systems, 2007, pp. 153–160.
[83] G. Hinton, R. Salakhutdinov, Reducing the dimensionality of data with neural networks, Science 313 (5786) (2006) 504–507.
[84] G.E. Hinton, S. Osindero, Y.-W. Teh, A fast learning algorithm for deep belief nets, Neural Computation 18 (7) (2006) 1527–1554.
[85] A.P. Dempster, N.M. Laird, D.B. Rubin, Maximum likelihood from incomplete data via the EM algorithm, Journal of the Royal Statistical Society, Series B, Methodological (1977) 1–38.
[86] T. Zhang, R. Ramakrishnan, M. Livny, BIRCH: an efficient data clustering method for very large databases, in: ACM Sigmod Record, vol. 25(2), ACM, 1996, pp. 103–114.
[87] D. Arthur, S. Vassilvitskii, k-means++: the advantages of careful seeding, in: Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035.
[88] S.B. Grant, J.-D. Saphores, D.L. Feldman, A.J. Hamilton, T.D. Fletcher, P.L. Cook, M. Stewardson, B.F. Sanders, L.A. Levin, R.F. Ambrose, et al., Taking the 'waste' out of 'wastewater' for human water security and ecosystem sustainability, Science 337 (6095) (2012) 681–686.
[89] L. Castellet, M. Molinos-Senante, Efficiency assessment of wastewater treatment plants: a data envelopment analysis approach integrating technical, economic, and environmental issues, Journal of Environmental Management 167 (2016) 160–166.

https://t.me/learningnets

[90] S. Dolnicar, A.I. Schäfer, Desalinated versus recycled water: public perceptions and profiles of the accepters, Journal of Environmental Management 90 (2) (2009) 888–900.
[91] P. Côté, S. Siverns, S. Monti, Comparison of membrane-based solutions for water reclamation and desalination, Desalination 182 (1–3) (2005) 251–257.
[92] I. Boujelben, Y. Samet, M. Messaoud, M.B. Makhlouf, S. Maalej, Descriptive and multivariate analyses of four Tunisian wastewater treatment plants: a comparison between different treatment processes and their efficiency improvement, Journal of Environmental Management 187 (2017) 63–70.
[93] K. Kazor, R.W. Holloway, T.Y. Cath, A.S. Hering, Comparison of linear and nonlinear dimension reduction techniques for automated process monitoring of a decentralized wastewater treatment facility, Stochastic Environmental Research and Risk Assessment 30 (5) (2016) 1527–1544.
[94] M. Henze, Activated sludge models ASM1, ASM2, ASM2d and ASM3, IWA Scientific and Tech. Rep., 2000.
[95] G. Mannina, D. DiTrapani, G. Viviani, H. Ødegaard, Modelling and dynamic simulation of hybrid moving bed biofilm reactors: model concepts and application to a pilot plant, Biochemical Engineering Journal 56 (1) (2011) 23–36.
[96] M. Plattes, E. Henry, P. Schosseler, A. Weidenhaupt, Modelling and dynamic simulation of a moving bed bioreactor for the treatment of municipal wastewater, Biochemical Engineering Journal 32 (2) (2006) 61–68.
[97] D. Dochain, P. Vanrolleghem, Dynamical Modelling and Estimation in Wastewater Treatment Processes, IWA Publishing, 2001.
[98] P. Vanrolleghem, H. Spanjers, B. Petersen, P. Ginestet, I. Takacs, Estimating (combinations of) activated sludge model No. 1 parameters and components by respirometry, Water Science and Technology 39 (1) (1999) 195–214.
[99] I. Skrjanc, L. Teslic, Monitoring of waste-water treatment plant using Takagi–Sugeno fuzzy model, in: The 14th IEEE Mediterranean Electrotechnical Conference, MELECON 2008, IEEE, 2008, pp. 67–70.
[100] J.-M. Lee, C.-K. Yoo, I.-B. Lee, New monitoring technique with an ICA algorithm in the wastewater treatment process, Water Science and Technology 47 (12) (2003) 49–56.
[101] J.-P. Steyer, D. Rolland, J.-C. Bouvier, R. Moletta, Hybrid fuzzy neural network for diagnosis-application to the anaerobic treatment of wine distillery wastewater in a fluidized bed reactor, Water Science and Technology 36 (6–7) (1997) 209–217.
[102] X. Wang, H. Ratnaweera, J. Holm, V. Olsbu, Statistical monitoring and dynamic simulation of a wastewater treatment plant: a combined approach to achieve model predictive control, Journal of Environmental Management 193 (2017) 1–7.
[103] A. Dias, M. Alves, E. Ferreira, Application of computational intelligence techniques for monitoring and prediction of biological wastewater treatment systems, in: Proceedings of the Int. IWA Conf. on Automation in Water Quality Monitoring, vol. 3, Gent, Belgium, Springer, 2007, pp. 1–8.
[104] T. Cheng, F. Harrou, Y. Sun, T. Leiknes, Monitoring influent measurements at water resource recovery facility using data-driven soft sensor approach, IEEE Sensors Journal 19 (1) (2018) 342–352.
[105] B. Mali, S. Laskar, PLS-based multivariate statistical approach for soft sensor development in WWTP, in: Control Instrumentation Systems, Springer, 2020, pp. 123–131.
[106] I. González, A. Serrano, J. García-Olmo, M.C. Gutiérrez, A.F. Chica, M.Á. Martín, Assessment of the treatment, production and characteristics of WWTP sludge in Andalusia by multivariate analysis, Process Safety and Environmental Protection 109 (2017) 609–620.
[107] Y. Liu, Y. Pan, Z. Sun, D. Huang, Statistical monitoring of wastewater treatment plants using variational Bayesian PCA, Industrial & Engineering Chemistry Research 53 (8) (2014) 3272–3282.
[108] M. Huang, Y. Ma, J. Wan, H. Zhang, Y. Wang, Modeling a paper-making wastewater treatment process by means of an adaptive network-based fuzzy inference system and principal component analysis, Industrial & Engineering Chemistry Research 51 (17) (2012) 6166–6174.

https://t.me/learningnets

[109] K. Villez, M. Ruiz, G. Sin, J. Colomer, C. Rosen, P. Vanrolleghem, Combining multiway principal component analysis (MPCA) and clustering for efficient data mining of historical data sets of SBR processes, Water Science and Technology 57 (10) (2008) 1659–1666.
[110] J.-M. Lee, C. Yoo, S. Choi, P. Vanrolleghem, I.-B. Lee, Nonlinear process monitoring using kernel principal component analysis, Chemical Engineering Science 59 (1) (2004) 223–234.
[111] C. Rosén, J. Lennox, Multivariate and multiscale monitoring of wastewater treatment operation, Water Research 35 (14) (2001) 3402–3410.
[112] D. Lee, P. Vanrolleghem, Monitoring of a sequencing batch reactor using adaptive multiblock principal component analysis, Biotechnology and Bioengineering 82 (4) (2003) 489–497.
[113] S.-P. Mujunen, P. Minkkinen, P. Teppola, R.-S. Wirkkala, Modeling of activated sludge plants treatment efficiency with PLSR: a process analytical case study, Chemometrics and Intelligent Laboratory Systems 41 (1) (1998) 83–94.
[114] H. Cheng, Y. Liu, D. Huang, B. Liu, Optimized forecast components-SVM-based fault diagnosis with applications for wastewater treatment, IEEE Access 7 (2019) 128534–128543.
[115] M. Miron, L. Frangu, S. Caraman, L. Luca, Artificial neural network approach for fault recognition in a wastewater treatment process, in: 2018 22nd International Conference on System Theory, Control and Computing (ICSTCC), IEEE, 2018, pp. 634–639.
[116] S. Wilcox, D. Hawkes, F. Hawkes, A. Guwy, A neural network, based on bicarbonate monitoring, to control anaerobic digestion, Water Research 29 (6) (1995) 1465–1470.
[117] J.-J. Zhu, L. Kang, P.R. Anderson, Predicting influent biochemical oxygen demand: balancing energy demand and risk management, Water Research 128 (2018) 304–313.
[118] T. Cheng, A. Dairi, F. Harrou, Y. Sun, T. Leiknes, Monitoring influent conditions of wastewater treatment plants by nonlinear data-based techniques, IEEE Access 7 (2019) 108827–108837.
[119] A. Dairi, T. Cheng, F. Harrou, Y. Sun, T. Leiknes, Deep learning approach for sustainable WWTP operation: a case study on data-driven influent conditions monitoring, Sustainable Cities and Society 50 (2019) 101670.
[120] S. Park, S.-S. Baek, J. Pyo, Y. Pachepsky, J. Park, K.H. Cho, Deep neural networks for modeling fouling growth and flux decline during NF/RO membrane filtration, Journal of Membrane Science 587 (2019) 117164.
[121] Z. Wang, Y. Man, Y. Hu, J. Li, M. Hong, P. Cui, A deep learning based dynamic COD prediction model for urban sewage, Environmental Science: Water Research & Technology 5 (12) (2019) 2210–2218.
[122] F. Harrou, A. Dairi, Y. Sun, M. Senouci, Statistical monitoring of a wastewater treatment plant: a case study, Journal of Environmental Management 223 (2018) 807–814.
[123] N.V. Bhattacharjee, E.W. Tollner, Improving management of windrow composting systems by modeling runoff water quality dynamics using recurrent neural network, Ecological Modelling 339 (2016) 68–76.
[124] A. Prieto, D. Vuono, R. Holloway, J. Benecke, J. Henkel, T. Cath, T. Reid, L. Johnson, J. Drewes, Decentralized wastewater treatment for distributed water reclamation and reuse: the good, the bad, and the ugly—experience from a case study, in: Novel Solutions to Water Pollution, ACS Publications, 2013, pp. 251–266.
[125] R.L. Siegrist, Decentralized Water Reclamation Engineering. A Curriculum Workbook, Springer International Publishing AG, Charm, 2017.
[126] D. Vuono, J. Henkel, J. Benecke, T. Cath, T. Reid, L. Johnson, J. Drewes, Flexible hybrid membrane treatment systems for tailored nutrient management: a new paradigm in urban wastewater treatment, Journal of Membrane Science 446 (2013) 34–41.
[127] A. Zeroual, F. Harrou, Y. Sun, N. Messai, Monitoring road traffic congestion using a macroscopic traffic model and a statistical monitoring scheme, Sustainable Cities and Society 35 (2017) 494–510.

https://t.me/learningnets

Chapter 9

Conclusion and further research directions

Developing efficient anomaly detection and isolation schemes that offer early detection of potential anomalies in the monitored process and identify and isolate the source of the detected anomalies is indispensable to monitor process operations in an efficient manner. This will further enhance availability, operation reliability, and profitability of monitored processes and reduce manpower costs. This book is mainly devoted to data-driven fault detection and isolation methods based on multivariate statistical monitoring techniques and deep learning methods.

The focus of this book is to offer a recent overview of anomaly detection and isolation methods, and it provides some new methods for the process monitoring purposes. Specifically, in the first part of this book, the objective was to tackle multivariate challenges in-process monitoring by merging the advantages of univariate and traditional multivariate techniques to enhance their performance and widen their practical applicability. Univariate monitoring schemes, such as EWMA (exponentially-weighted moving average) and CUSUM (cumulative sum) control charts, are widely used univariate control charts. The key ingredient to apply such tools to multivariate data is to apply appropriate multivariate dimension reduction techniques (e.g., partial least squares (PLS) and canonical correlation analysis (CCA)), according to the features of a process, and use control charts to monitor more informative variables in a lower dimension. Particularly, we presented methods, when relationships among variables are linear by developing latent variable regression (LVR)-based univariate monitoring techniques (EWMA, generalized likelihood ratio (GLR) test, and CUSUM), especially for detecting small faults in highly correlated multivariate data. Most commonly used monitoring techniques detect the anomaly as a shift in the means or variances of the process. In many real processes, the presence of an anomaly may manifest itself by a change in the process distribution (quantiles/extremes) rather than an additive bias in the means or variances. Thus, distribution-based process monitoring schemes (e.g., Kullback–Leibler, Hellinger metrics) were briefly presented to improve the monitoring performance of LVR-based approaches. To handle processes nonlinearity, we used a nonlinear LVR modeling approach, which is a powerful tool for processing nonlinearities. In this direction, nonlinear functions using an adaptive network-based fuzzy inference system are used as the inner relation of the LVR model

https://doi.org/10.1016/B978-0-12-819366-5.00015-2
Copyright © 2021 Elsevier Inc. All rights reserved.
https://t.me/learningnets

(i.e., mapping nonlinear relation between latent variable and output). Also, the nonlinear process monitoring using kernel PCA has been presented. Nonlinear LVR-based univariate (EWMA, CUSUM, GLR, Hellinger distance (HD), and Kullback–Leibler distance (KLD)) and multivariate (multivariate CUSUM (MCUSUM) and multivariate EWMA (MEWMA)) monitoring approaches with parametric and nonparametric thresholds have been developed to further improve process monitoring and the profitability of the developed mechanisms in practice. However, data observed from environmental and engineered processes are usually noisy and correlated in time, which makes fault detection (FD) more difficult as the presence of noise degrades FD quality, and most methods are developed for independent observations. Thereby, wavelet-based multiscale representation of a process are used to decorrelate data and reduce noise and then combine with LVR-based EWMA and CUSUM for both linear and nonlinear processes. In addition, we have presented fault isolation, which is an important component in-process monitoring, to identify variables that have been affected by the occurred fault. In this book, both traditional and modern fault isolation methods have been reviewed. We also briefly reviewed some metrics that may be used to verify the efficiency of fault isolation approaches along with two case studies for illustration.

In the second part of this book, we merged the desirable properties of shallow learning approaches, such as a one-class support vector machine and k-nearest neighbors and unsupervised deep learning approaches to develop more sophisticated and efficient monitoring techniques. First, we provide an overview of some shallow machine learning approaches used in anomaly detection and outlier detection in data mining, namely data clustering techniques. Afterwards, we introduced deep learning-based approaches to handle dependence in time series data. The efficiency of the RNN-RBM approach relies on the combination of a powerful type of deep learning model designed to handle dependence in time series data, namely RNN, with the well-known RBM model. Here, RNN plays the role of recurrent features extractor that learns from a high-dimensional sequence of complex temporal dependencies. It also relies on the detection capability of binary clustering schemes. Specifically, the output of the mixed RNN-RBM model is fed to the clustering algorithms for anomaly detection. Thereafter, we presented different energy-based deep learning models and stacked autoencoders. Furthermore, many monitoring approaches based on deep probabilistic models such as DBN and deep autoencoder have been presented. Later, we applied some of the presented approaches to monitor many processes, such as wastewater treatment plants at KAUST and Golden, CO, USA, detection of obstacles in driving environments for autonomous robots and vehicles, and ozone pollution.

As discussed above, there is no doubt that machine learning and deep learning have obtained a significant position in the existing state-of-the-art in-process monitoring field. As demonstrated in the previous chapters, the greater learning ability and the flexibility to approximate nonlinear functions make deep learning

https://t.me/learningnets

models a promising tool for modeling and monitoring multivariable linear and nonlinear processes. However, the work presented in this book raises a number of questions and provides some directions for future work. In particular, the following topics merit consideration from researchers:

- Widespread, successful applications of deep learning provide evidence that the depth of deep learning methods provides significant advantages for monitoring schemes. However, there are many problems with deep-learning-based monitoring frameworks. For instance, applications of deep learning have progressed faster than theory, which leads to some gaps and questions that must be addressed. For example, what is the optimal number of layers and hyperparameters for deep structures? Much future research is required to furnish practical and theoretical guidelines that address such questions. Moreover, due to the rapid advancement of computational technologies, deep learning could involve various kinds of learning, such as unsupervised, transfer, and reinforcement learning. It would be interesting to research a way to combine different types of learning to design a unified learning framework, as this may improve the feature learning capability of deep learning and, in turn, improve anomaly detection performance.
- Data-based methods using deep learning have shown to be effective for large and complex monitoring processes. However, in various practical processes (e.g., environmental processes, biology, and hydrology), data are functional in nature. For example, dust measurements for monitoring air quality can be viewed as a function of time. It has been shown that PCA for multivariate observations is not suitable for functional data [1–4]. For these data, functional PCA (FPCA) captures the most variation based on the first few orthogonal functional principal components [4–7]. Although FPCA is a popular statistical method for functional data analysis, it has not been used for process monitoring or fault detection and diagnosis. There are some existing monitoring charts that can handle functional data, but in order to design efficient methods to detect anomalies in functional data, much research is still required. For instance, in most existing charts for statistical monitoring of functional data, observations are assumed to be uncorrelated. However, they may be naturally autocorrelated over time or space (which is taken into consideration in conventional monitoring methods). Therefore, time-dependent information should be integrated when designing monitoring methods for functional data.
- Moreover, existing deep learning methods should apply a functional data structure for anomaly detection. To handle this issue, functional data analysis may be helpful as it can be integrated with deep learning to develop innovative methods for process monitoring. Future research should aim to develop deep-learning-based monitoring methods that integrate functional data.
- In this chapter, we proposed a deep-learning-based monitoring approach for detecting abnormal ozone measurements at the regional scale. We did not consider the spatiotemporal correlation in the air quality monitoring network.

https://t.me/learningnets

When monitoring approaches are used to monitor spatial data, the spatial data structure should be well accommodated in the monitoring method. There is a need to develop a more flexible deep learning model that considers spatiotemporal correlations, as such a model will be more accurate, able to capture spatiotemporal features, and exhibit improved anomaly detection performance. It is also important for researchers to extend existing deep-learning-based monitoring techniques to account for the spatiotemporal evolution of data (i.e., by including information from spatial lags in monitoring) and to utilize these improvements in various applications.

- Traditionally, deep learning and machine learning models are designed to achieve suitable results under the condition that the training and testing datasets are drawn the same distribution. On the assumption that the distribution of data changes, then a new model needs to be designed. However, designing a new model every time gathering new training data is time-consuming. To alleviate this challenge, transfer learning can be used for reducing the requirement to gather a large amount of training data. Generally speaking, the essence of transfer learning is to store knowledge obtained when solving one problem and apply it to another related problem. For instance, a deep learning model used for one task could be applied for a different task or another domain. Recently, the reinforcement learning concept has been widely applied in many disciplines including game theory, autonomous vehicles, and swarm intelligence [8–10]. As future work, it is of interest to develop other sophisticated technologies for process monitoring based on transfer learning and extreme learning.
- Few statisticians are engaged in this stream of research, but there is a need to understand and improve deep learning models based on the promising combination of a deep learning framework and statistical methods.

References

[1] N. Das, Non-parametric control chart for controlling variability based on rank test, Economic Quality Control 23 (2) (2008) 227–242.
[2] F. Ferraty, P. Vieu, Nonparametric Functional Data Analysis: Theory and Practice, Springer Science & Business Media, 2006.
[3] J.O. Ramsay, Functional data analysis, in: Encyclopedia of Statistical Sciences, vol. 4, 2004.
[4] H.L. Shang, A survey of functional principal component analysis, AStA Advances in Statistical Analysis 98 (2) (2014) 121–142.
[5] J.R. Berrendero, A. Justel, M. Svarc, Principal components for multivariate functional data, Computational Statistics & Data Analysis 55 (9) (2011) 2619–2634.
[6] Y. Sun, M.G. Genton, D.W. Nychka, Exact fast computation of band depth for large functional datasets: how quickly can one million curves be ranked?, Stat - The ISI's Journal for the Rapid Dissemination of Statistics Research 1 (1) (2012) 68–74.
[7] F. Yao, H.-G. Müller, J.-L. Wang, Functional data analysis for sparse longitudinal data, Journal of the American Statistical Association 100 (470) (2005) 577–590.
[8] H.-W. Ng, V.D. Nguyen, V. Vonikakis, S. Winkler, Deep learning for emotion recognition on small datasets using transfer learning, in: Proceedings of the 2015 ACM on International Conference on Multimodal Interaction, 2015, pp. 443–449.

https://t.me/learningnets

[9] S. Khan, N. Islam, Z. Jan, LU. Din, J.J.C. Rodrigues, A novel deep learning based framework for the detection and classification of breast cancer using transfer learning, Pattern Recognition Letters 125 (2019) 1–6.
[10] M. Long, H. Zhu, J. Wang, M.I. Jordan, Deep transfer learning with joint adaptation networks, in: Proceedings of the 34th International Conference on Machine Learning, vol. 70, JMLR.org, 2017, pp. 2208–2217.

https://t.me/learningnets

Index

A

Abnormal
  event, 4, 7, 55, 56, 60, 61, 241, 242, 247
  features, 62, 204, 250, 265, 271, 286, 290, 293, 295, 297
Abrupt
  anomaly, 3, 147
  faults, 147, 284, 286
Actuator faults, 2
Adaptive EWMA, 36
Adaptive monitoring methods, 10
Advanced process monitoring, 109
Anaerobic WWTP, 289
Anomaly, 1, 3, 4, 6, 8, 9, 34, 35, 37, 55, 57, 60–63, 121, 140, 149, 193, 194, 205, 217, 218, 241, 243, 246, 250, 264, 276, 277, 283, 286, 289, 293
Anomaly detection, 5, 6, 13, 19, 26, 36, 37, 119, 122, 131, 135, 177, 193, 194, 227, 236, 241, 257, 264, 276, 279, 305–307
Area under curve (AUC), 270
Artificial neural networks (ANN), 13, 82, 275
Autoencoders, 195, 206–208, 216, 217, 257, 261–263
Automatic process monitoring, 4
Average run length (ARL), 13, 93, 100

B

Bias fault, 174, 284, 286
Bidirectional LSTM, 242
Boltzmann machine (BM), 210, 211, 215, 235, 239
Border gateway protocol (BGP), 241
Busy scenes, 264–266, 268, 269, 272

C

Canonical correlation analysis (CCA), 305
Canonical variate analysis (CVA), 11
Chemical oxygen demand (COD), 290
Classification and regression trees (CART), 275
Clustering algorithms, 193, 195, 196, 218, 226, 227, 241, 243, 246, 249–251, 306
Coastal WWTP, 290
Collinearity, 19, 20, 26, 27, 30, 119, 149, 156, 157
Collinearity problem, 21
Complexly related variables, 111
Conditional RBMs, 236, 237
Continuous latent space, 207
Contractive autoencoder, 206, 209, 210
Contrastive divergence (CD), 213, 236, 238
Control limits, 33, 35, 43, 172, 174, 188
Controller area network (CAN), 241
Conventional
  autoencoders, 208
  LVR, 20, 47, 180–182
  monitoring, 50, 130, 131, 155, 158, 164–166, 170, 187, 307
  PLS modeling techniques, 143
  RNNs, 226
  Shewhart, 172–175
Convolutional neural networks (CNN), 257
Cumulative percent variance (CPV), 24, 31

https://t.me/learningnets

CUSUM, 8, 10, 20, 32–34, 36, 37, 50, 130, 157, 158, 168, 170, 171, 186, 305, 306
  decision statistic, 34
Cyclic variables, 106

D

Decentralized WWTP, 290, 291, 297
Deep belief network (DBN), 195, 211, 213–215, 218, 226, 239, 269, 273, 275, 276, 278, 279
Deep Boltzmann machines (DBM), 13, 195, 211, 213, 215, 239
Deep learning, 13, 193, 195, 206, 210, 213, 217–219, 225–227, 236, 239, 250, 256, 257, 261, 263, 264, 267, 272–275, 306–308
Deep learning methods, 195, 257, 275, 285, 305, 307
Deep stacked autoencoder (DSA), 195
  model, 258, 263, 264, 266–268, 285
Denoising autoencoders, 206, 208
Designing monitoring, 158, 307
Detection
  efficiency, 61, 174, 246
  performance, 8, 13, 37, 46, 140, 143, 147, 148, 157, 172, 173, 175, 185, 186, 201, 218, 246, 273, 286, 287
  sensitivity, 227
  threshold, 8, 130, 135–137, 146, 172, 201, 265, 266, 274, 295
Discrete wavelet transform (DWT), 161
Dissolved oxygen (DO), 291
Driving assistance systems (DAS), 255
DSA, *see* Deep stacked autoencoder

E

Effective sample size (ESS), 91
Environmental processes, 1, 4, 6, 9, 14, 19, 30, 42, 47, 50, 63, 119, 124, 225, 227, 241, 242, 307
EWMA, *see* Exponentially weighted moving average
Expectation maximization (EM), 201
Expected error rate (EER), 101
Exponentially weighted moving average (EWMA), 8, 10, 12, 20, 32, 34–37, 50, 93, 94, 130, 138, 158, 168, 170, 171, 180, 186, 201, 243, 250, 295, 305, 306
  charting statistics, 95
  charts, 33
  decision function, 35
  filter, 168
  monitoring scheme, 39, 156
  monitoring statistic, 34
  multiscale, 157, 171
  multivariate, 11, 44, 94, 97, 306
Expressive RNNs, 240
Extreme learning machine (ELM), 99

F

False alarm rate (FAR), 185
False negatives (FN), 13
False positive rate (FPR), 13, 61, 246
False positives (FP), 13
Fault
  classification, 99
  diagnosis, 27, 50–53, 71, 92, 98, 112
  identification, 5, 99
  location, 103
  management, 4
  measurement, 3, 9, 77
  probability, 89
  prognosis, 112
Fault detection (FD), 306
Fault isolation (FI), 4, 20, 50, 51, 53, 71, 72, 74, 78–80, 82, 85–89, 92, 95–98, 100, 101, 104, 108, 112, 306
Faultless datasets, 136, 137, 140
Fiber grating (FG), 255
Finite impulse response (FIR), 31, 167
Fraud detection, 41
Functional principal components, 150

G

Gated recurrent unit (GRU), 226, 230, 234
Gaussian mixture model (GMM), 12
Gaussian processes (GP), 225
Generalized likelihood ratio (GLR), 8, 20, 37, 274, 305

H

Hellinger distance (HD), 41, 130, 146, 306

https://t.me/learningnets

Hidden layer, 206, 208, 209, 211, 214–216, 227, 232, 234, 235, 240, 262, 267, 284
Hidden units, 210, 212, 213, 216, 230, 236–238, 240
Historical faults, 99
Human detection, 256

I

In control (IC), 71
Independent component analysis (ICA), 11, 12
Independent variables, 24
Industrial process, 2, 10, 12, 20, 32, 42, 45, 121, 137, 155, 177
Industrial process monitoring, 1, 10, 11, 112
Influent characteristics (IC), 20, 55, 290
Inspected process, 8–10, 12, 19, 39, 42, 47, 121, 150, 241–243
Intermittent
  anomaly detection, 148
  sensor faults, 286
Intrusion detection, 226, 241, 242, 250

K

Kernel density estimation (KDE), 41, 50, 110, 130, 137, 199
Kernel entropy component analysis (KECA), 99
Kernel principal components analysis (KPCA), 12, 120, 131, 132, 134, 136–138, 140, 142, 150, 289, 306
  detection thresholds, 137
  monitoring, 135
  multiscale, 137
KPCA, *see* Kernel principal components analysis

L

Lagged variables, 32, 150
Lasso EWMA (LEWMA), 93–95, 97
Latent space, 121–124, 126, 129, 207
Latent variable, 12, 19, 20, 22, 23, 26, 28, 29, 119, 122, 124, 125, 127, 143, 156, 176, 178, 187, 201, 202, 206, 207, 210, 305, 306
Latent variable regression (LVR), 19, 20, 22, 23, 119, 156, 305
Linear discriminant analysis (LDA), 289
Linear LVR methods, 119–122, 149
Linear LVR models, 20, 63, 122
Linear PLS, 120, 123–126, 128, 129, 142
Local monitors, 98
Locally linear embedding (LLE), 120, 289
Long short-term memory, *see* LSTM
Lower control limit (LCL), 33
LSTM, 13, 226, 227, 230, 231, 234, 235, 239–242, 250
LVR, *see* Latent variable regression

M

Markov chain Monte Carlo (MCMC), 212
MCUSUM, 44, 45, 47, 48, 50, 306
  decision statistic, 45
Mean absolute error (MAE), 145
Mean squared error (MSE), 143, 180
Missed detection, 37, 173, 175
Missed detection rate (MDR), 102, 166, 172–175, 185
Monitoring, 5, 10, 12, 19, 32, 39, 43, 71, 83, 88, 95, 96, 119, 121, 130, 148–150, 157, 170, 171, 177, 183, 264, 275, 287, 288, 290, 291, 297, 307, 308
  air quality, 307
  autocorrelated data, 165
  industrial processes, 10
  influent measurements, 55
  KPCA, 135
  multiscale, 172
  nonlinear processes, 121, 137
  ozone measurements, 287
  ozone pollution, 297
  univariate schemes, 39
Multiperson detection, 271
Multiple kernel anomaly detection (MKAD), 241
Multiscale
  EWMA, 157, 171
  KPCA, 137
  LSTM, 241
  monitoring, 10, 157, 171, 172, 177
  PLS models, 178
  representation, 157, 159, 164, 167, 170, 173, 174, 187

https://t.me/learningnets

Multivariate
CUSUM chart, 98
distributions, 87, 98
fault detection, 71, 157, 166, 187
fault isolation, 97
normal distribution, 73, 79
process monitoring, 32, 43, 104, 140, 177, 179
process variables, 170
statistical monitoring techniques, 305
statistical process monitoring, 112
Multivariate EWMA (MEWMA), 11, 44–48, 50, 83, 94, 97, 274, 306
monitoring schemes, 44, 46

N

Neural networks (NN), 127, 193
Nonlinear
LVR models, 121, 122, 305
process monitoring, 63, 289, 306
Nonlinear PLS (NLPLS), 120, 121, 125, 127, 143–145, 149, 150
Nonparametric threshold, 41, 61, 109, 110
Nonshifted variables, 97

O

Obstacle detection, 217, 218, 255–258, 260, 261, 263–265, 269–273
OCSVM, 13, 137, 140, 202, 204, 217, 218, 243, 244, 247, 249, 250, 276, 278, 279, 283, 285, 286, 288, 290, 291, 295
detection algorithm, 293
detector, 138
Offline detection methods, 5
One-class SVM, *see* OCSVM
Online detection, 5
Ordinary least squares (OLS), 20, 21
Outlier detection, 306
Ozone monitoring, 278
Ozone pollution, 13, 274–276, 287, 306

P

Parameter selection score (PSS), 97
Partial least squares (PLS), 11, 20, 27, 98, 119, 156, 305
Partitioning around medoids (PAM), 196
Performance
KPCA, 140
monitoring, 39, 155, 305
NLPLS, 121
obstacle detection, 273
Performance assessment, 87
Photochemical ozone pollution, 283
PLS, *see* Partial least squares
Plug flow reactor (PFR), 143
Polynomial PLS, 12, 123, 126, 130, 142
Postprocess variables, 112
Potential anomalies, 204
Prediction performance, 181
Predictor variables, 26, 105, 109
Principal component analysis (PCA), 11, 77, 109, 119, 194, 274
Principal component regression (PCR), 11, 20, 156
Principal component subspace (PCS), 77
Principal components (PC), 20, 23–25, 28, 47, 55, 62, 77, 95, 109, 120, 124, 131, 133, 178, 179
Probability density functions (PDF), 37, 40
Process
faults, 4
WWTP, 139, 288
Progressive faults, 149

R

Radial basis function (RBF), 135, 203, 283
RadViz, 52, 53, 62, 63
Ramp fault, 148
Raw residuals, 148
RBM, *see* Restricted Boltzmann machines
Recurrent deep learning, 227
Recurrent neural network (RNN), 13, 226–230, 244, 290
Regularized canonical correlation analysis (RCCA), 119
Residual subspace (RS), 77
Residuals, 8, 12, 25–29, 31, 47, 48, 57, 61, 106, 107, 109, 111, 129, 130, 146, 149, 156, 165, 184, 185, 187, 242, 243, 250
uncorrelated, 39, 156
Response variables, 105–107

https://t.me/learningnets

Restricted Boltzmann machines (RBM), 13, 210–212, 215, 226, 237, 244
  model, 211, 212, 215, 237
Ridge regression (RR), 20, 21
RNN, *see* Recurrent neural network
Road traffic monitoring, 257
Root mean squared error (RMSE), 145

S

Satisfactory detection performances, 137
Shallow OCSVM algorithm, 285, 286
Shewhart
  fault detection, 187
  multiscale, 174
  scheme, 33, 36, 39, 44, 174, 187
Shifted variables, 71, 73, 74, 77, 79, 81–83, 85, 87, 91–93, 95, 97, 100–103, 110
Single fault, 79, 91
Singular value decomposition (SVD), 24
Spline PLS model, 126
Squared prediction error (SPE), 12, 49, 129, 139
Stacked autoencoder, 195, 207, 213, 216, 263, 275
Stacking several RBMs, 211
Standardized residuals, 91, 92, 107
Step fault, 175
Superior performance, 181
Supervised WWTP, 138
Support vector data description (SVDD), 194, 195, 203
Support vector machine (SVM), 13, 99, 193, 202, 256
Support vector regression (SVR), 275
SVM, *see* Support vector machine

T

Temporal dependencies, 226–229, 236, 237, 239, 240, 244, 250, 251, 306
Tennessee Eastman (TE) process, 98, 99
Termed variables, 72
Timely detection, 155
Total suspended solids (TSS), 291
True negatives (TN), 13
True positive rate (TPR), 13, 61, 270
True positives (TP), 13

U

Uncorrelated residuals, 39, 156
Uncovering intermittent faults, 148
Undercomplete variational autoencoders, 206
Undetected anomalies, 138
Undetected faults, 1
Unfiltered residuals, 146, 148
Univariate
  CUSUM scheme, 45
  EWMA, 43
  EWMA scheme, 36, 43
  monitoring schemes, 10, 20, 32, 39, 42–44, 170, 305
  process monitoring, 32
Unshifted variables, 79, 83, 95, 100, 101
Unsuited detection performance, 247
Unsupervised anomaly detection, 194, 195
Unsupervised anomaly detection methods, 193, 194
Unsupervised deep learning, 217, 257, 273, 290, 297, 306
Upper control limit (UCL), 33, 80

V

Vanilla RNNs, 227, 228
Variational autoencoder, 207
Visible layer, 214–216, 239, 240
Visualization RadViz, 20
Visualized variables, 52
Volatile organic compounds (VOC), 274, 282

W

Wastewater treatment plants (WWTP), 19, 55, 57, 59, 60, 63, 138, 225, 244, 246, 288–291, 296, 297
Weak detection, 61
WWTP, *see* Wastewater treatment plants

https://t.me/learningnets

STATISTICAL PROCESS MONITORING USING ADVANCED DATA-DRIVEN AND DEEP LEARNING APPROACHES

Theory and Practical Applications

Provides an in-depth understanding of fault detection and attribution in complex and multivariate systems

Due to advances in technology, researchers today face chemical engineering and environmental processes that have become far more complex, with multiple key variables that need to be monitored simultaneously. ***Statistical Process Monitoring using Advanced Data-Driven and Deep Learning Approaches*** tackles multivariate challenges in process monitoring by merging the advantages of univariate and traditional multivariate techniques to enhance their performance and widen their practical applicability. The book proceeds with merging the desirable properties of shallow learning approaches—such as a one-class support vector machine and k-nearest neighbors and unsupervised deep learning approaches—to develop more sophisticated and efficient monitoring techniques.

These developed approaches are applied to monitor many processes, such as wastewater treatment plants, detection of obstacles in driving environments for autonomous robots and vehicles, robot swarm, chemical processes (continuous stirred tank reactor, plug flow rector, and distillation columns), ozone pollution, road traffic congestion, and solar photovoltaic systems. ***Statistical Process Monitoring using Advanced Data-Driven and Deep Learning Approaches*** provides a concise guide for practitioners and researchers in academia and industry working in chemical and environmental engineering.

Key Features

- Familiarizes the reader with the most suitable data-driven based techniques in statistical process modeling, including multivariate statistical techniques and deep learning-based methods
- Provides an in-depth understanding of fault detection and attribution in complex and multivariate systems
- Includes case studies and comparison of different methods

Fouzi Harrou received his PhD in systems organization and security from the University of Technology of Troyes in Troyes, France. After holding many prior research positions, since 2015 he has been a post-doctoral fellow at the Division of Computer, Electrical, and Mathematical Sciences and Engineering at King Abdullah University of Science and Technology.

Ying Sun received her PhD in statistics from Texas A&M University in 2011, followed by a two-year postdoctoral research position at the Statistical and Applied Mathematical Sciences Institute and at the University of Chicago. Previously an assistant professor at Ohio State University, she is currently a researcher at King Abdullah University of Science and Technology, where she founded and now leads an environmental statistics research group.

Amanda S. Hering received her PhD from Texas A&M University in statistics in 2009 and is currently an associate professor at Baylor University in Waco, Texas. Her research interests are in modeling big, multivariate, spatial datasets; developing methods for categorical spatial data; and detecting outliers and faults for process and data control.

Muddu Madakyaru received his PhD in process control from the Indian Institute of Technology in Mumbai, India and is an associate professor in the Department of Chemical Engineering at the Manipal Institute of Technology, Manipal Academy of Higher Education, India. His research interests are in advanced process control, including system identification, fault detection and diagnosis, model predictive control and latent variable regression modeling using wavelets.

Abdelkader Dairi received his PhD degree in computer science from University of Oran 1 Ahmed Ben Bella in Algeria. With over 20 years of programming experience, he has held positions such as senior Oracle database administrator (DBA) and enterprise resource planning (ERP) manager. His research interests include deep learning approach for autonomous robot navigation, computer vision, image processing, and mobile robotics.

ISBN 978-0-12-819365-5

9 780128 193655

ELSEVIER elsevier.com/books-and-journals

https://t.me/learningnets