ADVANCES IN BIG DATA ANALYTICS FOR MODELING, OPTIMIZATION AND CONTROL: APPLICATIONS IN PROCESS SYSTEMS ENGINEERING

A Dissertation

by

MELIS ONEL

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Efstratios N. Pistikopoulos
Committee Members,	Sergiy Butenko
	Mahmoud El-Halwagi
	M. M. Faruque Hasan
Head of Department,	Arul Jayaraman

December 2019

Major Subject: Chemical Engineering

ABSTRACT

The advancement in technology and computational power has enabled large amounts of data collection in real time, which has initiated the *"Big Data"* era. Big data analytics is playing an essential role in academy, business as well as government, providing assistance in decision-making in numerous fields. In this work, selected challenges in process systems engineering are addressed through advances of and applications in big data analytics.

First, challenges in chemical process monitoring, such as fault detection and diagnosis, are addressed by exploiting the industrial data abundance. Data-driven process monitoring has become one of the key approaches in industry to maintain a safe and robust operation while increasing process efficiency to ensure high standards in product quality. In this work, a novel fault detection and diagnosis framework based on nonlinear Support Vector Machine-based feature selection and modeling algorithm is developed for the simultaneous fault detection and diagnosis of chemical processes (s-FDD framework) in both continuous and batch modes. The major advantage of the s-FDD framework is its ability to identify the optimal number of process variables diagnosing the fault while providing highly accurate models for fault detection. The s-FDD framework is further improved with the integration of (i) maintenance optimization strategies, and (ii) multi-parametric model predictive control (mp-MPC) in order to maximize the process profitability and resilience while minimizing process downtime. A novel "parametric fault-tolerant control" concept has been developed for chemical/biochemical processes that serves as an active fault tolerant strategy. This work can serve as an online decision support tool during process operations to enable (i) early detection and diagnosis of process faults, and (ii) rapid actions to adapt altering process or controller conditions to achieve smarter operation.

Secondly, we address challenges in understanding the environmental health impact of complex substance/mixture exposures during environmental emergency-related contamination events (i.e. hurricanes). A data-driven framework is developed to group complex substances with known chemicals by analyzing high dimensional analytical chemistry data, and predict their impact on the environmental health. This facilitates the communication of substance characteristics and decision-making via read-across in order to mitigate the adverse environmental health effects.

DEDICATION

To my loving family; My parents, Nursevin and Tevfik, My husband, Onur, My sister, Nil.

ACKNOWLEDGMENTS

I would initially like to offer my sincerest gratitude to my Ph.D. advisor, Professor Efstratios N. Pistikopoulos. This dissertation would not have been possible without his guidance, and immense support. I have greatly benefited from his wisdom throughout this journey both on academic and personal levels. I am also grateful to him for providing me a unique opportunity to complete my Ph.D. studies with him after the sudden passing of my late advisor Professor Christodoulos A. Floudas. It has been an honour to work under the guidance of Professor Floudas in the first two years of my Ph.D. He always pushed me to reveal the very best of me for my future professional and personal life. I have been privileged to work with the two great mentors in the process systems engineering field, which has played a pivotal role in shaping my career, and I will treasure it for the rest of my life.

I would like to express my gratitude to my dissertation committee members: Professor Sergiy Butenko, Professor Mahmoud El-Halwagi, Professor M. M. Faruque Hasan, and late Professor Sam Mannan for their insightful suggestions and valuable time. I am thankful to our academic collaborators within the Texas A&M Superfund Research Center who have helped shaping my dissertation work: Professor Ivan Rusyn, Professor Fred Wright, Professor Lan Zhou, and Professor Mike Mancini. I would like to thank to the Texas A&M Energy Institute staff with whom I have shared the same work place and felt their constant support whenever we need them. Special thanks to Valentini Pappa, Robyn Pearson, Lisa Groce and Jeff Sammons for their significant help for our Texas A&M Energy Research Society activities. I would like to thank Ashley Henley from the Artie McFerrin Department of Chemical Engineering for helping me throughout my Ph.D. by patiently answering my endless questions.

I have greatly enjoyed being a member of the Floudas and Pistikopoulos labs. I owe a great deal to Dr. Nikolaos A. Diangelakis, Dr. Maria Papathanasiou, Dr. Ioana Nascu and Dr. Richard Oberdieck for easing the transition to the Pistikopoulos group. I am grateful to interact with the senior Floudas lab members; Dr. Fani Boukovala, Dr. Chris Kieslich, Dr. Phanourios Tamamis, Dr. Yannis Guzman, Dr. George Khoury, Dr. Alexander Niziolek, Dr. Logan Matthews and Dr. Onur Onel. Their encouragement has kept me moving forward during the hard times after the untimely

passing of Professor Floudas. I would also like to appreciate the constant motivation and support from the Floudas family; Mrs. Fotini Flouda, Ismini Flouda and Stefanos Baratsas. I would like to offer my sincerest gratitude to Burcu Beykal, whom I have worked together during my Ph.D. studies and proudly represented the data science team of the group. I greatly appreciate our fruitful discussions and fun memories during the conferences. It was a pleasure working with Baris Burnak and generating cutting-edge research together in the last year of my Ph.D. I have enjoyed interacting with Dr. Rajib Mukherjee, Dr. Styliani Avraamidou, Gerald Ogumerem, Justin Katz, Cosar Doga Demirhan, William Tso, Yuhe Tian, Iosif Pappas, Cory Allen, Christopher Gordon, and Denis Su-Feher during my studies. I would also like to extend a special thank you to Salih Emre Demirel, Iosif Pappas, Dr. Nikolaos Diangelakis and Yaling Nie for their company in the office after the work hours.

I cannot express my appreciation enough to my family and close friends. I am grateful to my loving parents, Nursevin and Tevfik for always believing in me and raising me into the woman I am today. The distance did not hinder my mother to be my main source of motivation throughout the challenging projects. My father has set a great example in my life with his hard-working and perseverance. I have learned not to give up when I face with challenges, yet use them to challenge and advance my intellect. I am blessed to have the support of my sister, Nil. We have been in constant communication despite the time difference which has helped me to stay focused and motivated. I could not ask for a better sister in this world. I would like to thank my aunt Esra Canbaz for being a role model in my academic life, and Metin Canbaz for supporting my career as a chemical engineer. A special thank you to my dear cousin Sera Canbaz for keeping me motivated during my studies. I am thankful to my parents-in-law, Selma and Orhan. Their regular visits to the USA keep me and Onur focused on our work when we miss home. I am grateful to my grandmother Feriha Yilmaz for raising me and supporting me. I would like to extend a special thank you to my close friends from Turkey; Gulce Guldur, Dilruba Ekici, Can Ekici, Serhat Ersahin, Ozgur Yasar Caglar, and Mustafa Turkekul. They have proved that distance is not an obstacle to remain my closest friends. I am grateful to the support of the Edmonds family in Princeton. Another special thank you to Melda Sezen-Edmonds and Merrill Edmonds. Finally, I am blessed to have my loving husband, Onur by my side. He has been my number one support from the first day of my Ph.D. journey. He has always eased my life by solving the problems, and his endless love and support

have kept me moving towards to see the end of my challenging Ph.D. journey. This dissertation is dedicated to my parents; Nursevin and Tevfik, my husband, Onur and my sister, Nil.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a dissertation committee consisting of Professors Efstratios N. Pistikopoulos, Mahmoud El-Halwagi, M. M. Faruque Hasan of the Artie McFerrin Department of Chemical Engineering and Professor Sergiy Butenko of the Department of Industrial and Systems Engineering.

The data analyzed for Chapter 6 was provided by Professor Ivan Rusyn from the Department of Veterinary Integrative Biosciences at Texas A&M University. The experiments to generate the data analyzed in Chapter 6 were conducted by Kyle Ferguson of the Department of Veterinary Integrative Biosciences under the guidance of Professor Ivan Rusyn.

All other work conducted for the dissertation was completed by the student independently.

Funding Sources

This work was funded by the Texas A&M Energy Institute, U.S. National Institute of Health (NIH) grant P42 ES027704, and National Science Foundation (NSF CBET- 1548540). The content of this work does not necessarily represent the official views of the NIH or NSF.

NOMENCLATURE

2D	2-Dimensional
3D	3-Dimensional
ANN	Artificial Neural Network
AUC	Area Under the ROC Curve
CART	Classification And Regression decision Trees
CAS	Chemical Abstracts Service
CPV	Current Process Variable
C-SVM	C-parameterized Support Vector Machines
DOE	Department Of Energy
DPCA	Dynamic Principal Component Analysis
F-M	Fowlkes-Mallows
FAR	False Alarm Rate
FDA	Fisher Discriminant Analysis
FDI	Fault Detection and Identification
FDR	Fault Detection Rate
FN	False Negative
FP	False Positive
FTC	Fault Tolerant Control
GCxGC-FID	Two-dimensional Gas Chromatography with Flame Ionization Detector
GC-MS	Gas Chromatography-Mass Spectrometry
GERG	Geochemical and Environmental Research Group

HPB	Historic Process Behavior
HPV	Historic Process Variable
HRP	Heavy Refining Products
IM-MS	Ion Mobility-Mass Spectrometry
IoT	Internet of Things
KNN	K-Nearest Neighbor
LRP	Light Refining Products
mp-MPC	Multi-parametric Model Predictive Control/Controller
MPC	Model Predictive Control
MPCA	Multi-way Principal Components Analysis
NIST	National Institute of Standards and Technology
NOC	Normal Operating Condition
OGO	Other Gas Oils
PAH	Polycyclic-Aromatic Hydrocarbon
PARAFAC	Principal Factors Analysis
PAROC	Parametric Optimization and Control
PCA	Principal Component Analysis
PenSim	Penicillin production Simulation
PID	Proportional-Integral-Derivative control
PLS	Partial Least Squares
PSE	Process Systems Engineering
RAYMOND	RAYpresentative MONitoring Data
RBF	Radial Basis Function
RF	Random Forest
ROC	Receiver Operating Characteristics

s-FDD	Simultaneous Fault Detection and Diagnosis
SOM	Self-Organizing Map
SPM	Statistical Process Monitoring
SRGO	Straight Run Gas Oils
SRM	Standard Reference Materials
SRP	Superfund Research Program
SVC	Support Vector Classification
SVD	Singular Value Decomposition
SVDD	Support Vector Data Description
SVM	Support Vector Machine
SVR	Support Vector Regression
TAMU	Texas A&M University
TN	True Negative
ToxPi	Toxicological Prioritization Index
TP	True Positive
UVCB	Unknown or Variable composition, Complex reaction products, or Biological materials
VHGO	Vacuum and Hydro-treated Gas Oils

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

1. INTRODUCTION*

The advancements in technology and computational power has enabled large amounts of data collection in real time, which has initiated the *"Big Data"* era. Big data analytics has become an essential tool for academy, business as well as government, providing assistance in decision-making process in numerous fields, specifically process systems engineering (PSE).

In this dissertation, selected challenges in PSE are addressed through advances and applications in big data analytics for modeling, optimization and control of processes/systems. The tackled challenges related to energy and process efficiency, and environmental health are introduced respectively, where data can facilitate the interpretation of processes and undesired conditions, and further provide solution. The motivation and background behind the stated challenges are presented, and the targeted objectives towards solving them are provided below.

1.1 Smart Manufacturing

Energy and process efficiency is a key player in industry in order to maximize the profit and minimize the adverse impacts on the environment. Simultaneous achievement of high efficiency, safety, and profitability is of utmost importance and urgency in modern manufacturing and process industries, yet a challenging one due to the (i) growing demand for higher product quality, (ii) constant need for higher process efficiency at minimum cost, (iii) increasing stringency of environmental and safety regulations (7; 5). This need has urged industry to adopt and automatize novel process technologies and methodologies that can optimize their process, in other words to pursue *"Smart Manufacturing"*(8). Recently, smart manufacturing, which integrates automated, digital technologies with advanced manufacturing capabilities throughout the product life-cycle(9), has gained significant interest from academia, industry and government encouraging advancements in information, characterization, process, and sensing technology. It improves energy efficiency and accordingly manufacturing performance by implementing next-generation sensor and control

*Part of this chapter is reprinted with permission from "Big data approach to batch process monitoring: Simultaneous fault detection and diagnosis using nonlinear support vector machine-based feature selection" by Onel, M. and Kieslich, C.A. and Pistikopoulos, E.N., 2018, *AIChE Journal*, Vol. 65, Issue 3, pp 992-1005, John Wiley Sons [2018] by John Wiley and Sons and Copyright Clearance Center and "A nonlinear support vector machine-based feature selection approach for fault detection and diagnosis: Application to the Tennessee Eastman process" by Onel, M. and Kieslich, C.A. and Guzman, Y.A. and Floudas, C.A. and Pistikopoulos, E.N., 2018, *Computers & Chemical Engineering*, Vol. 116, pp 503-520, Elsevier [2019] by Elsevier and Copyright Clearance Center

technologies along with communication devices. This ensures and expedites the collection and dissemination of large amounts of process data, often referred as *"Big Data"*, in real time. This concept has also given birth to emergence of industrial Internet of Things (IoT), which refers to the network of inter-connected industrial equipment and systems that can exchange and process the collected high-dimensional data. In 2012, the potential of Big Data for decision-making in industry has been recognized by former president, Mr. Obama, with the launch of $200 million "Big Data Research and Development Initiative", where the main goals are set as: (i) advancing state-of-the-art core technologies needed to collect, store, preserve, manage, analyze, and share huge quantities of data, (ii) expanding the workforce needed to develop and use big data technologies, and (iii) harnessing technologies to accelerate the pace of discovery in science and engineering, strengthening the national security, and transforming teaching and learning (10). This has launched the Big Data era in numerous PSE fields, including process monitoring, fault detection and diagnosis (11; 12; 13), process optimization and control (14; 15; 16), and healthcare (17). In this work, advances in data-driven fault detection and diagnosis, maintenance optimization and fault-tolerant control of chemical processes is presented.

1.2 Process Monitoring, Maintenance Optimization and Control

The emergence of the fourth industrial revolution, Industry 4.0, along with the recent Big Data initiatives (18; 12) has enabled a research breakthrough in the field of data-driven (or statistical) process monitoring. The main goal is to ensure *smart manufacturing*, a concept envisioned by numerous agencies including the US Department of Energy (DoE) and the National Institute of Standards and Technology (NIST), which describes the motivation to design intelligent factories that can rapidly adapt to changes/disturbances by sharing and analyzing process data during manufacturing operation. Thus, integration of the advancements in information technology (i.e. enhanced networking, cloud services, and data analytics) with operations technology (i.e. adaptive automation, sensor and software technology) is of utmost importance to produce a network communication between process instruments known as industrial Internet of Things (19). As the industry is moving towards such automated and integrated process architecture, the analysis of process data produced in large amount in real time is becoming more practical in various engineering applications to understand underlying trends, and subsequently improve decision-making

for operation. Data-driven process monitoring is one of these major fields where industrial process data play a significant role in accurate and timely decision-making to maintain a safe and profitable operation. As the modern process industry aims for smarter, safer, and more efficient operation, the need for developing novel, more accurate, thus powerful process monitoring and fault detection and diagnosis framework continues to grow. While the computational and technological advancements are leading processes to integrate more operating variables under closed loop control and cause increase in process structure complexity, which obfuscates process control, the Big Data outbreak in industry immensely facilitates process monitoring. Today, by using process big data (historic and/or simulation-based), one can detect faults, diagnose them from key process variables, predict future state of process variables, and prevent any undesired conditions (12).

Data-driven process monitoring exploits multivariate statistics and data mining methods to determine whether a fault has occurred or not during industrial process operations. Here, fault is defined as abnormal process behavior due to an unpermitted deviation in at least one observed variable or computed parameter of the system, where controllers cannot reverse it (20). Faults may occur due to equipment failure, equipment wear, or extreme process disturbances (21). Early and rapid detection and diagnosis of process faults is one of the top major challenges of industry in order to sustain a safe operation and minimize losses in productivity(22). These issues are addressed via process monitoring (i.e. fault detection and diagnosis) techniques, where deviations from normal operation (i.e. faulty operation) and resolving (diagnosing) the characteristics of the detected problem (i.e. fault diagnosis) is the main interest.

1.2.1 Process Monitoring

Process monitoring techniques can be classified into three categories: model-based (a.k.a first-principle based), knowledge-based and data-based methods (20). Model-based methods (23) are based on first-principles which uses of *a priori* physical and mathematical knowledge of the process. Therefore they are apt to yield more accurate solutions than the other techniques. However; the success of model-based methods heavily depend on the accuracy of process models, which is significantly challenging to ensure as the modern industrial processes are becoming increasingly complex in structure. Knowledge-based methods gather the available information on the process performance and develop qualitative or semi-quantitative relations via causal analysis with

signed directed graphs (24; 25), decision trees (26), pattern recognition techniques like artificial neural networks and self-organizing maps (27; 28; 29). The major drawback of these techniques is their dependency on human knowledge which may produce solutions that are susceptible to change (30; 31). On the other hand, data-driven process monitoring methods do not involve any prior knowledge, where models for fault detection and diagnosis are constructed solely on data. Compared to the other two categories, data-driven methods are advantageous in capturing intrinsic complexity of the industrial processes by benefiting of the abundance in process data. Thus, data-driven methodologies have sparked significant interest within the last two decades and their applications have become prevalent in wide range of industries including the chemical, energy, medical, photo-voltaic, semiconductor manufacturing, and steel industries (32; 33; 34; 35; 36; 37; 38; 39).

Today, data-driven or statistical process monitoring (SPM) techniques(40; 5) are providing promising results due to the availability of large amounts of recorded process data. These techniques exploit multivariate statistical analysis and machine learning algorithms to build data-driven models that can determine deviations from normal operation, and partition high dimensional data space into distinct fault regions for identification. The widely accepted, so-called traditional, technique for data-driven fault detection is anomaly/outlier, out-of-control situation, identification via the Hotelling's T^2 and Q-statistics (20; 5). Other data-based methods centering around classification/regression-based analysis have been proposed that employ Artificial Neural Network (ANN) (41) and more recently Deep Learning algorithms (42), Classification and Regression Decision Trees (CART) (43; 44) as well as different Support Vector Machines (SVM) formulations being Support Vector Classification (SVC) (3; 45; 46; 47), Support Vector Regression (SVR) (48), and Support Vector Data Description (SVDD) (49). In particular, a major advantage of Support Vector Machines is their ability to provide nonlinear and robust models for non-Gaussian distributed process data, and due to their succinct representation as convex nonlinear optimization problem to obtain global parameters for models.

Regardless of the data type and mining technique, the ultimate goal in data-driven process monitoring is to attain the highest model accuracy with lowest false-positive rate for fault detection and diagnosis, which is one of the major challenges when dealing with high-dimensional process data. As the data volume grows, the risk of having spurious features increases, which would in turn deteriorate the accuracy of the data-driven models. In particular, the increase in (i) plant-wide pro-

cess control structure complexity with new technologies, and (ii) speed in process data collection has risen the number of process variables and parameters to consider during process monitoring. This paves the way for the use of advanced dimensionality reduction techniques during the model-building phase (50; 51; 52; 53; 54). Accordingly, this has opened a new research area, where there is a growing need for novel data-driven fault detection and diagnosis techniques development that employs powerful dimensionality reduction methodologies, thus produces more accurate and reliable models. Dimensionality reduction can be achieved via either: (i) feature extraction, or (ii) feature selection. In process monitoring setting, features represent process variables used in fault detection model development. Feature extraction entails projection of the features of original space into a new, lower dimensional space, where the extracted features become linear combinations of the original ones.

Prominent feature extraction based dimensionality reduction methods include latent variable-based models being Principal Component Analysis (PCA) (55; 56; 57; 58; 59; 60), Partial Least Squares (PLS) (61; 62; 63; 64), Fisher Discriminant Analysis (FDA)(21), and Correspondence Analysis (65; 66). These methods project the original data into a lower-dimensional space where accurate and simplified characterization can guide process monitoring (40). Nonlinear and dynamic extensions of these techniques (i.e. Kernel PCA/PLS (67; 68), Dynamic PCA/PLS (69; 70)) have also been introduced to handle nonlinearity and serial (temporal) correlations of process data, respectively. However, a major disadvantage of these methods is that they assume Gaussian distributed process data, which poses limitation in producing accurate fault detection and diagnosis (30). Independent Component Analysis (71) is another feature extraction technique which does not assume Gaussian distributed data. Nonetheless, the disadvantages of this technique include (i) unstable monitoring performance due to the random initialization, (ii) ambiguity in the selection of retained independent components, (iii) unclear value assessment of each independent component, and finally (iv) perplexing control limit determination due to non-Gaussian distribution of the extracted independent components (30). Finally, regardless of the assumption on data distribution, feature extraction based methods cause significant loss in information during the transformation of the original features into a lower dimensional space which impairs the interpretation. This reflects as a delay or even deterioration in fault diagnosis due to loss in physical interpretation. This can be prevented with the adoption of feature selection techniques.

Feature selection is the process of selecting the most informative and relevant original features (e.g., process attributes, variables) characterizing the system which is crucial to encapsulate highly nonlinear and interconnected nature of the process data inputs accurately. This in turn improves data-driven, predictive model accuracy and robustness by reducing the probability of overfitting. Furthermore, obtaining the highest model accuracy with "minimum" number of features is of utmost interest for facilitating the results interpretation. Use of feature selection in process monitoring yields high model accuracy without impairing interpretation for diagnosis, and also provides a reference in configuration of sensors in the process for data collection. The later notably becomes important when the measurement resource is limited (72). Therefore, applications of feature selection in fault detection and diagnosis field have been significantly increasing (73; 74; 75; 76), where the researchers have now focused on determining the optimal set of process information (process variables, equipment health data, process parameters etc.) to (i) improve the model accuracy, thus reliability, for fault detection and diagnosis, (ii) minimize the number of process variables diagnosing the detected fault for facilitated interpretation, thus expedited recovery, and (iii) minimize the operation cost by ensuring optimal sensor configuration and sensor network design.

1.2.2 Maintenance Optimization

In order to maintain a safe and profitable operation, the next steps after accurate and rapid fault detection and diagnosis are (i) fault identification and reconstruction, (ii) fault isolation, and (iii) process recovery and maintenance. Fault identification is the determination of the type of the occurring fault. While fault detection corresponds to discovering any abnormalities in an ongoing process, fault identification is finding the exact type of the occurring fault. It is very common to observe multiple different fault types in process industries, therefore fault identification plays a key role in deciding on the next steps for process recovery (30). In addition to the fault type, the magnitude of the disturbance as well as the direction need to be explored to determine the level of corrective actions. Revealing the quanititative details of the detected fault type is known as fault reconstruction which is another important component for process recovery. If fault reconstruction is not precise, this may cause bigger problems during correction actions even though fault detection and diagnosis is done accurately. In other words, even if we know a certain type of fault occurring within the process and the cause, we still need to learn the exact amount of deviation in the causing

process variables to fix the problem and return the operation to the nominal stage. Once the fault type and magnitude are identified, the next task is fault isolation in order to prevent fault propagation. Any failure in ensuring fault isolation may end up causing plant-wide oscillations, and in turn deteriorates control performance of the whole plant. In such cases, diagnosis becomes more challenging due to unexpected propagation pathways. In fact, root cause analysis in plant-wide oscillations has been extensively studied in the literature (77; 78; 79; 80; 81; 82; 83; 84). Final step in process monitoring methodology is to recover the faulty process to normal by selecting and applying necessary maintenance strategies while using the revealed information on the fault.

Maintenance optimization plays a key role for sustaining a safe and profitable operation, recovering and further preventing from faulty condition. Maintenance optimization is a term used to describe the engineering decisions and associated actions performed to optimize (i) process safety, (ii) product quality, and (iii) process efficiency (85; 86). Regardless of the maintenance type, the main goal is to maximize system availability and reliability to meet the targeted product output level and quality, thus to achieve high profitability, without compromising safety and environmental issues (87; 88). Therefore, maintenance optimization is crucial and necessary for industrial operations. One of the major challenges in industry has been to increase plant availability and reliability at the lowest cost. This has even become harder with the emergence of Industry 4.0 (12) and Smart Manufacturing (89) initiatives, where mechanization and automation in industry has reduced the number of production personnel, increased the capital employed in production equipment and civil structures, and thus placed maintenance costs as the second largest spending in operation budget after energy costs (90). Therefore, there is an eminent need for improving

Maintenance strategies are grouped under two broad categories: (i) corrective, and (ii) preventive. The major difference between corrective and preventive maintenance strategies is the existence of a fault/failure in a process. In particular, a fault or failure occurs prior to the application of corrective maintenance tasks, whereas preventive tasks are applied to hinder abnormalities in the process. Therefore, corrective maintenance is a reactive approach, whereas preventive maintenance actions fall under the proactive approach category. In particular, corrective maintenance describes the all actions to correct, repair and if necessary replace the failed units in a system or process. Based on the extremity of the abnormality in a process, corrective maintenance can be grouped into two categories: (i) “run-to-failure” reactive category, where complete unit failure

occurs and equipment replacements are the only solution to correct the abnormality, and (ii) corrective category, where individual/minor faults are identified and addressed when they occur and tasks are performed to avoid a complete equipment failure (91). In other words, reactive maintenance is performed after a failure occurs in a system, whereas corrective maintenance strategies are followed when a fault is detected in the system to remediate the process.

On the other hand, preventive maintenance strategies are further divided into two groups: (i) timed maintenance, where maintenance tasks are performed based on a pre-determined schedule to prevent failures, and (ii) predictive maintenance, where the schedule of preventive maintenance tasks are based on numerous factors such as equipment condition, process risk level, reliability condition etc (92; 93; 94). Figure 1.1 summarizes the grouping of maintenance strategies. The advantage of corrective maintenance strategies is fully utilization of the equipment until a problem is detected, thus lower short-term maintenance costs. Whereas the disadvantage is the increased long-term maintenance costs due to increased process downtime in the event of unexpected abnormalities, thus lower availability and reliability. Therefore, it is crucial to take proactive approach and perform preventive maintenance tasks before failure occurs. Yet, one also needs to avoid scheduling unnecessary preventive tasks, which increases the operation cost, and find the optimal maintenance schedule to attain high availability and reliability. As a result, the object of maintenance optimization is finding the balance between the two main maintenance strategies. Specifically, the goal is to find optimal frequency and timing of corrective and preventive maintenance tasks that would maximize the process availability and reliability at minimum cost (95). In fact, with the increase in data collection in real time and huge amount of process and equipment data availability, data analytics has become an essential in achieving this goal, which is supported by the increase in studies that combine data analytics with maintenance optimization (96; 97; 98; 99; 100).

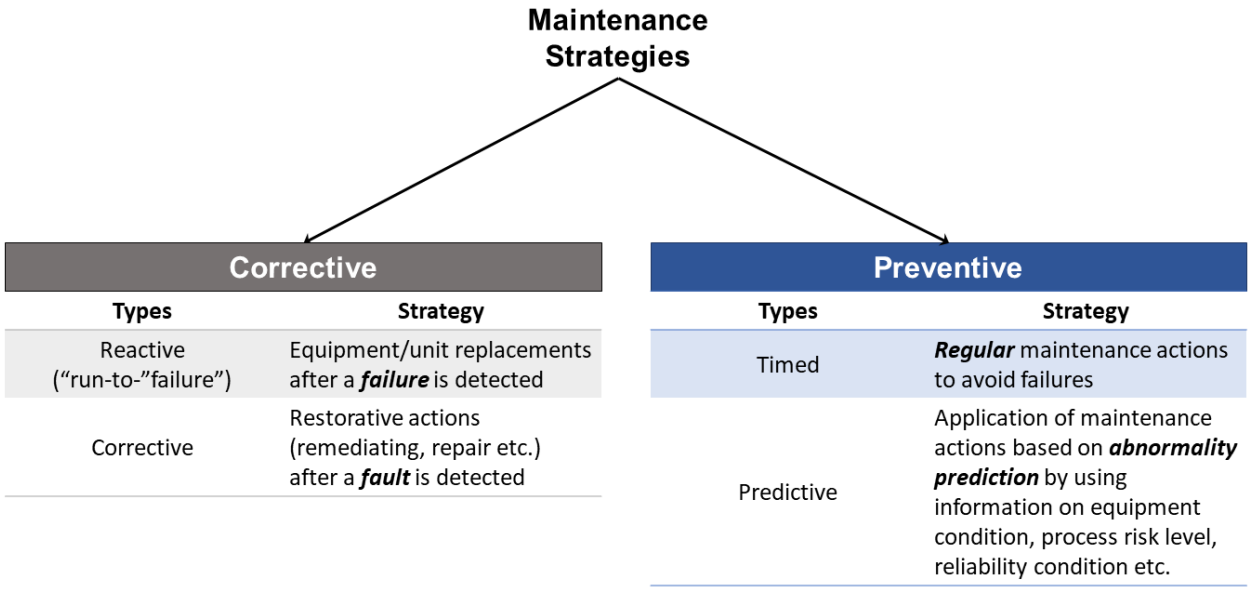


Figure 1.1: Maintenance strategies.

In addition to process availability and reliability, achieving high process resilience is of utmost importance and one of the major growing demands in process systems engineering (101). As the automation increases in industry, the systems become more vulnerable to faults (102). Deficiencies in sensors, actuators, controllers or disturbances in a process may cause fault occurrence, which can be amplified within a closed-loop control systems and lead to a serious failure unless faulty process is recovered rapidly and returned back to the nominal condition (103). Therefore, early detection and diagnosis of faults is crucial on order to prevent fault propagation and further development of simple faults into failure. One way to handle this problem is to build fault-aware or fault-tolerant control (FTC) systems, which would understand the existence of faults in a process and adjust the controller actions accordingly to guarantee stability and satisfactory performance.

1.2.3 Fault-tolerant Control

FTC methods are classified into (i) active and (ii) passive fault-tolerance strategies. Passive approaches use robust control techniques to protect the system from instabilities, hence ensure the closed-loop control system to stay insensitive to certain faults, by using the existing controller parameters. On the contrary, active fault-tolerant strategies do not necessarily use existing controller

parameters. They use online fault detection and identification (FDI) mechanism to monitor the process and get information on faults for further fault accommodation. Therefore, reliability of online FDI mechanisms play a significant role in the effectiveness and robustness of active FTC strategies. Yet, regardless of the FTC approach, the main goal is to recover the original system performance by using the same control objective (104; 105). Active approaches are further grouped under two categories: (i) projection-based, and (ii) online reconfiguration/adaptation. For projection based FTC approaches, in addition to an accurate and robust FDI mechanism for getting online fault information, *a priori* knowledge on expected fault types is required to design controllers *a priori* for all possible faults that can be observed in the process. Hence, this technique necessitates offline calculation and storage of control laws. Once the information is received on a certain fault from online FDI system, the corresponding projected controller actions are activated via one of the three approaches: (i) model switching or blending, (ii) scheduling, and (iii) prediction (102). On the other hand, active FTC can also be achieved via adaptive control and re-configuration/re-structuring of the control signal distribution (a.k.a control allocation). The taxonomy of FTC methods is summarized in Figure 1.2.

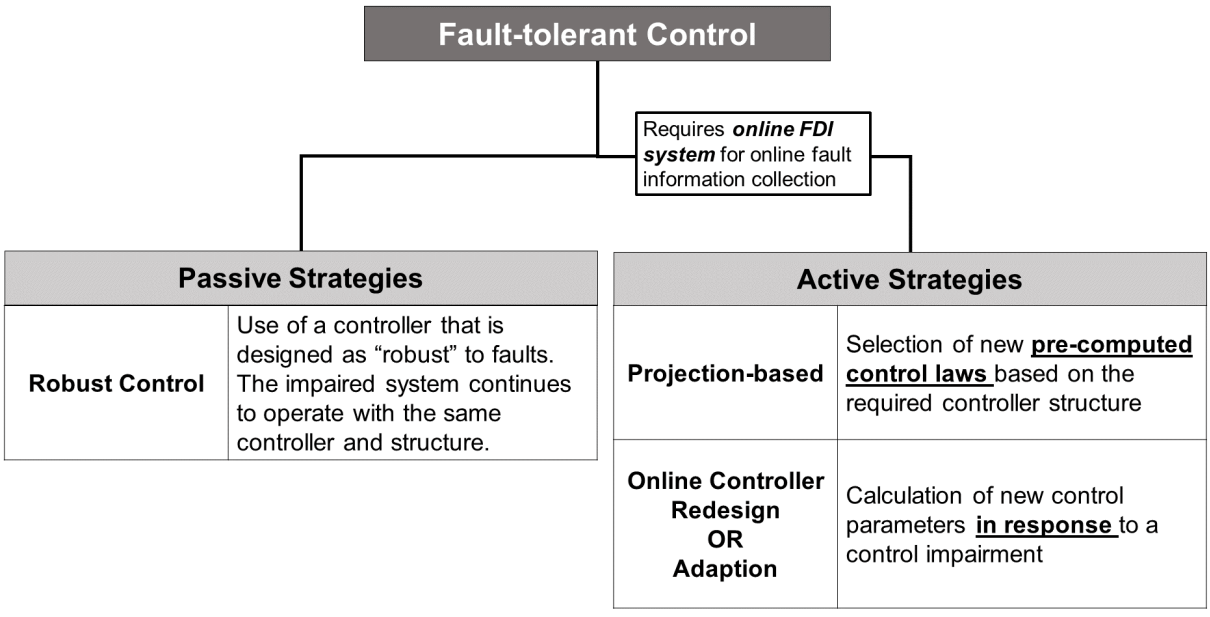


Figure 1.2: Categorization of fault-tolerant control strategies.

FTC has become an emerging research field in automatic control in the late 70s in order to overcome the limitations posed by conventional feedback control, where conventional feedback control design may end up performing poorly and lead to instabilities in the event of actuator, sensor or another system component malfunctions (104). The motivation in designing fault-tolerant systems have been driven by the problems observed in aircraft control systems, where particular automatic fault accommodation strategies are needed to guide pilots, and prevent development of simple faults into severe failures that may lead to accidents (106; 107). FTC has been studied extensively in the literature (108; 109; 110; 111); however the interest has spiked especially in the late 90s and early 2000s (112; 103; 113; 114), and the applications have started to become prevalent especially in safety-critical systems with the increase in computational power and advancements in sensor technology (115; 116). Fault-tolerant systems are widely used in numerous fields including aircrafts (107; 117; 118), mechatronics (119), power plants (120; 121), spacecrafts (122; 123; 124; 125), and industrial plants producing hazardous materials such as nuclear plants (126; 127). The number of the application areas is yet increasing as the demand for higher process availability and profitability grows, and tolerance for process failures decreases in industry under smart manufacturing initiatives (111). Moreover, development of novel fault tolerance strategies is of great interest for enabling operational reliability and resilience, another key pillar for smart manufacturing.

1.2.4 Objectives

The common practice in industry is to (i) monitor chemical process data, (ii) detect faults occurring during operation, and (iii) diagnose faults after detecting them by performing further analysis. In this work, our perspective is to combine detection and diagnosis in order to decrease the process runtime spent under faulty condition, and expedite the corrective actions so as to maintain safe operation and prevent further reduce in profit. Therefore, the idea is to build data-driven models which uses a reduced, particularly optimum (i.e. the most informative) set of features (process variable information), to detect abnormalities in a process. This can be achieved via performing modeling and model-informed feature selection simultaneously.

Model-informed feature selection is finding the optimal subset of the input process data variables by using the model accuracy information during training stage. In other words, the feature

elimination criteria is based on the model performance with selected set of features, where optimal set of features can be identified by an iterative procedure. Using only the most descriptive process features, one can (i) achieve high-performance models, (ii) reduce the model complexity, and (iii) lessen data collection and storage requirements. In terms of process monitoring, this corresponds to developing accurate, simple yet reliable models for fault detection and diagnosis. Thus, the first objective is to develop a framework for simultaneous fault detection and diagnosis, which employs simultaneous modeling and model-informed feature selection analysis, and then test it extensively on both benchmark batch and continuous chemical processes. To this end, Support Vector Machines are inherited since they can (i) capture nonlinearity among process variables, (ii) handle high-dimensionality of data, and (iii) be used for simultaneous modeling and feature selection due to their succinct formulation as convex nonlinear optimization problem, where binary variables are introduced in the objective function for each variable (128).

The main goal is to design an online decision support tool for data-driven process monitoring which (i) enables rapid and accurate fault detection/identification and diagnosis, (ii) produces precise fault reconstruction to facilitate recovery, and prevents propagation of the detected fault, (iii) provides maintenance strategies to process operators while maximizing profit. Rapid and accurate fault detection and diagnosis is aimed to be achieved and validated within first two objectives. Under the second objective, we aim to (i) implement the framework within a benchmark simulation to validate online process monitoring performance, and (ii) employ advanced regression analysis to provide information on the detected fault's magnitude and direction, where different advanced data mining algorithms, including Support Vector Regression along with model-informed feature selection, are tested to find the most precise model.

Third objective is the integration of different maintenance tasks to the proposed framework in order to optimize the process efficiency and profit, known as maintenance optimization. Maintenance optimization is an essential component of industry operations to minimize process downtime, and maximize profit while sustaining safety and sustainability. In this dissertation, corrective and periodic preventive maintenance strategies are integrated to the proposed framework. The premise is that the integration of maintenance strategies to the framework minimizes the operation cost by simultaneously monitoring the process for early detection of abnormalities and improving the equipment health, which is crucial to achieve targeted end-product quality.

Final objective is to propose a novel corrective maintenance strategy that can eliminate process downtime, maximize process reliability and profit by integrating multi-parametric model predictive control to the proposed framework for fault detection, diagnosis and reconstruction. The traditional corrective maintenance is online re-tuning of existing controllers which causes a significant time spent under faulty condition, which may extend the total operation time to achieve targeted end-product quality or even further deteriorate it. Although switching based active fault-tolerant control strategies have been introduced in the literature, which aims to minimize the process downtime by storing pre-calculated control laws, the major challenge has remained to have a reliable and robust FDI system which can provide accurate fault information and minimize the number of false-alarms. Therefore, in order to minimize process time spent under abnormal condition while increasing process reliability and resilience via accurate fault detection and diagnosis, we aim to introduce a novel *parametric fault-tolerant control* strategy. The goal is to design a multi-parametric model predictive controller for both normal and faulty operations, where the control strategies are affine functions of the system states and the magnitude of the detected fault. The premise is that integration of the proposed framework and parametric fault-tolerant control ensures increased process resilience and eliminates process downtime by enabling rapid switches between *a priori* mapped control action strategies.

1.3 Data Analytics for Improving Environmental Health Decision-Making

Environmental emergencies and disasters are inevitable in life, and the risks they thread to health and the environment due to ensuing chemical contamination is evident. In order to reduce these risks, the scientific community and stakeholders in policy and administration need high-performance models and tools that can allow them to make rapid decisions to decrease the adverse effects of chemical exposure during environmental emergencies. In fact, over the last decade both academic and applied public health communities demand increase in post-disaster research activities (129) that can significantly mitigate the adverse effects caused by these events. Therefore, in this section, characterization of environmental and health risks of unknown environmental contaminants through computational models/tools that use biomedical and chemical big data is addressed and presented.

1.3.1 Motivation and Background

Environmental emergencies and disasters (i.e. hurricanes, water-flooding) pose significant challenges on human health and environment. The environmental mobilization of contaminants by natural disasters cause chemical contamination (i.e. re-distribution of hazardous chemical substances) which increases the post-disaster chemical exposure risks. For instance, in 2005, hurricane Katrina caused serious damage to land due to the wind, storm surge and levee breach. The breakdown of the sewage systems caused distribution of contaminants all over the city leaving New Orleans residents in a toxic environment (130). Studies are still ongoing to identify exposed chemicals, understand their post-effect and accordingly develop effective solutions (131; 132; 133). Frequent water flooding disasters happening in Colorado (in 2011, 2013, 2015) regularly mobilize mine tailings and cause them to intermix with sediments, asphalt from roads, distributed oils, etc. In 2015, because of the discharge of such complex chemical mixtures into the Colorado River, the color of the river was turned into orange for about a 100 miles (134). More recently, hurricane Harvey hit Texas coasts in September 2017 and lead to potential environmental hazards resulting from chemical plant explosions, flooding of San Jacinto River, a contaminated site, and toxic emissions from a refinery located in the Manchester neighborhood (129). These examples, covering only a very small fraction of the entire disaster cases, reveal the importance and need for post-disaster research activities. Additionally, climate change and domestic economic shifts further exacerbate the exposure levels, and elevate risks towards human health (134; 135). Therefore, precise and rapid examination of the complexity of the hazardous chemical exposures is essential to identify the potential adverse environment and health impacts, and subsequently to provide immediate solutions and/or prevent further catastrophic events during environmental emergencies. The advances in science and technology, computational power enable researchers to collect, generate and utilize large amounts of biomedical, chemical and environmental data, which can be further used to develop efficient tools and models to create a decision support system during environmental emergencies.

The rapid advent of high throughput and/or "omics" sequencing technologies yield major amounts of high dimensional biomedical and analytical chemistry data, referred as "big" data. Big data have enabled researchers to obtain overwhelming amount of information about genomic profile (8), patient's disease history (136), environmental monitoring (137), and chemical finger-

printing (138), which creates great opportunity for solving challenges related to environmental health. Therefore, an immediate and key question is how to use the advantage of the data abundance and extract knowledge for generating useful models/tools that can be employed as decision support systems. With the increase in today's available computer power, big data analytics and statistical tools can be of great assistance.

In order to support university-based multidisciplinary research on human health and environmental issues related to hazardous substances and pollutants, the National Institute of Environmental Health and Sciences (NIEHS) has established Superfund Research Program (SRP). The ultimate goal is to decipher the relation between chemical exposure and disease. Specifically, Texas A&M University Superfund Research Program (TAMU SRP) the aim is to develop comprehensive tools and models for addressing exposure to unknown chemical mixtures, and accordingly design solutions for the community during environmental emergency-related contamination events to mitigate the adverse effects on human health and the environment (TAMU Superfund Research Center, 2018). Understanding the complexity of the hazardous chemical exposures is crucial, and necessary in order to mitigate the adverse impacts on human and environmental health, especially during environmental emergency-related contamination events (i.e. hurricanes). However, characterization of the potential risks from exposures to hazardous complex substances and mixtures is compounded by their inherent chemical complexity and variability in composition between manufacturers or even batches. This complicates the detailed chemical characterization of these substances, which in turn obfuscates their grouping for human and environmental health assessment. Environmental health-based risk assessment of such substances, referred as unknown or variable composition, complex reaction products, or biological materials (UVCBs), is one of the most challenging tasks in regulatory toxicology (139). Products of petroleum refining are prototypical UVCB (Unknown or Variable composition, Complex reaction products and Biological materials) substances. UVCBs are some of the most challenging substances for the industry and regulators because there are few established frameworks for how to evaluate UVCBs under current chemical regulatory policy and ensuring that there is no underestimation of hazard to either workers or the general users of the end-products (140). Indeed, the complexity of the chemical composition of petroleum substances, in particular their multi-constituent nature and variability in product composition based on the variability in crude oil stocks, poses unique challenges to the

regulators and registrants of these substances.

Typically, the individual substances are grouped into product categories based on the similarities in manufacturing processes, phys/chem properties (including refining history and boiling point/carbon number ranges), and limited analytical chemical information (such as hydrocarbon classes) (140; 141). However, such broad similarity parameters may not always be considered sufficient and new approaches to facilitate grouping of UVCBs are needed. Indeed, recent developments in high-resolution and multi-dimensional analytical techniques have greatly improved the depth of chemical characterization of complex substances (142; 143). Despite these advances, full chemical characterization of complex substances, such as a petroleum UVCB substances, is still largely unattainable (144). This presents a challenge for defining "sufficient similarity" for a substance of interest in comparison to those substances that may have already been tested for their potential human and ecological effects (145; 146).

A variety of analytical methods that can be used to rapidly profile chemical composition of environmental samples and UVCBs produce complex high-dimensional data sets (144; 147). Quantitative interpretation of these high-dimensional data has been an active area of statistics and a number of algorithms have been applied to classify unknown samples, or to derive discriminating data features (143; 148). For example, data integration, clustering and visualization techniques using ion mobility-mass spectrometry (IM-MS) data of a subset of UVCBs was used to determine the group-specific similarities (146). Comparative analyses have also been performed. For example, de Carvalho Rocha et al. (2) utilized principal components analysis (MPCA), principal factors analysis (PARAFAC), and self-organizing map (SOM) analysis to differentiate among various types of fuels via pattern recognition. Although SOM produces visually appealing grouping maps, comparative assessment to determine the optimal grouping is a challenge (149; 150). Additional pattern recognition analysis techniques (141; 151) have been explored to interpret the grouping information of complex substances; however, the outcomes or these methods are largely qualitative in nature and rely on the subjective visual evaluation of the grouping outcomes rather than quantitative comparative metrics. To tackle this problem and delineate optimal grouping of complex substances by evaluating different grouping structures via quantitative metrics and visual factors, data-driven modeling and dimensionality techniques can be of immense help. In particular, data analysis, modeling and dimensionality reduction techniques can provide immense guidance on ex-

perimental design and decision-making by using biomedical and environmental high dimensional data sets. Hence, as part of the Data Science Core of Texas A&M Superfund Research Program, data analytics models/tools are needed to be designed and developed to group unknown environmental contaminants and mixtures with known chemicals to facilitate read-across. Finding the optimal grouping is of utmost importance for the development of novel broad-acting, high-capacity sorbents, enterosorbents, where the aim is to implement them in diets to reduce the bioavailability of the complex mixtures and substances.

1.3.2 Objectives

Detailed chemical characterization of a chemical mixture is challenging due to the variation in chemical composition during environmental emergencies. To provide rapid solutions, grouping an unknown chemical mixture to a group of well-studied, "known", chemicals is critical. With the aim of creating accurate and effective decision support tool for regulators, the first objective is to design a framework that can optimally group unknown chemical mixtures with the known chemicals by using multi-dimensional analytical chemistry and bioactivity profile data extracted from complex substances. To do this, the aim is to develop a data-driven framework that includes two separate workflows: (i) unsupervised data analysis workflow, (ii) supervised analysis workflow for building classification models based on manufacturing characteristics of complex substances. While the perspective of unsupervised analysis workflow is to understand the biological and chemical fingerprint of complex substances, the idea of building classification models based on manufacturing characteristics is built upon read-across hypothesis. This hypothesis advocates that similar complex substances that are grouped together according to their phys/chem properties may have similar effects. Hence, once an unknown chemical mixture is grouped into a cluster of known chemicals, read-across between cluster members would bridge the gap between data-poor and data-rich chemical substances.

Furthermore, finding the optimal grouping of complex substances and providing quantitative and visual communication of the grouping results is of utmost importance and interest for environmental health regulators and decision-makers. Therefore, the second objective is to bridge the gap between the optimal grouping of complex substances and the quantitative evaluation and communication of the grouping outcomes. To do this, unsupervised grouping is performed via hierarchical

clustering, where results can be demonstrated through dendrograms and grouping quality is evaluated against existing manufacturing classes using Fowlkes-Mallows index. Second, supervised analysis workflow for classification of complex substances based on their manufacturing characteristics is targeted to be evaluated based on accuracy and confusion matrices. Although the common approach in the literature is qualitative assessment based on grouping visuals, the introduction of quantitative metrics into the proposed framework are expected to improve comparative assessment between different grouping structures, which can then immensely facilitate the interpretation of results.

The overall premise of the proposed framework is to provide (i) optimal grouping of complex substances, (ii) improved interpretation of the grouping results for decision-makers with the use of visualization techniques and identification of the most informative features, and (iii) comparative assessment of the grouping results by reporting quantitative metrics (i.e. Fowlkes-Mallows index for clustering, and accuracy for classification analysis). The optimal grouping information of complex substances via the proposed framework is expected to facilitate decision-making on sorbent material design as well, which are aimed to be implemented in diets to reduce the bioavailability of chemical mixtures during environmental emergencies.

1.4 Dissertation Objectives and Structure

The objectives of this dissertation are summarized below.

1. To develop a data-driven framework that enables simultaneous fault detection and diagnosis in order to minimize the process runtime spent under faulty condition, and expedite the corrective actions, thus to ensure high safety and profitability.

2. To extend the developed framework to enable simultaneous fault identification and diagnosis in order to decrease the number of models used during online process monitoring.

3. To extensively test and validate the performance of the developed framework for both batch and continuous processes.

4. To integrate major maintenance optimization strategies with the developed framework to increase process efficiency and profit.

5. To introduce a novel corrective maintenance strategy, parametric fault-tolerant control, to eliminate process downtime, and maximize process reliability and profit by merging the developed framework and multi-parametric model predictive controller design.

6. To establish a data-driven framework for grouping of unknown complex substances with the known chemicals in order to enable read-across during environmental emergency-related contamination events (i.e. hurricanes).

7. To advance the communication of the complex substance grouping results via quantitative metrics and visual techniques.

Table 1.1 outlines the dissertation structure and maps the objectives listed above to the corresponding dissertation chapters.

Table 1.1: Dissertation structure.

Objective Number	Dissertation Chapter
1, 2	Chapter 2
3	Chapter 3 and Chapter 4
4, 5	Chapter 5
6, 7	Chapter 6

2. SIMULTANEOUS FAULT DETECTION AND DIAGNOSIS (S-FDD) FRAMEWORK*

In this chapter, we present the s-FDD framework for simultaneous fault detection and diagnosis. The developed framework advances the data-driven process monitoring by producing highly accurate fault detection models with optimal process variables, which are further used in the diagnosis of the detected fault. s-FDD is a data-driven framework which utilizes a custom nonlinear (kernel-dependent) SVM-based feature selection algorithm. The methodology that enables simultaneous modeling and feature selection is derived from the sensitivity analysis of the dual SVM objective and utilizes a greedy algorithm to rank features which guides fault diagnosis. Therefore, the trained end-models use the optimal features, process variables, for fault detection, where the selected features are used for instantaneous fault diagnosis. As a result, s-FDD framework minimizes process time spent under faulty operation, and accordingly ensures high process safety and profitability. Below, we provide background on Support Vector Machines in Section 2.1. The details of the nonlinear SVM-based feature selection algorithm used in the development of the s-FDD framework are presented in Section 2.2 (128). Then, Section 4.2 introduces the s-FDD framework for simultaneous fault detection/identification and diagnosis during process monitoring of chemical processes.

2.1 Support Vector Machines

Support Vector Machines (SVM) is a widely-used machine learning algorithm that has produced significantly successful results in extensive set of supervised learning problems (i.e classification, regression, and outlier detection) from various fields. Specifically, the main idea behind SVM classification is to transform training data into a higher dimension via nonlinear Kernel functions, where a linear hyperplane in the mapped space (nonlinear in the original domain) can separate the data by class. SVMs are formulated as convex optimization problems which enable them to be solved to global optimality. Therefore, with an appropriate nonlinear mapping to a

*Part of this chapter is reprinted with permission from "Big data approach to batch process monitoring: Simultaneous fault detection and diagnosis using nonlinear support vector machine-based feature selection" by Onel, M. and Kieslich, C.A. and Pistikopoulos, E.N., 2018, *AIChE Journal*, Vol. 65, Issue 3, pp 992-1005, John Wiley Sons [2018] by John Wiley and Sons and Copyright Clearance Center and "A nonlinear support vector machine-based feature selection approach for fault detection and diagnosis: Application to the Tennessee Eastman process" by Onel, M. and Kieslich, C.A. and Guzman, Y.A. and Floudas, C.A. and Pistikopoulos, E.N., 2018, *Computers & Chemical Engineering*, Vol. 116, pp 503-520, Elsevier [2019] by Elsevier and Copyright Clearance Center

sufficiently high dimension, data from two classes can always be separated optimally (Figure 2.1). Although training SVM models is computationally demanding, SVMs produce highly accurate models due to their ability to resolve complex nonlinear decision boundaries with globally optimum parameters. They are also less prone to most data-driven modeling pitfalls (e.g. over-fitting, multicollinearity), which make them popular among myriad of research fields.

In this study, fault detection is formulated as a supervised learning problem (i.e. classification) with l training instances correspond to batches, where $x_i \in \mathbb{R}^n$. Indices $i, j = 1, 2, \ldots, l$ belong to batches, whereas indices $k, k' = 1, 2, \ldots, n$ correspond to input process data features (i.e. process measurements at specific time points). To determine whether an ongoing batch is faulty or normal, we utilize the C-parameterized SVM (C-SVM) classification formulation with nonlinear Kernel functions. The basic C-SVM formulation with hinge loss, ℓ_2-norm penalty, and linear kernel (152; 153) is written as:

$$\begin{aligned} \min_{w,b,\xi} \quad & \frac{1}{2}\|w\|_2^2 + C\sum_{i=1}^{l}\xi_i \\ \text{s.t.} \quad & y_i(w \cdot x_i + b) \geq 1 - \xi_i \quad i = 1, \ldots, l \\ & \xi_i \geq 0 \qquad\qquad\qquad\quad i = 1, \ldots, l \end{aligned} \tag{2.1}$$

where w is the weight vector of features (process measurements). ξ_i are slack variables for each instance (i.e. batch) i that are misclassified. C is a regularization parameter and controls the level of training error to be introduced in the cost function for the sake of creating more generalizable models. $y_i \in \{-1, 1\}$ denotes the group label of batch i, as normal or faulty, respectively. Finally, b represents the bias parameter. When model 2.1 is solved to global optimality, resulted optimal solution (w^*, b^*, ξ^*) yields linear decision function (w^*, b^*, ξ^*) whose sign predicts the group membership of the new batch x:

$$w^* = \sum_{i=1}^{l} \alpha_i^* y_i x_i \tag{2.2}$$

$$f(x) = w^* \cdot x + b^* \tag{2.3}$$

where α_i are dual variables. Here, due to the nonlinear nature of batch process data, C-SVM

with linear kernel may not provide accurate and robust solutions for batch process monitoring. Therefore, we exploit nonlinear Kernel functions, $K(x_i, x_j)$, within the C-SVM formulation which provides us to train nonlinear decision functions in the input (original) space which implicitly map the data to a different, possibly infinite dimensional feature space where a linear decision function can separate the mapped data (154). This is also known as Kernel trick (Figure 2.1). Kernel functions are introduced in the Lagrange dual formulation of model 1, which is written as:

$$\begin{aligned} \max_{\alpha} \quad & \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \\ \text{s.t.} \quad & \sum_{i=1}^{l} \alpha_i y_i = 0 \\ & \alpha_i \in [0, C] \qquad i = 1, \ldots, l \end{aligned} \tag{2.4}$$

The resulting linear decision function consisting the optimal dual solution α^* becomes:

$$\begin{aligned} f(x) &= w^* \cdot \phi(x) + b^* \\ &= \sum_{i=1}^{l} \alpha_i^* y_i K(x_i, x) + b^* \end{aligned}$$

where $\phi(x)$ is the function providing Kernel-induced implicit mapping. Here, w^* may no longer belong to $\mathbb{R}^n$ and is possibly of infinite dimension.

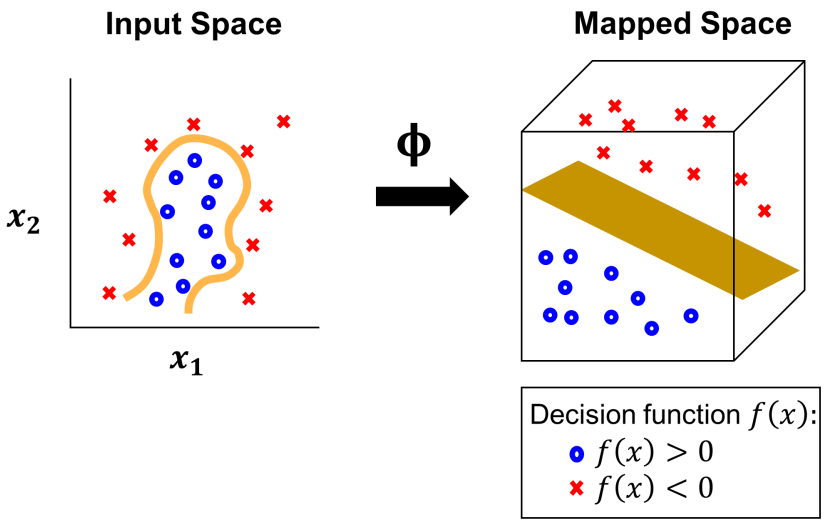


Figure 2.1: Nonlinear Kernel-induced implicit mapping to higher dimensional space in SVM modeling.

The interested reader in the derivation of the Lagrange dual problem and resulting decision functions can refer to the key references(153; 152; 154).

2.2 Nonlinear Feature Selection Algorithm Based on Support Vector Machines

We introduce the feature selection algorithm based on nonlinear Support Vector Machines which is formulated to attain the most descriptive original process measurements (128). Elimination of redundant measurements is highly valuable for high-performance model development for batch process monitoring. It significantly reduces probability of over-fitting in the built data-driven models detecting faults, where the size of unfolded and time-evolving batch process data grows significantly. Furthermore, performing dimensionality reduction while protecting the original feature space is highly valuable for rigorous fault diagnosis, where the selected top descriptive features yield the major causes of the detected fault. Below, we present brief theoretical background of the adopted feature selection algorithm which is based on nonlinear Support Vector Machines.

In order to perform model-informed feature selection, we introduce binary variables $z \in \{0,1\}^n$ in the Lagrange dual formulation with nonlinear Kernel functions (model 2.4), where $z_k = 1$ corresponds to the selection of feature k as being one of features in the optimal subset and $z_k = 0$

corresponds to the elimination of feature k. The resulting model becomes:

$$\begin{aligned}
\min_{z}\max_{\alpha} \quad & \sum_{i=1}^{l}\alpha_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\alpha_i\alpha_j y_i y_j K(x_i\circ z, x_j\circ z) \\
\text{s.t.} \quad & \sum_{i=1}^{l}\alpha_i y_i = 0 \\
& \alpha_i \in [0,C] \qquad i=1,\ldots,l \\
& \sum_{k} z_k = m \\
& z_k \in \{0,1\} \qquad k=1,\ldots,n
\end{aligned} \tag{2.5}$$

where m represents the size of the optimally reduced subset of input features and operator $\circ$ is the Hadamard product operator for component-wise multiplication.

Model 2.5 delineates the explicit formulation of the feature selection problem via nonlinear Support Vector Machines, which results in a challenging bi-level problem. Solving model 2.5 to global optimality is highly challenging and impractical in real-life applications (128), hence we propose to utilize sensitivity of the objective function provided in model 2.5 with respect to z_k at $(\alpha^*; z)$, where z_k is treated as a fixed parameter. In order to attain the first-order sensitivity of the objective function of the model 2.5 at an optimal solution with respect to the parameter z_k, which is located in the objective function and constraints, we use the partial derivative of the Lagrange function of the model 2.5 (155; 156; 157):

$$\begin{aligned}
&\frac{\partial}{\partial z_k}\left[\sum_{i=1}^{l}\alpha_i^* - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\alpha_i^*\alpha_j^* y_i y_j K(x_i\circ z, x_j\circ z)\right. \\
&\qquad \left. +\lambda\left(\sum_{i=1}^{l}\alpha_i^* y_i\right) - \sum_{i=1}^{l}\mu_i^{(1)}\alpha_i^* + \sum_{i=1}^{l}\mu_i^{(2)}\left(\alpha_i^* - C\right)\right]\Bigg|_{z=z^*} \\
&= -\frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\alpha_i^*\alpha_j^* y_i y_j \left.\frac{\partial K(x_i\circ z, x_j\circ z)}{\partial z_k}\right|_{z=z^*}
\end{aligned}$$

where $\lambda\in\mathbb{R}, \mu^{(1)}, \mu^{(2)} \in [0,\infty)^n$ are Lagrange multipliers.

Lagrangian sensitivity allows us to guide the perturbations on element z_k. The formulation yields the perturbation criterion for feature k as follows:

$$\text{crit}_k = -\frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\alpha_i^*\alpha_j^* y_i y_j \left.\frac{\partial K(x_i \circ z, x_j \circ z)}{\partial z_k}\right|_{z=1} \tag{2.6}$$

$$k_{\text{worst}} = \arg\max_k \text{crit}_k \tag{2.7}$$

Using the criterion given in model 2.6, we eliminate features one at a time (feature worsening the model 2.5 objective the most, k_{worst}) as we build nonlinear C-SVM models. This yields a greedy reductive algorithm for model-informed feature ranking. The iterative procedure allows us to perform simultaneous modeling and model-informed feature elimination via C-SVMs, where in each iteration, the feature worsening the Lagrange function of model 2.5 the most is eliminated. Here, one can also eliminate features in blocks (i.e as % of total features). The presented feature selection algorithm based on nonlinear SVMs is summarized in Figure 2.2.

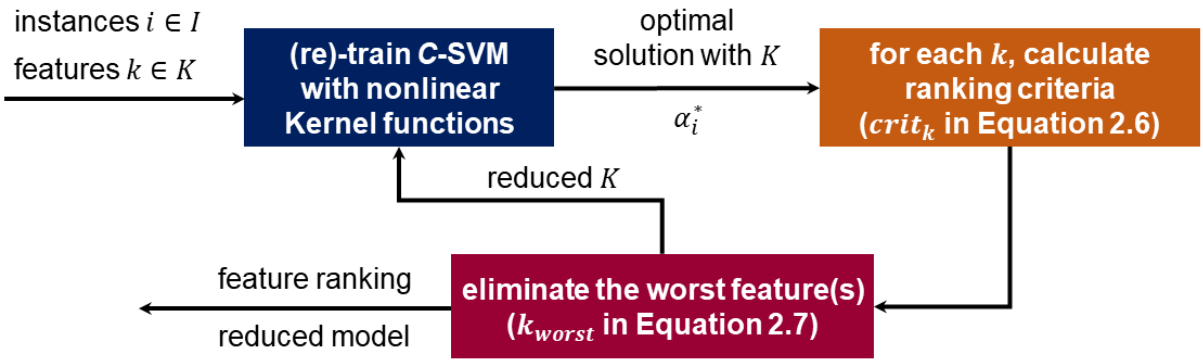


Figure 2.2: Greedy reductive algorithm for simultaneous modeling and model-informed feature selection.

Of note, the algorithm provided here is equivalent to recursive feature elimination (RFE)-SVM classification algorithm(158) when performing linear classification. The algorithm has been implemented in C++/Python environment using the LibSVM library(159). The presented feature selection algorithm has been applied in numerous fields including bioinformatics, energy, environmental health. Specifically, we have achieved profoundly accurate predictions in HIV-1 co-receptor

usage(160) and protein structure prediction(161). This dissertation presents the applications in energy and environmental health.

2.3 Simultaneous Fault Detection and Diagnosis (s-FDD) Framework

The feature selection algorithm based on nonlinear SVM presented in Section 2.2 allows simultaneous modeling and feature selection. This enables optimal data-driven modeling by selecting the most descriptive features of the studied data set. The data-driven modeling applications are becoming more prevalent everyday as the rwal-time data collection expedites. This increases the importance of having efficient feature selection techniques that do not hinder the interpretation. The Smart Manufacturing and Industry 4.0 initiatives have lead industry to adopt data-driven techniques in numerous areas, where process monitoring is one of the major ones. Therefore, in this dissertation, we have developed simultaneous fault detection and diagnosis (s-FDD) framework by benefiting from the simultaneous modeling and feature selection algorithm presented in Section 2.2. The framework constitutes from four main steps: (i) data pre-processing, (ii) initial parameter tuning, (iii) simultaneous modeling and feature selection to obtain feature ranking, and (iv) building the final model with the optimal feature subset. The details of each steps are provided in Section 2.3.1, where we construct binary classification models for fault detection. Section 2.3.2 describes the extension of the framework for fault identification, where multi-class classification analysis is performed. The applications of the s-FDD framework to the selected benchmark continuous and batch processes are presented in Chapter 3 and 4.

2.3.1 Binary Classification for Fault Detection

For fault-specific model development, two-class classification models using *C*-SVM formulation have been built, which require data from both normal and each faulty operation separately. Although process data belongs to specific fault may not be readily available in industrial process, fault-specific process data can be easily simulated with the dynamic model of the studied process.

2.3.1.1 Data Pre-processing

This is the initial step of each data-driven analysis. Data needs to be formatted, cleaned and standardized prior to training in order to acquire accurate and robust models. Data pre-processing steps of the s-FDD framework include (i) data formatting, (ii) missing data handling, (iii) outlier

removal, (iv) attribute construction (optional), (v) data scaling, and (vi) *a priori* elimination of redundant features (a.k.a. data reduction).

Data Formatting: The source of the input process data to the s-FDD framework can be either historical industrial data or simulated process data. In either case, formatting of the data is necessary and important. The input data to the classification analysis needs to be 2-dimensional (2D), where the rows belong to the batch IDs for batch process monitoring, and operation IDs for continuous process monitoring. The features can be solely process variable measurements (i.e. continuous operations) or time-specific process variable measurements (i.e. batch/fed-batch operations). In particular, batch process data is 3D (i.e. batch X process variable measurements X time), which needs to be further unfolded into 2D prior to modeling steps. Unfolding can be achieved via three ways for a 3D process data. Specifically for batch process data, unfolding can be done via: (i) batch-wise, where instances become the batch IDs and features are time-specific measurements, (ii) measurement-wise, where instances become process variable IDs and features are time-specific batch IDs, and finally (iii) time-wise, where instances become the sampling points and features are batch-specific process variable measurements (162). In general, we prefer batch-wise unfolding for batch process monitoring applications of the s-FDD framework for the sake of simplicity during the online implementation of the models. When we train the classification models via batch-wise unfolded 2D data, we inquire time-specific measurements of each batch operating during the online phase to generate a binary answer for indicating the fault occurrence.

Missing Data Handling: Missing data can be handled via traditional (*ad-hoc*) and advanced machine learning methods. *Ad-hoc* techniques are the easiest, fastest solutions for handling missing data, therefore they are widely used in literature (163). However; despite being fast, ad-hoc techniques are claimed to induce biased results (164). *Ad-hoc* methodologies include (i) deletion of the column or row of the missing data location (a.k.a column or row-wise deletion) and performing analysis with the available data, (ii) replacement of missing data with zero or random variable, and (iii) missing data imputation with column or row mean or median values. Due to the ubiquitous nature of missing data in several different variables of datasets, we may have to exclude substantial fraction of the original data set with deletion techniques. Thus, in order to avoid sacrificing from the data set size and further lose any informative features for modeling, imputation or replacement techniques are preferred over deletion techniques. Moreover, the advanced machine

learning techniques for missing data imputation are becoming prevalent (165). One of the common techniques that fall under the advanced methods category is K-nearest neighbor (KNN) based imputation (166), where pattern recognition analysis is performed on the data in order to fill the missing data without hindering the existing data pattern. Within s-FDD framework, we propose to adopt one of the advanced imputation techniques if data access is limited. If not, we suggest to adopt row-wise deletion techniques, where row corresponds to batch or continuous operation ID.

Outlier Removal: Removing outliers is one of the data cleaning steps. In this dissertation, we apply the s-FDD framework to the simulated process data sets which are free of outliers, therefore we have not applied outlier removal analysis. However, outlier data points can be widely observed in industrial historical data sets. In such cases, z-score threshold based techniques can be adopted for outlier removal (167).

Attribute Construction: Attribute construction is the extraction of additional process descriptors to include in the process data set in order to help the data mining process. Specifically, this step may improve the classification model performance by including more features that can capture the nonlinear dynamics of the chemical processes. This step is mostly used in batch process monitoring, where inherent non-stationarity and batch-to-batch variability further obfuscate the data mining process.

Data Scaling: Process data sets are scaled via standard score (z-score) calculation as shown below:

$$z = \frac{X - \mu}{\sigma} \tag{2.8}$$

where X is a data point from a selected attribute, and μ is the mean, and σ is the standard deviation of that corresponding attribute (i.e. feature).

Data Reduction: Final pre-processing step of the s-FDD framework is to eliminate redundant features that have less than 10^{-8} standard deviation.

2.3.1.2 Parameter Tuning

Within the s-FDD framework, we train C-SVM (two-class) classification models with the Gaussian radial basis function (RBF) as the nonlinear Kernel function. This step is essential in order to prevent any bias during modeling, and further improve the model performance. There are two

parameters that we need to tune: (i) C (cost) parameter of C-SVM, and (ii) γ parameter of the Gaussian Radial Basis kernel function. C parameter serves as a regularization parameter which controls the trade-off between training and testing error. Low training error indicates higher model complexity, thus low generalizability. This means that we have built a model that is too specific to the training data set, however that model performs poorly for the unseen data. Whereas low testing error comes with the lower model complexity, yet higher training error, thus less accurate results. Therefore, we need to find a balance between model complexity and model generalization, which can be achieved by tuning the C parameter. The second parameter that we tune is the γ of the Gaussian RBF. This parameter can be interpreted as the inverse of the influence radius of the samples selected as support vectors by the model ().

The default value for the Gaussian RBF kernel parameter, γ, is $1/n$, where n is the number of features, in LIBSVM (159). Therefore, we tune parameter $\hat{\gamma}$ where the relation between $\hat{\gamma}$ and γ is:

$$\gamma = \frac{2^{\hat{\gamma}}}{n}. \tag{2.9}$$

Additionally, parameter $\hat{C}$ is tuned, where

$$C = 2^{\hat{C}}. \tag{2.10}$$

The parameter tuning step is performed via grid search with the entire set of features of the process data set. However, during the iterative feature selection algorithm, described in Section 2.2 of Chapter 2, we can tune $\hat{\gamma}$ after every feature elimination step with the remaining feature subset:

$$\gamma = \frac{2^{\hat{\gamma}}}{z^{\top}\mathbf{1}} \tag{2.11}$$

2.3.1.3 Feature Ranking and Modeling

Once the optimal parameters are obtained, next step is to produce the feature ranking by applying the simultaneous model-informed feature selection and modeling algorithm introduced in Section 2.2. This procedure is iterative. Initially, we train a C-SVM model with the Gaussian RBF kernel by using the complete set of process data set. This means that we use the entire features (pro-

cess measurements during continuous process monitoring and time-specific process measurements during batch process monitoring). Then, at each iteration, we eliminate the feature that yields the lowest Lagrangian sensitivity of the dual objective function of the built *C*-SVM model (objective function of model 2.5) with respect to the feature subset size. The criteria is given in Equation 2.6. We eliminate the features one by one until by training *C*-SVM model in each iteration with the remaining set of features. This iterative procedure produces a feature ranking list. Note that model training is performed via cross-validation, which means that for each fold we obtain an individual feature ranking list. For instance, when training is performed via 10-fold cross-validation, we produce 10 individual feature ranking lists. They may not be necessarily highly different from each other, yet exact lists are not obtained due to the randomness of the process data. Therefore, an average feature ranking list is derived by examining the statistical distribution of the feature ranks among the obtained feature rank lists. This is used as the final feature rank list in further steps.

Next, we re-train *C*-SVM models via cross-validation, and eliminate features one by one based on the feature rank list. This is done in order to determine the optimal feature subset via cross-validation. The performance of the trained models are evaluated by examining: accuracy, fault detection rate (a.k.a. recall), area under the curve (AUC), and false alarm rate. These are extracted from the confusion matrix that yield the classification results. Specifically, a confusion matrix is a n X n matrix, where n is the number of classes. It demonstrates the classified group of all data samples (Figure 2.3).

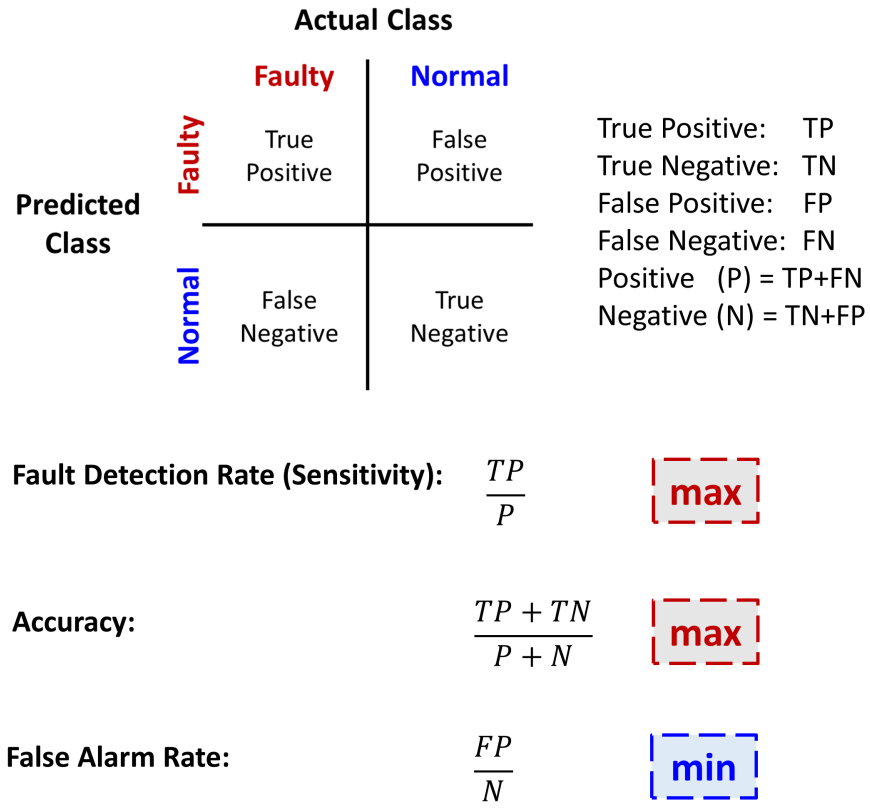


Figure 2.3: Machine learning metrics used for classifier model performance assessment.

Final step is to build the end C-SVM models with the optimal feature subset, and optimal parameters for the online monitoring phase. The models are then used during the online operation to check for fault occurrence. The decision functions generate binary answer, where $+1$ indicates fault occurrence, and -1 indicates normal condition.

2.3.2 Multi-class Classification for Fault Identification

The question needs to be answered here is that what if the operators do not have knowledge on the distinct faulty operation data, yet still need to detect faults within the process. Another question awaiting to be answered is that what happens when there occurs multiple fault types in a process and how can we diagnose them in an effective way. In order to answer these questions, we propose to extend the s-FDD framework for fault identification.

In such cases, one needs to (i) detect whether process is in a faulty condition or not, and (ii) use the s-FDD framework presented in Section 2.3.1 to identify the fault type when a fault is detected. When we have multiple fault types occuring in a process and further multiple normal operation modes, we need to adopt multi-class classification analysis. Multi-class classification can be performed via two ways: (i) one versus one, or (ii) one versus all (Figure 2.4). The first one includes the binary model development for each combination of fault groups, which becomes computationally significantly expensive as the number of the groups increases. On the other hand, the later focuses on the generation of binary classification models to separate one group from the rest at each step. The later approach proposes to build a cascaded approach with sequential two-class classifiers. It is computationally more efficient since we can terminate the model building as we detect the fault type along the hierarchy of separation. Thus, in this work, we are adopting the later approach to decrease the computational expense.

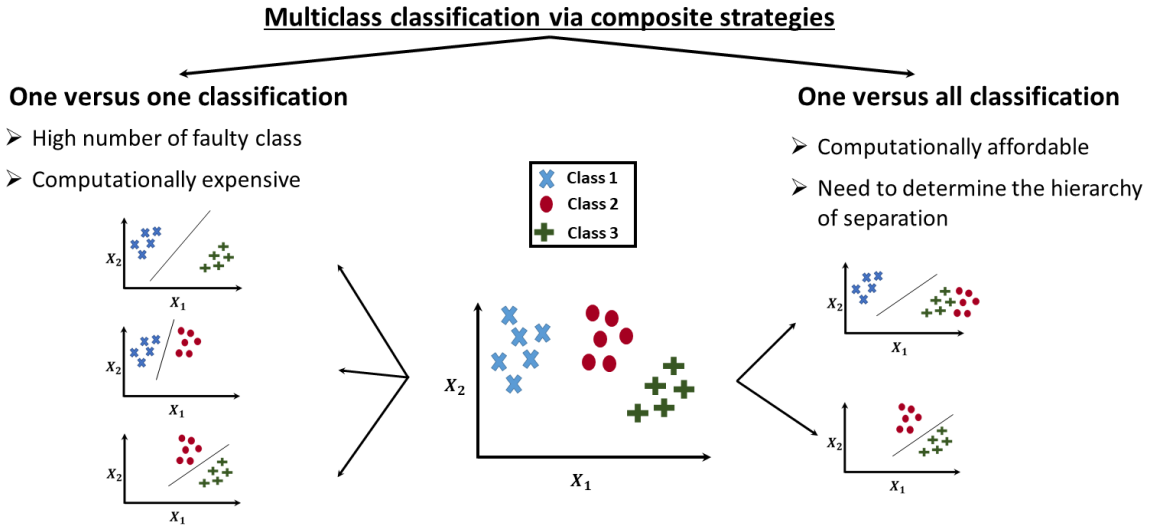


Figure 2.4: Multi-class classification strategies.

In this work, we merge pattern recognition analysis with one versus all binary classification analysis via s-FDD framework. Our goal is to create a decision-tree based classification scheme while benefiting from the model-informed feature selection algorithm presented in Section 2.2. Specifically, we use hierarchical clustering to understand the patterns among the operation data to

characterize the operation conditions. Here, one can use different clustering algorithms (i.e. K-means, hierarchical) to perform pattern recognition analysis. Note that the proposed classification scheme monitors both normal and faulty operation data, where there may exist multiple operation conditions within the normal operation, and multiple fault types under faulty operation. However, the number of the normal operating conditions and the number of the fault types may not be known. In this dissertation, we focus on the later one, which is also known as fault identification. We divide the multi-class classification analysis via s-FDD framework for fault identification into two parts: (i) offline phase, and (ii) online phase analysis (Figure 2.5).

2.3.2.1 Offline Phase: Determining the Hierarchy of Fault Types

During the offline phase, we (i) build two-class *C*-SVM models for fault detection, (ii) calculate the mean process trajectories from the process data and cluster them via hierarchical clustering, and finally (iii) build two-class *C*-SVM models for fault type groups based on the extracted hierarchy. The first step is described in detail in Section 2.3.1, therefore skipped here. The rest of the main steps of the offline phase tasks are summarized below.

2.3.2.1.1 Calculation of the Mean Process Trajectories for Hierarchical Clustering

In order to obtain a hierarchy of fault types, we need to summarize the process data observed among different batch or continuous processes. Therefore, we need to calculate the mean process trajectories, which we refer as the "characteristic snapshot" of the process, to understand the characteristics of the operation. For continuous operation monitoring, we simply take the average of the process variable measurements among the entire continuous operation IDs. For example, let's assume that we have 100 simulations of continuous operation with various fault types, where we do not know the number of fault types. If we measure 15 process variables, this yields us a process data set size of 100 X 15. The characteristic snapshot of this continuous process is the average across the 100 simulations, which is a 15-dimensional vector. On the other hand, calculation of the characteristic snapshot for batch process monitoring slightly differs from the one for continuous process monitoring. In batch processes, first we need to align the historical or simulated batch process operations by adopting one among the dynamic time warping (168; 169; 170), correlation optimized warping (171; 172), indicator variable (173) or curve registration techniques

(174). Secondly, we need to unfold the 3D aligned batch process data into 2D via batch-wise unfolding technique as described in the data pre-processing step of the Section 2.3.1. This yields time-specific process variable measurements as the features. Then, we can take the average of the obtained 2D batch process data to capture the characteristic snapshot of the batch process and further use it for the hierarchical clustering analysis to investigate the fault type patterns.

Hierarchical Clustering: The extracted mean process trajectories are n-dimensional vectors, where n is the number of features of the process data set. In order to use them for the hierarchical clustering analysis, we initially calculate the Euclidean distance matrix, a square matrix that contains the pair-wise Euclidean distances of the features. The obtained square matrices are then used as the similarity matrix for the hierarchical clustering. We perform the clustering analysis by using the "hclust" function of the "stats" package of R statistical software, where we adopt the Ward's method (175) for the linking the clusters. The obtained hierarchical tree demonstrates the structure of fault types and is used during the binary classifier training for identification of the particular fault type.

Two-class C-SVM Model Development for Fault Identification: Here, we build two-class C-SVM models for separating distinct fault types from each other, where the label of the fault types are guided by the hierarchical clustering analysis output. Specifically, we cut the hierarchical tree into two in order to select two groups of fault types. Then, we train two-class C-SVM classifier for separating the selected two fault groups from each other. The separated groups are further cut into two and this is performed iteratively until we build a binary classifier for each pair of fault groups. Here, the binary classifier building steps are identical with the s-FDD framework for fault detection classifier building steps. By following this iterative tree-based classification scheme, we extend the s-FDD framework for fault identification. The obtained classifiers are then implemented for the online phase.

2.3.2.2 Online Phase: Fault Identification

During the online operation, we monitor the process by using the binary classifier decision function built for fault detection. If this model produces a positive response, which indicates the fault occurrence, we use the rest of the binary classifier decision functions in the order of the hierarchical tree. This approach enables a grid search to reveal the fault type while the extracted

optimal set of features reveal its diagnosis.

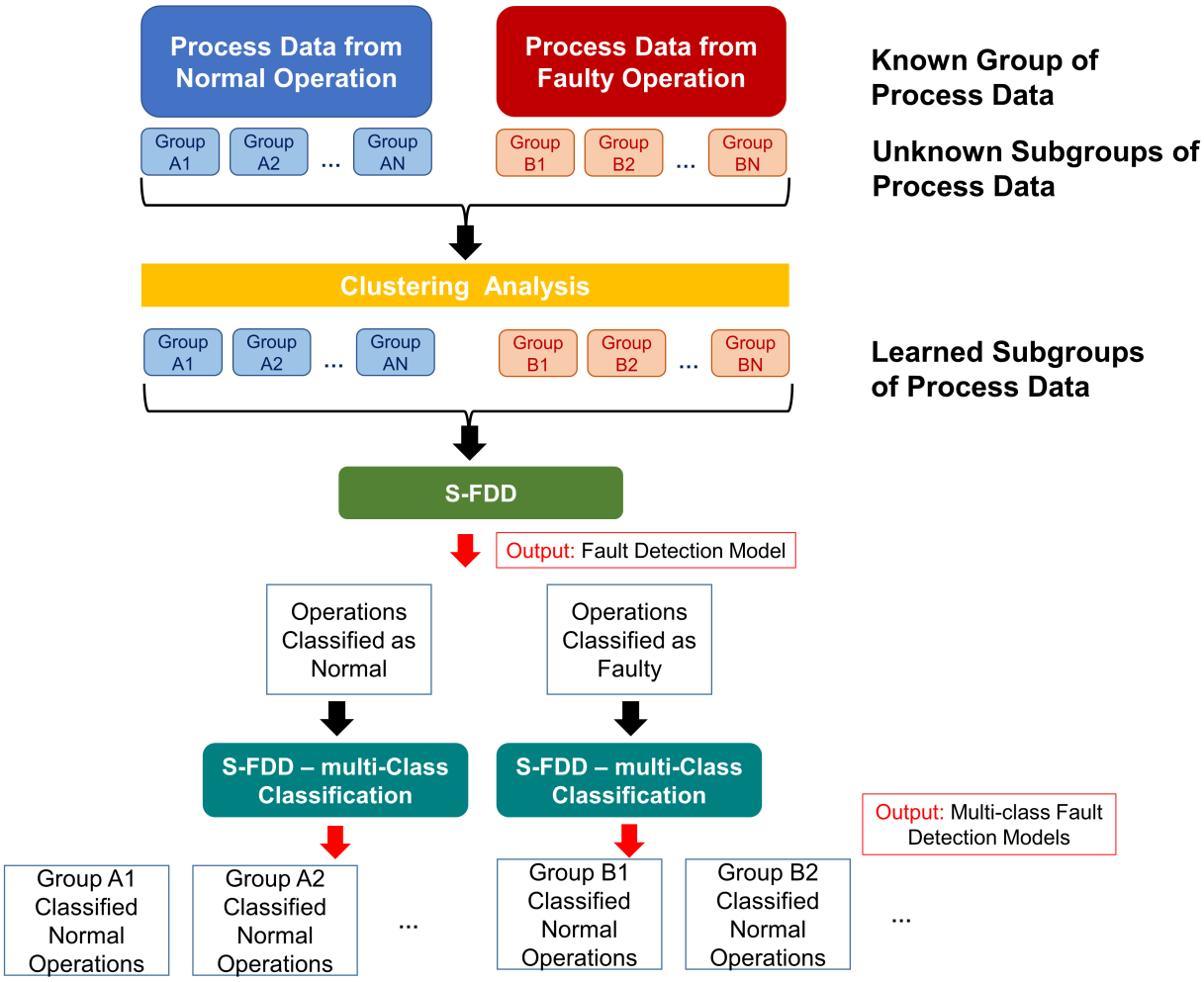


Figure 2.5: Multi-class classification of process data with the s-FDD framework.

3. S-FDD FRAMEWORK FOR CONTINUOUS PROCESS MONITORING*

In this chapter, we present the application of the s-FDD framework to process monitoring and fault detection of continuous processes. Here, we extensively test the performance of the s-FDD framework for (i) fault detection and diagnosis, and (ii) fault identification and diagnosis. Specifically, for the first one, we train fault-specific two-class *C*-SVM models to detect faulty operations, while using the feature selection algorithm to improve the accuracy of the fault detection models and perform fault diagnosis. For the latter one, we perform multi-class classification by training sequential two-class *C*-SVM models for a hierarchy of faulty operations. We present results for the Tennessee Eastman process as a case study and compare our approach to existing approaches for fault detection, diagnosis and identification.

3.1 Tennessee Eastman Process: Model and Data Set

The Tennessee Eastman process (Figure 3.1), an extensively used benchmark case for comparative assessment of process monitoring algorithms, was designed by the Eastman Chemical Company (176). Numerous data-driven fault detection and diagnosis methodologies tested with the Tennessee Eastman process are available in the literature (20; 3; 4; 5). The process is based on a real industrial process, in which the components, kinetics, and operating conditions have been modified for proprietary reasons (20). There are five primary units in the process being: a reactor, condenser, compressor, separator, and a stripper, where chemicals G and H are produced from feedstocks A, C, D and E with byproduct F and inert compound B. The process contains 11 manipulated and 41 measured variables. The detail for the process variables is provided in the Appendix A.

Among several simulation designs, we adopt the one whose plant-wide control structure is provided by Lyman and Georgakis (1). In this study, we use two sets of simulation data set based on the Tennessee Eastman process with the 2^{nd} control structure in Lyman and Georgakis. The first simulation data set is adopted from Chiang and Braatz (20) which includes measurements

*Reprinted with permission from "A nonlinear support vector machine-based feature selection approach for fault detection and diagnosis: Application to the Tennessee Eastman process by Onel, M. and Kieslich, C.A. and Pistikopoulos, E.N., 2019, *AIChE Journal*, Vol. 65, Issue 3, pp 992-1005, John Wiley Sons [2019] by John Wiley and Sons and Copyright Clearance Center

from normal and 21 distinct faulty operations (Table 3.1). It includes single set of simulation for normal and 21 faulty operations separately (yielding 22 simulation data sets). The later is taken from Rieth et. al (177) having measurements from normal and first 20 faults provided in Table 3.1. It involves 500 set of simulations for normal and 20 faulty operations separately (yielding 10500 simulation data sets). In this work, we have randomly selected 2 out of 500 sets of simulations from Rieth et. al data set which lead us to employ a twice size of Chiang et. al data for model building. The aim in using two different simulation data sets with different size is to test the importance of data size involved in model development for fault detection and diagnosis. Training and test sets have been collected by running 25 and 48 hours of simulations respectively, where faults have been introduced 1 and 8 hours into the simulation and each variable is sampled every 3 minutes. Thus training sets consists of 500 samples, whereas test sets contain 960 samples per set of simulation. Further information on the process and simulation can be found in other references (176; 20; 177).

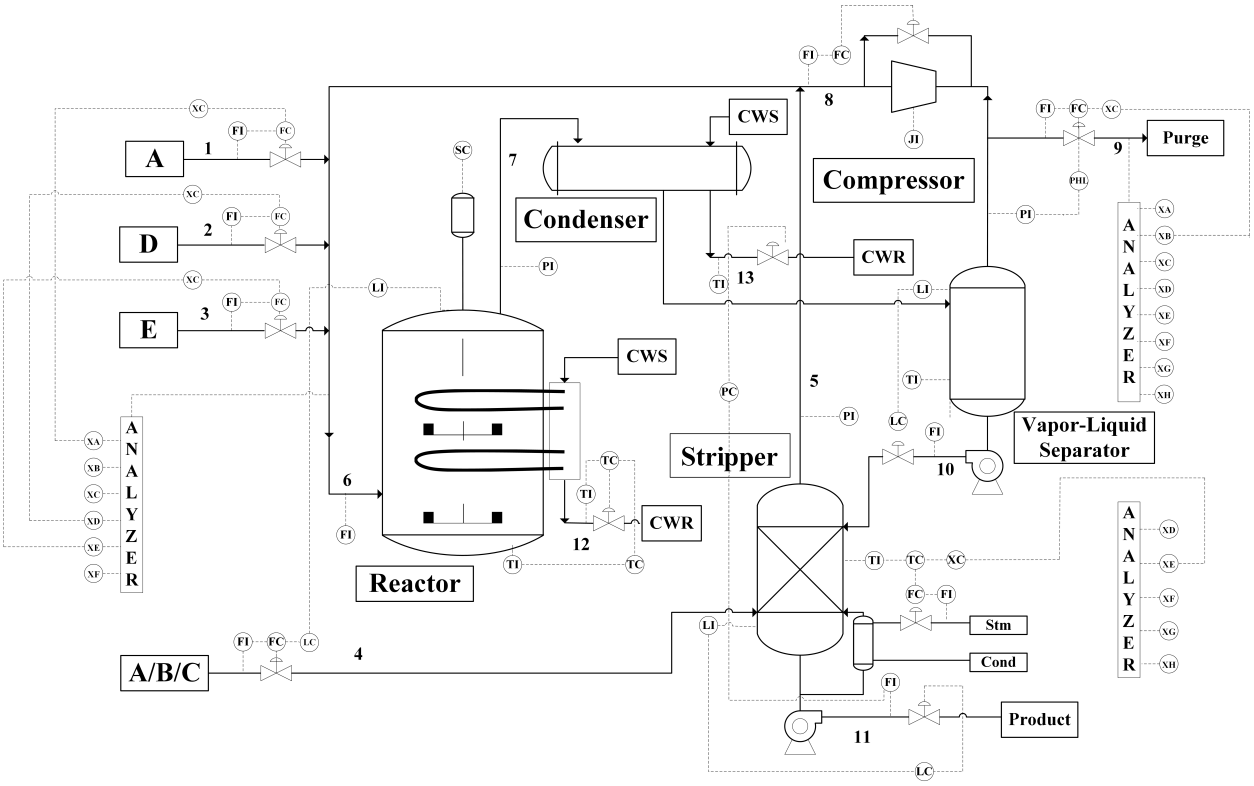


Figure 3.1: Tennessee Eastman process flowsheet with the 2^{nd} control structure in *Lyman and Georgakis* (1).

Table 3.1: Overview of faults and corresponding fault types in the Tennessee Eastman process data set.

Fault No	Fault	Fault Type
1.	A/C Feed Ratio	Step
2.	B Composition	Step
3.	D Feed Temperature	Step
4.	Reactor Cooling Water Inlet Temperature	Step
5.	Condenser Cooling Water Inlet Temperature	Step
6.	A Feed Loss	Step
7.	C Header Pressure Loss	Step
8.	A,B,C Feed Composition	Random Variation
9.	D Feed Temperature	Random Variation
10.	C Feed Temperature	Random Variation
11.	Reactor Cooling Water Inlet Temperature	Random Variation
12.	Condenser Cooling Water Inlet Temperature	Random Variation
13.	Reaction Kinetics	Slow Drift
14.	Reactor Cooling Water Valve	Sticking
15.	Condenser Cooling Water Valve	Sticking
16.	Unknown	N/A
17.	Unknown	N/A
18.	Unknown	N/A
19.	Unknown	N/A
20.	Unknown	N/A
21.	The Valve for Stream 4	Constant Position

3.2 Application of the s-FDD Framework

In this study, we are building SVM binary classifiers for 21 (20 for *Rieth et. al*) different faults (fault-specific classifier) introduced in the process data. Thus, for each of the model building phase, we combine data from normal and relevant faulty operation. The s-FDD framework consists of two

phases: (i) Offline phase includes the formulation of the fault-specific models for fault detection and diagnosis via signal process data where the optimization-backed feature selection algorithm is used; (ii) Online phase monitors ongoing process in real-time by employing the fault-specific models, raises alarm when faults occur and reports diagnosis of the detected fault simultaneously.

Furthermore, as presented in Section 2.3.2 of Chapter 2, we integrate hierarchical clustering analysis and the s-FDD framework with multi-class classification for fault identification. Fault identification is significant in the case of multiple fault occurrences in a process, where the number of the occurring fault or fault types is unknown. In order to tackle this challenge, we build a cascaded, decision tree based classification scheme via s-FDD framework. In the offline phase, we build a two-class *C*-SVM model for fault detection, calculate the mean process trajectories from the *Chiang et. al* process data and cluster them via hierarchical clustering, and finally train two-class *C*-SVM models for fault type groups based on the extracted hierarchy from the clustering dendrogram. Whereas in the online phase, we use the built classifiers in an order guided by the hierarchical clustering dendrogram. Note that, here, the pairwise fault type groups are obtained by cutting the tree into two iteratively. Common to each phases of both analysis, data needs to be re-organized and/or processed *a priori* as described below.

3.2.1 Data Pre-processing

The common first step in data-driven modeling is the assessment of data quality. This is achieved via several different data pre-processing techniques such as (i) data cleaning that involves identification and removing outliers, smoothening the noisy data, and imputation of any missing values, and (ii) data transformation that includes scaling and normalization of the data to give all features equal weight, thus avoid bias during model development. In this study, we are using simulation based data set which is free of outliers or missing values, and solely involves Gaussian white noise (20). Therefore, only normalization is performed on process data by calculating their corresponding z-scores, by subtracting the mean of relevant measurements and then dividing into the standard deviation of them, prior to the offline phase. In the online phase, where actual process data is monitored in real time, both data cleaning and transformation steps are performed prior to the use of the developed models.

3.2.2 Offline Phase: Model Building for Fault Detection and Diagnosis

In this phase, we build fault-specific two-class *C*-SVM models by using simulation based process signal data. Here, the initial step is to collect relevant faulty operation process data and process data under normal operation. Next, we normalize the process data as described in Section 3.2.1 and construct balanced training and validation sets to be used in model building. Use of imbalanced data sets may cause insufficient learning of one class than the other, thus may lead to inaccurate models. Therefore, we initially create balanced training and validation sets via 100 runs of 5-fold cross validation where each fold includes 480 (960) normal and 480 (960) faulty samples for *Chiang et. al* data set (*Rieth et. al*). Next, we build binary fault-specific *C*-SVM classifier models for each of the 21 (20) faults separately. Specifically, since we have 52 process variables in Tennessee Eastman process, we build 52 *C*-SVM classifiers for each 21 (20) faults (one per each feature subset). Finally, we select the end-model for each fault (fault-specific end-model) which has the optimal feature subset yielding best model performance, for online implementation. The performance metrics utilized throughout model building phase are provided in the Appendix A. The iterative model building procedure, which consists of 3 main steps, is described below and illustrated in Figure 3.2.

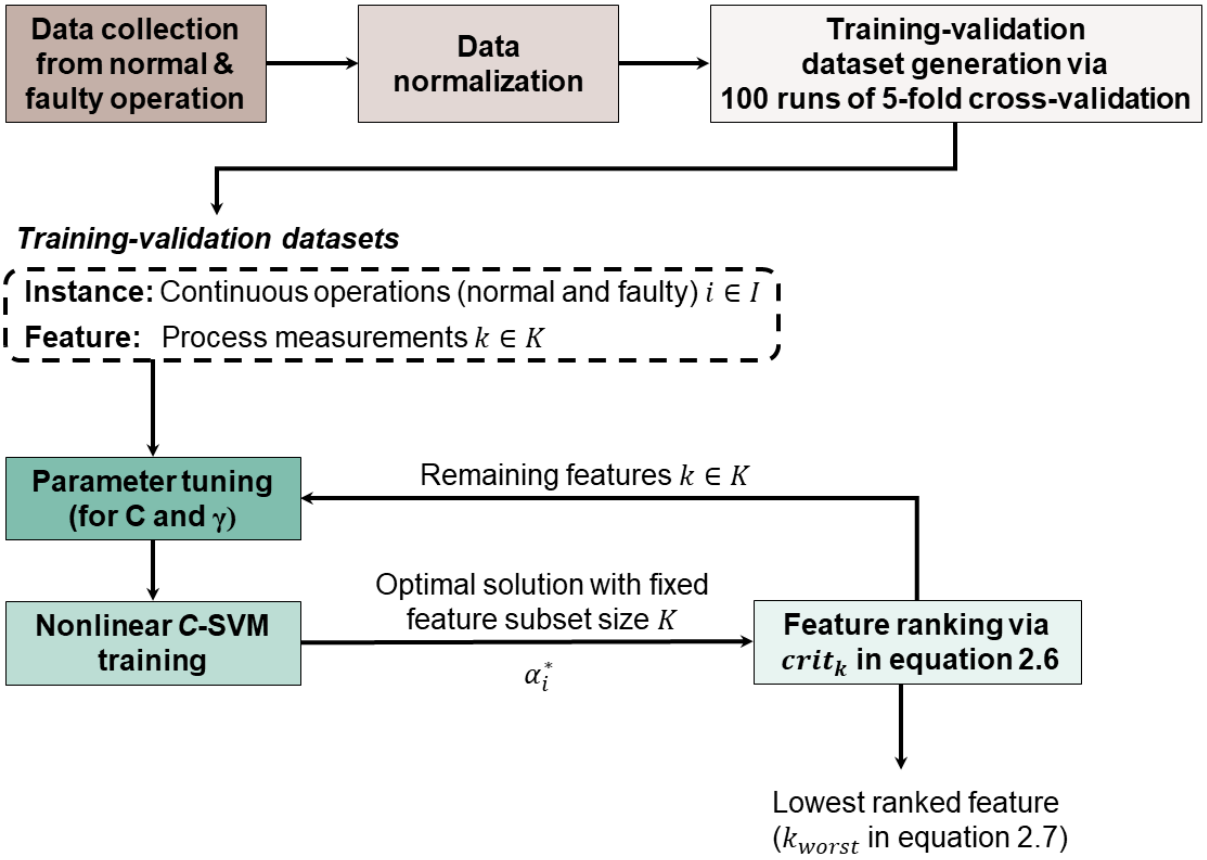


Figure 3.2: Schematic representation of the offline phase - model building section. Gaussian Radial Basis kernel is used for nonlinear *C*-SVM training. Iterative procedure is performed for each fault separately.

3.2.2.1 Tuning C-SVM Hyperparameters with the Active Set of Features

Parameter tuning is essential and required for developing generalizable models that will be implemented as a decision tool in online phase. In this work, we build *C*-SVM classification models by adopting one of the widely used nonlinear Kernel function, Gaussian radial basis function (RBF) (Equation (4.1)).

$$K(x_i, x_j) = \exp\left(-\gamma \left\| x_i - x_j \right\|^2\right), \tag{3.1}$$

Hence, we have hyperparameters C and γ that are tuned by using training and validation data

sets with the active set of features (whole feature set in the first iteration). During the selection of hyperparameter γ, data density plays a critical role to prevent overfitting in the obtained decision function. Thus, we tune parameter $\hat{\gamma}$ where

$$\gamma = \frac{2^{\hat{\gamma}}}{n}. \tag{3.2}$$

In each iteration of model building, where features are eliminated in a greedy reductive manner (Section 2.2), $\hat{\gamma}$ is updated with the available set of features as follows:

$$\gamma = \frac{2^{\hat{\gamma}}}{z^{\mathsf{T}}\mathbf{1}} \tag{3.3}$$

Additionally, we tune parameter $\hat{C}$ where $C = 2^{\hat{C}}$.

We perform a grid search to tune parameters $\hat{C}$, and $\hat{\gamma}$ for all value combinations between -10 : 10. In each iteration, we train and validate two-class C-SVM models using 500 training-validation data set pairs that include the corresponding active set of features. Then, the hyperparameter combination yielding the highest average testing AUC, accuracy, recall along with minimum false alarm rate across the 500 data set pairs for each feature subset are chosen for further modeling steps.

3.2.2.2 Training Fault-specific C-SVM Classifiers for Each Feature Subset

We adopt the selected hyperparameters from the previous step and build C-SVM classifiers with Gaussian Radial Basis Function, where the class probabilities are smoothened by using median of probabilities with a window size of 3.

3.2.2.3 Feature Rank Criteria Calculation and Elimination of the Least Informative Feature

We obtain feature ranks by using Equation (2.6). Then, according to the criteria formulated as in Equation (2.7), we eliminate the "worst", which is most redundant or least informative, feature from the data set.

This iterative framework involving greedy reductive feature elimination leads us to have fault-specific C-SVM models for each feature subset. Thus, we attain one C-SVM model per feature subset per fault. This leads to 52 model generation for each fault classification, which renders 1092 (1040 for Rieth et.al data set) fault-specific classifiers. The final step of the offline phase is the

selection of the fault-specific end-models. These are the fault-specific models that yield highest model performance with the optimal feature subset. Specifically, these models produce highest AUC along with minimum number of features, false alarm rate, false negative rate and latency (fault detection time). In this work, we have picked 21 (20 for Rieth et. al data set) fault-specific end-models among 1092 (1040) developed models.

3.2.3 Offline Phase: Model Building for Fault Identification

For fault identification, we first need to understand the characteristics of fault types. Here, we assume that we do not have the prior information on the faults. Therefore, we gather all faulty continuous operations under one major faulty process data set. The goal is to generate classifiers that can be used in online process monitoring to identify the fault type and further diagnose it. Below, we summarize the main steps for the calculation of the mean process trajectories from the process data, hierarchical clustering analysis to obtain the pairwise fault type groups to separate and the order of separation from the extracted hierarchy from the clustering dendrogram.

3.2.3.1 Building a Fault Detection Model for Normal Versus Faulty Process Identification

Here, the goal is to train a unique classifier for identifying the normal operation data from the rest of the faulty data. Therefore, we merge entire faulty operation data into a single faulty data set, then train a two-class *C*-SVM model that can accurately differentiate the normal operating condition (NOC) process data from any type of faulty data. The procedure of this classifier training is identical to the training of each two-class fault specific *C*-SVM model training as described in Section 4.2.2. Therefore we omit the details of this step for the sake of brevity.

3.2.3.2 Hierarchical Clustering and Two-class C-SVM Model Development for Fault Identification

We calculate the average of each faulty continuous process operations provided in the *Chiang et. al* data set to extract the characteristics of each fault type. This has produced a 52-dimensional vector per each faulty operation. Next, we calculate the pairwise Euclidean distance among 21 faulty operation and generate the input square distance matrix for the hierarchical clustering. We have utilized the Ward's method for the linkage technique within the hierarchical clustering. The analysis is performed by using the "hclust" function of the "stats" package of R statistical software.

The output of this analysis yields a clustering dendrogram which demonstrates the hierarchy of the clusters. Next, we extract the pairwise fault groups (clusters) by cutting the clustering dendrogram into two starting from the top. We cut the tree iteratively until we separate each fault from each other. The order of the separated fault clusters determines the order of the two-class *C*-SVM classifiers' use during the online phase.

Once the hierarchy of fault separation is determined, we build two-class *C*-SVM models for each pair-wise fault clusters. We follow the general steps of the s-FDD framework, which is described in detail for developing the fault-specific end-models in Section 4.2.2. Here, instead of separating individual faulty operations from the normal operation data, we collect process data from each faulty operation clusters. By dividing the dendrogram into two groups iteratively, we obtain 20 different levels of fault groupings. Therefore, we train 20 two-class *C*-SVM classifier that separates one faulty operation from another. With this iterative tree-based classification scheme, we are able to identify the ongoing faults in the process while reporting the optimal process variables to diagnose the specific fault or fault groups. The 20 end-classifiers obtained at the end of this stage are then implemented for the online phase.

3.2.4 Online Phase: Fault Detection, Identification and Diagnosis in Real Time

In this phase, 21 fault-specific end models that are chosen at the end of the offline phase are implemented to monitor the online process data for fault detection and diagnosis. Here, we are using the test data sets obtained from the simulation of Tennessee Eastman process, which includes 160 normal and 800 faulty samples, to assess the performance of the selected models. The test data sets are normalized with the mean and standard deviation obtained from the training sets (to attain z-scores) before being sent into the end-models. In an industrial setting, the real time process data needs to be pre-processed with cleaning and transformation (i.e. normalization) steps as described in Section 3.2.1 before use of the implemented *C*-SVM models for fault detection and diagnosis. Fault-specific end-models generate binary answer for detection of each fault independently. Here we adopt an alarm policy to identify the fault occurrence. In compliance with the studies of Mahadevan and Shah (3), and Russell et. al (69), fault occurrence is reported after we observe 6 consecutive positive alarms within the system. Once a fault is detected from any of the end-models, the developed framework simultaneously produces the root-cause analysis by solely

checking the corresponding optimal set of features (process variables).

Additionally, we implement 21 classifiers for fault identification in online monitoring. These are 21 hierarchy-level specific models that include 1 classifier for general fault detection, which separates normal operation data from the rest of the 21 faulty operation data, and 20 fault or fault group-specific classifiers for fault identification. Initial step is to use the fault detection classifier to understand whether a fault is occurring in the process or not. If a fault is detected, then we start testing the process data by using the 20 C-SVM decision functions in the order of the clustering tree levels. We adopt the similar alarm policy, where we require 6 consecutive positive answer for the identification of the fault or fault group. This cascaded procedure is similar to a branch-and-bound technique, where arms of the tree levels can be fathomed if 6 consecutive alarms are raised for the other arm of the tree. Therefore, it saves computational time and facilitates the interpretation of the fault characteristics significantly. Furthermore, the selected optimal feature subset of the classifier raising the alarm provides the diagnosis of the process abnormality. This is essential in order to take further rapid actions to recover the process back to the nominal condition.

3.3 Results

Fault-specific C-SVM binary classifier models have been built by using the training data sets created from the two different simulation data sets of different size. The adopted simulation data sets, namely *Chiang et. al* and *Rieth et. al* data sets, include all of the 52 process variables but differ in terms of the number of simulated continuous operations where the latter is the twice size of the former one. The evaluation of the proposed framework for continuous processes is examined via fault detection performance in Section 3.3.1, where increasing number of instances have been observed to increase the model accuracy for the detection of distinct faults. Section 3.3.2 compares the fault detection latency of the reported end-models with the ones in the literature. Furthermore, diagnosis of the successfully detected faults is provided according to the most accurate C-SVM model in Section 3.3.3. Finally, we adopt the *Chiang et. al* data set to demonstrate how to use the s-FDD framework for fault identification in Section 3.3.4.

3.3.1 Fault Detection

In this section, we evaluate the performance of the chosen fault-specific end-models for fault detection. The 21 (20 for Rieth et. al data set) end-models yield highest AUC along with minimum

number of features, false alarm rate, false negative rate and latency (fault detection time). The results are tabulated in Tables 3.2 and 3.4, respectively for Chiang et. al and Rieth et. al data sets. As mentioned earlier, in compliance with the studies of Mahadevan and Shah (3), and Russell et. al (69), we report fault occurrence at the end of 6 consecutive positive alarms within the system. This policy is widely adopted in industry to minimize the number of false alarm rates, thus disruption of the operator. Furthermore, in accordance with the same studies, the fault detection latency is reported as the first time when the initial alarm is raised.

The end-models reported in Table 3.2 are obtained with the *Chiang et. al* data set, which is smaller compared to Rieth et. al data. In particular, fault-specific models are trained with 480 normal and 480 faulty samples, and tested on 48 hour simulation data where each fault is introduced at the end of 8^{th} hour, which corresponds to having 160 normal and 800 faulty samples sequentially.

An ideal model would give 100% AUC, accuracy, detection rate along with 0% false alarm and negative rates. Among the introduced performance metrics (Appendix A), accuracy and fault detection rate (recall) are the two common ones used for model performance evaluation in the literature (45; 3). Yet, evaluation based on only these two metrics would be insufficient for a thorough analysis. In this study, we inspect the model performance via collective evaluation of AUC, fault detection rate, accuracy as well as false alarm rate and false negative rate. We believe collective judgment of AUC, fault detection rate, accuracy along with false alarm rate and false negative rate is essential and required since a model may simultaneously yield high accuracy, and fault detection rate, but also high false alarm rates which would lead to misleading conclusions. Such models would be very sensitive and frequently raise fault alarm, consequently making them unreliable.

Table 3.2: Performance results of the selected models developed with *Chiang et. al* data set. Asterisks imply that the end-model is selected for online implementation.

Fault	Optimal Feature Subset Size	AUC	Accuracy	Detection Rate	False Negative Rate	False Alarm Rate	Latency (min)
1*	2	100.0	99.9	99.9	0.1	0.0	6
2*	5	99.5	98.1	97.8	2.3	0.0	57
3	13	62.0	67.8	72.1	27.9	53.8	3
4	1	100.0	100.0	100.0	0.0	0.0	3
5	4	100.0	99.9	99.9	0.1	0.0	6
6	2	100.0	100.0	100.0	0.0	0.0	3
7*	3	100.0	100.0	100.0	0.0	0.0	3
8	7	99.4	96.5	95.8	4.3	0.0	60
9	17	66.3	65.2	69.0	31.0	53.8	3
10*	14	90.2	83.3	85.8	14.3	28.8	12
11	29	84.9	86.8	96.6	3.4	62.5	3
12	9	99.9	95.7	100.0	0.0	25.6	3
13*	7	98.2	93.2	91.9	8.1	0.0	153
14	3	100.0	100.0	100.0	0.0	0.0	3
15	27	75.6	67.2	65.5	34.5	24.4	3
16	5	86.9	87.7	96.9	3.1	58.1	3
17	32	99.5	94.1	92.9	7.1	0.0	72
18	34	94.4	91.7	90.0	10.0	0.0	231
19	2	29.8	75.4	88.5	11.5	90.0	3
20	14	94.5	86.8	85.0	15.0	4.4	45
21*	1	99.7	100.0	100.0	0.0	0.6	3

As can be seen from the Table 3.2, fault-specific models perform well excluding Fault 3, 9, 11, 15, 16, and 19. Specifically, faults 3, 9 and 15 are the ones that could not be detected with the available algorithms in the literature. This is due to the absence of observable change in the process

variable behavior (mean and standard deviation) between their corresponding faulty and normal operation (69). In other words, these faults could not be detected with the set of provided process variables, however models can always be improved by considering additional process variable information. For instance, for fault 15, which is sticking condenser cooling water valve, additional process variables such as position of the valve and/or condenser pressure would have been extremely helpful for the detection. Then, by using the presented simultaneous modeling and feature selection algorithm (Section 2.2), we can obtain the optimal feature subset that would produce the most accurate model to detect these faults. Particularly for faults 3, 9, and 15, although high fault detection rate and accuracy have been obtained in particular models built for the detection, high false alarm rates have also been recorded, whereas corresponding AUC metric has fluctuated around 42.8%-64.51%, 42.51%-66.29%, and 42.08%-75.71%, respectively. Of note, 50% of AUC indicates random assignment of the class label. Specifically, highest AUC received for fault 3, 64.51%, is recorded for the fault-specific model developed with 35 optimal features yielding 82.19% accuracy, 98.62% fault detection rate but with 100% false alarm rate. This means that the model is very sensitive and raises alarm for fault detection frequently regardless of the operation characteristics, which turns the model into an inaccurate tool. Similarly, for fault 9, highest AUC yielding model is obtained with 17 features which produce 65.21% accuracy, 69.00% fault detection rate together with 53.75% false alarm rate; whereas for fault 15, highest AUC This shows that AUC metric becomes significantly informative in data-driven model selection, especially when the testing data set is unbalanced with the number of samples from two different classes. Moreover, since unbalanced data set would be very common in continuous operation online data, we offer use of AUC metric in the future fault detection studies, where problem is formulated as classification problem, to attain a more complete picture of the results.

Moreover, Table 3.2 reveals that our proposed data-driven algorithm has achieved to detect fault 21, regarding a fixed Stream 4 valve at the steady state position, successfully with an AUC of 99.7%, accuracy and detection rate of 100% with 0% false negative rate and 0.6% false alarm rate. Here, we would like to highlight that this is one of the most challenging faults of the Tennessee Eastman process simulation, where, to our knowledge, highest fault detection rate recorded in the literature is 59.4% along with 26.1% false alarm rate via one-class Support Vector Machine algorithm (4). Again, to have a detailed analysis of this model, we strongly suggest to evaluate AUC

and false negative rate metrics as well. Here, we show that with the advances in C-SVM formulation for feature selection, we achieve to detect fault 21 with high AUC, accuracy, detection rate and minimum false negative and alarm rates by considering solely one feature, which is a manipulated process variable - total feed flow rate of Stream 4. This also demonstrates the requirement for feature elimination during model development. Consideration of any further process variable in addition to 45^{th} process variable, has deteriorated the model performance significantly. In other words, other process variables become redundant to detect this fault.

The selection of end-models is a multi-objective task. The fault-specific end-models are the ones producing the highest AUC with minimum number of features, false alarm and negative rates, and latency (Table 3.2 and 3.4). Here, we also report alternative models demonstrating similar performance to end-models but with lower number of features (process variables) (Table 3.3 and 3.5).

On the other hand, the end-models trained via *Rieth et. al* data set perform well for all faults excluding only Fault 3, 9, 15. This shows that models can be improved with addition of more simulation data. As the *Big Data* era has started playing significant role in industrial decision making, today large amount of process data collection has been extremely facilitated. Therefore, accessibility to further process data is assumed not to be an issue. As for the opposite scenario, where historical process data is not available or not adequate, one can simulate more process data with the dynamic model of a process to improve model performance with the proposed framework. Here, we see that using larger data has improved fault detection model performances for faults 11, 16, 19, and 20. This is the result of the fact that the models have learned much better by being trained with increased number of scenarios (i.e. simulations) for both normal and faulty operation. Furthermore, we would like to highlight that the addition of new and more training data (scenarios) also affects the learning pattern of the models and due to the nonlinear dynamics of the process, this may lead to the selection of different feature sets with two different data sets used in this study. Yet the key goal in data-driven modeling is to obtain generalizable models, and in this study we ensure this by using 100 runs of 5-fold cross validation technique during model development.

Table 3.3: Performance results of the selected alternative models developed with *Chiang et. al* data set. Asterisks imply that the end-model is selected for online implementation.

Fault	**Optimal Feature Subset Size**	**AUC**	**Accuracy**	**Detection Rate**	**False Negative Rate**	**False Alarm Rate**	**Latency (min)**
8*	4	99.2	95.9	95.8	4.2	3.1	63
17*	27	99.1	93.4	92.1	7.9	0.0	75

Table 3.4: Performance results of the selected models developed with *Rieth et. al* data set. Asterisks imply that the end-model is selected for online implementation.

Fault	Optimal Feature Subset Size	AUC	Accuracy	Detection Rate	False Negative Rate	False Alarm Rate	Latency (min)
1	18	100.0	99.8	99.8	0.3	0.0	15
2	10	99.7	99.2	99.1	0.9	0.0	24
3	10	50.2	17.0	0.4	99.6	0.0	2442
4*	1	100.0	100.0	100.0	0.0	0.0	3
5	14	100.0	100.0	100.0	0.0	0.0	3
6*	2	100.0	100.0	100.0	0.0	0.0	3
7	4	100.0	100.0	100.0	0.0	0.0	3
8	12	99.0	94.5	93.4	6.6	0.0	54
9	2	54.3	55.1	55.9	44.1	49.4	15
10	15	87.9	80.1	77.6	22.4	7.5	72
11*	2	99.8	95.9	95.1	4.9	0.0	18
12*	5	99.9	99.4	99.3	0.7	0.0	15
13	8	95.5	87.3	84.8	15.2	0.0	90
14*	2	100.0	100.0	100.0	0.0	0.0	3
15	16	56.0	17.6	1.1	98.9	0.0	1635
16*	2	96.7	88.3	87.4	12.6	7.2	9
17	28	96.9	92.9	91.5	8.5	0.0	210
18	5	98.6	96.1	95.3	4.7	0.0	75
19	10	100.0	99.6	99.6	0.4	0.0	18
20	14	97.1	92.7	91.2	8.8	0.0	99

Next, we compare the obtained end-models with *Chiang et. al* and *Rieth et. al* data sets provided in Tables 3.2 and 3.4 as well as the alternative models given in Tables 3.3 and 3.5. We select the simplest end-models for the online decision making. According to the scientific interpretation of the Occam's razor philosophy, if one has two competing theories that would yield the same

predictions, the simpler one is the better (178). By following this principle, we select the fault-specific end-models (for faults 1-20) with lower number of features if they demonstrate similar performance between the two data sets. The selected "simple" end-models are marked with asterisks. This also facilitates relevant sample data collection, and analysis due to decreased number of sample collection and analysis.

Table 3.5: Performance results of the selected alternative models developed with *Rieth et. al* data set. Asterisks imply that the end-model is selected for online implementation.

Fault	Optimal Feature Subset Size	AUC	Accuracy	Detection Rate	False Negative Rate	False Alarm Rate	Latency (min)
5*	3	100.0	99.9	99.9	0.1	0.0	6
18*	2	98.2	95.3	94.3	5.7	0.0	105
19*	3	99.7	99.2	99.0	1.0	0.0	9
20*	13	97.2	92.1	90.5	9.5	0.0	102

Finally, we pick our fault-specific models yielding highest fault detection rate among the developed 1092 and 1040 fault-specific models from two simulation data sets, and compare our results with the available data-driven methods performed on the Tennessee Eastman process data (3; 4; 5). These methodologies are based on well-known and widely used algorithms, where in some of them only normal operating data is used (3; 4) and in the others normal and faulty operation data is utilized simultaneously (5). Table 3.6 shows that our proposed framework produces better results than the other methods, however we need to highlight that the models producing highest detection rate do not necessarily produce the most reliable models for all faults. As mentioned earlier, a model can produce high fault detection rate but also high false alarm rate. In fact, this is the case for faults 8, 10, 11, 13, 16, 17, 19, and 20 reported in Table 3.6. For these faults, although we achieve high fault detection rates as reported in Table 3.6, we observe false alarm rate of 68.1%, 91.2%, 100.0%, 83.8%, 81.9%, 100.0%, and 97.5%, respectively. Therefore, we strongly suggest to evaluate all metrics described in Appendix A in order to make a fair comparison between models. Specifically,

the end-models reported in this study are selected based on AUC, which considers fault detection rate and false alarm rate, false negative rate and latency (fault detection time).

Table 3.6: Comparison of fault detection rate between the fault-specific models producing highest fault detection rate of this study and the models reported in the literature (3; 4; 5). Best results of *Xiao et. al* is adopted.

Ref.	**Mahadevan & Shah, 2009**					**Xiao et. al, 2016**	**Yin et. al, 2014**	**This Study**
Fault	**PCA-T^2**	**PCA-Q**	**DPCA-T^2**	**DPCA-Q**	**1-class SVM**	**1-class SVM**	**2-class SVM**	**2-class SVM**
1	99.2	99.8	99.4	99.5	99.8	99.5	99.5	**99.9**
2	98.0	98.6	98.1	98.5	98.6	98.3	98.1	**99.1**
4	4.4	96.2	6.1	100.0	99.6	47.4	99.9	**100.0**
5	22.5	25.4	24.2	25.2	**100.0**	45.2	90.8	**100.0**
6	98.9	**100.0**	98.7	**100.0**	**100.0**	99.2	60.1	**100.0**
7	91.5	**100.0**	84.1	**100.0**	**100.0**	70.1	98.9	**100.0**
8	96.6	97.6	97.2	97.5	97.9	97.4	96.0	**100.0**
10	33.4	34.1	42.0	33.5	87.6	68.0	81.0	**99.4**
11	20.6	64.4	19.9	80.7	69.8	65.8	80.2	**100.0**
12	97.1	97.5	99.0	97.6	99.9	98.8	97.8	**100.0**
13	94.0	95.5	95.1	95.1	95.5	95.0	92.5	**100.0**
14	84.2	**100.0**	93.9	**100.0**	**100.0**	93.9	91.0	**100.0**
16	16.6	24.5	21.7	29.2	89.8	73.1	89.4	**100.0**
17	74.1	89.2	76.0	94.7	95.3	75.2	81.6	**98.2**
18	88.7	89.9	88.9	90.0	90.0	89.3	89.5	**95.3**
19	0.4	12.7	0.7	24.7	83.9	43.6	85.9	**100.0**
20	29.9	45.0	35.6	51.0	90.0	69.0	80.5	**100.0**
21	26.4	43.0	35.6	44.2	52.8	59.4	-	**100.0**

3.3.2 Fault Detection Latency

The average latency among the reported faults, all faults excluding Faults 3,9, and 15, has been stated as 306.19, 145.58, 263.12, 151.00, and 98.50 min for PCA-T^2, PCA-Q, DPCA-T^2, DPCA-Q, and 1-class SVM (3), respectively whereas the latency information is not provided in Yin et. al (45). In this work, we report significantly lower fault detection latency along with higher detection accuracy as reported in Tables 3.2 and 3.4. When we exclude Faults 3, 9, and 15, faults that cannot be detectable from the available process variable data accurately, the average fault detection latency among the remaining 18 (17) faults of Chiang et. al (Rieth et. al) data set is 37.50 (42.00) min. Moreover, the average fault detection latency for the selected "simple" end-models for online implementation that are marked with asterisks is 35.83 min. This reveals the power of the proposed framework for rapid and precise fault detection and diagnosis.

3.3.3 Fault Diagnosis

Here, we present the root cause diagnosis of the detected faults by using the chosen end-models for online implementation (the models with asterisks). Below, we discuss the obtained diagnosis results for the selected faults 3.7. The diagnosis with the end-models reported in Tables 3.2 and 3.4 is provided in the Appendix A. Of note, we may observe distinct feature sets for different types of faults but related with the same unit in the process (e.g. Faults 5 and 12 in Table A7 in Appendix A). This is mainly because of the fact that varying fault types realize themselves in distinct ways due to the nonlinear dynamics of the process, which leads to the selection of different process variables as key ones for fault diagnosis.

3.3.3.1 Fault 1 - Sudden Decrease in A/C Feed Ratio (Stream 4)

Fault 1 occurs due to a step change in the A/C feed ratio which also changes B composition constant at Stream 4. The chosen end-model (Table 3.2) is able to detect this fault by monitoring process variables 16, and 44, that are stripper pressure on Stream 5 and A feed flow rate, respectively. Here, sudden increase in C flow rate causes an increase in the stripper pressure, which is a measured process variable. To compensate this sudden effect and maintain the B composition constant on Stream 4, the flow controller increases the feed flow rate of A. This, in turn, reverses the stripper pressure to the original operating range, however, the raise in the A feed flow rate carries the operation to a new steady-state which can be clearly observed in Figure 3.3.

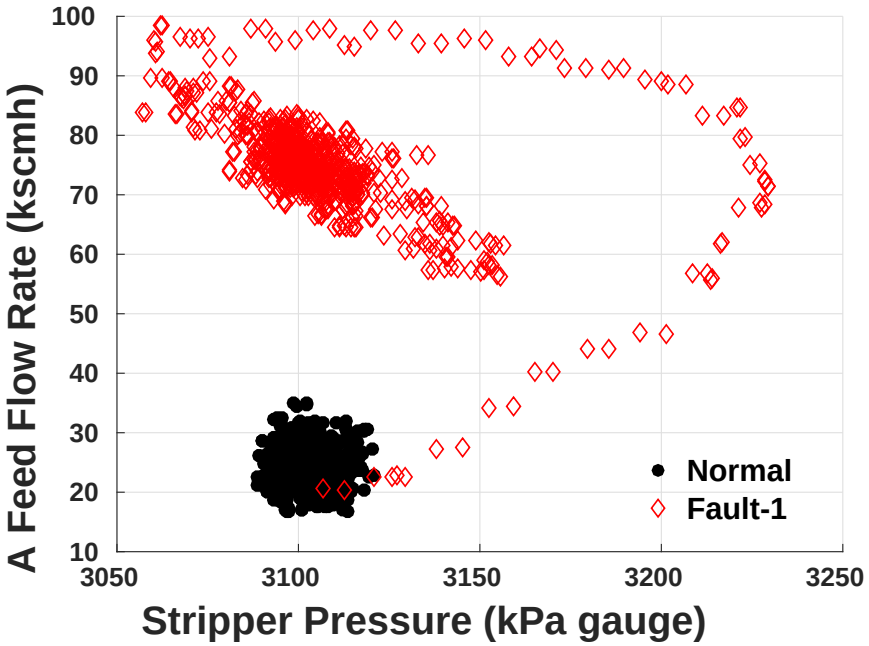


Figure 3.3: Fault 1 diagnosis - plot of the root cause variables.

3.3.3.2 Fault 4 - Step Change in Reactor Cooling Water Inlet Temperature

The end-model selected for detection of this fault is given in Table 3.4. By monitoring the manipulated process variable 51, we are able to detect this fault with 100.0% accuracy along with 0% false negative and alarm rates in 3 min. To decrease the elevated reactor cooling water inlet temperature, the controller increases the condenser cooling water flow rate (Figure 3.4). Therefore monitoring of this process variable provides valuable insights for the identification of this fault.

Table 3.7: Diagnosis from the selected end-models (marked with asterisks in Tables 3.2, 3.3, 3.4, and 3.5) via Occam's Razor principle. Faults 3, 9, and 15 are excluded.

Fault	Optimal Feature Subset Size	Selected Process Variables
1	2	16, 44
2	5	7, 16, 10, 47, 13
4	1	51
5	3	52, 11, 17
6	2	44, 1
7	3	45, 7, 13
8	4	39, 44, 16, 20
10	14	41, 39, 38, 37, 40, 50, 19, 18, 20, 7, 13, 16, 31, 29
11	2	9, 51
12	5	16, 38, 35, 25, 11
13	7	39, 40, 18, 7, 38, 23, 3
14	2	51, 9
16	2	19, 50
17	27	38, 39, 40, 41, 21, 37, 19, 20, 33, 27, 34, 30, 1, 11, 25, 28, 24, 23, 35, 36, 26, 10, 3, 2, 22, 14, 48
18	2	22, 8
19	3	13, 16, 46
20	13	38, 39, 41, 16, 52, 17, 18, 30, 35, 29, 40, 13, 7
21	1	45

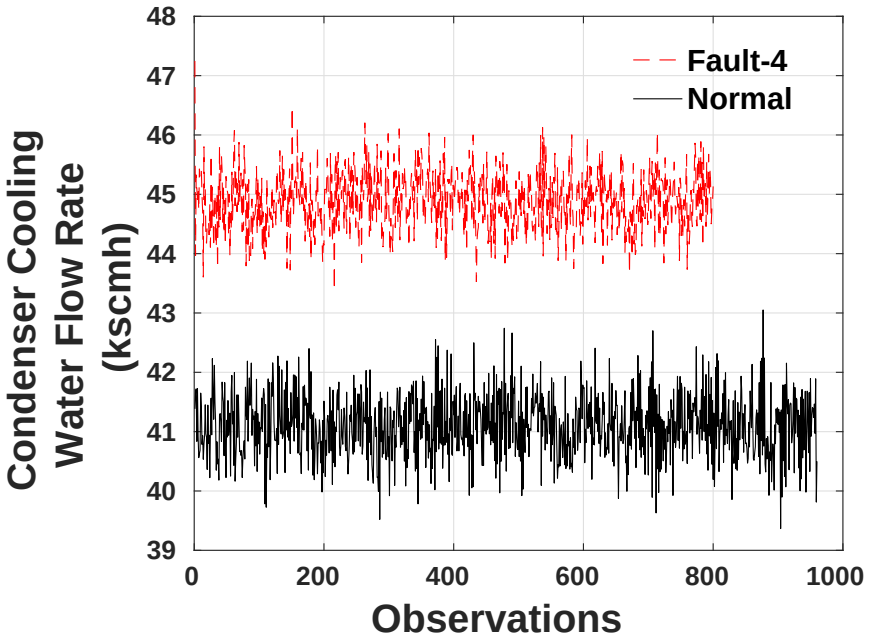


Figure 3.4: Fault 4 diagnosis - plot of the root cause variables.

3.3.3.3 Fault 5 - Step Change in Condenser Cooling Water Inlet Temperature

Fault 5 is generated with a step change in the condenser cooling water inlet temperature in the simulations. We are able diagnose this fault by monitoring process variables 52, 11, and 17, which are agitator speed (manipulated variable), product separation temperature (measured variable), and stripper underflow (measured variable), respectively. Because of the temperature increase in the cooling water, the cooling performance of the condenser decreases. In order to compensate this adverse effect, the flow controller increases the flow rate of the condenser cooling water by increasing the agitator speed (process variable 52). The step change in cooling water temperature also affects the product separation temperature, and accordingly stripper flow rate. We have plotted the selected process variables, that provide the most informative set of samples to detect this fault, and clearly seen the distinction between normal and faulty operation (Figure 3.5). By monitoring these 3 process variables, the fault-specific model reported for fault 5 in Table 3.5 is able to detect the fault with 99.9% accuracy with 0.0% false alarm rate and 0.1% false negative rate in 6 min.

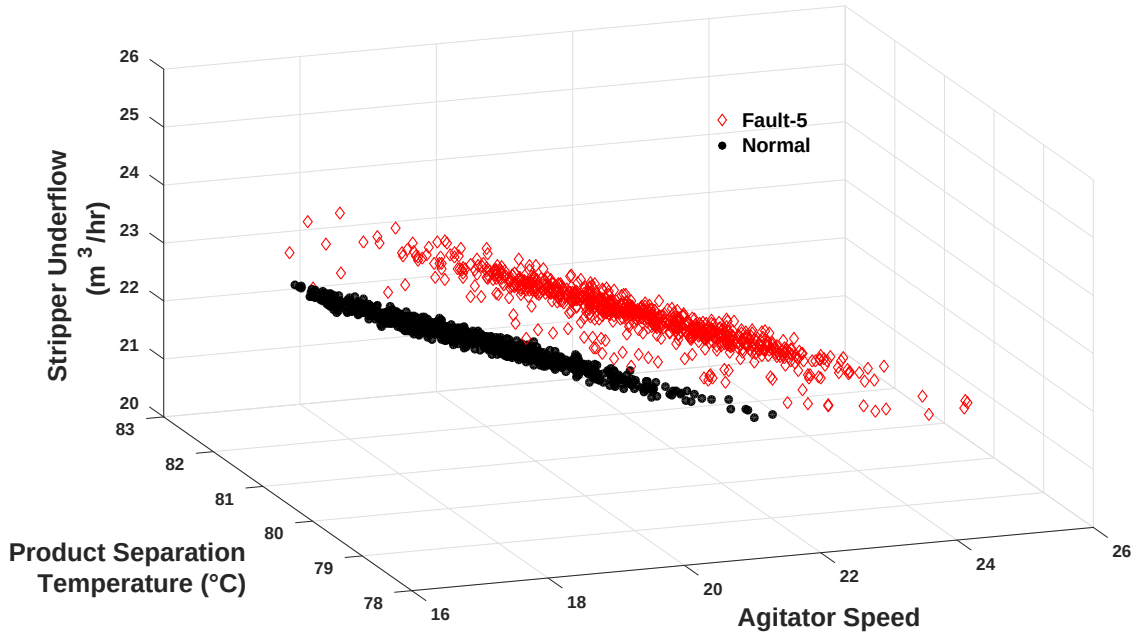


Figure 3.5: Fault 5 diagnosis - plot of the root cause variables.

3.3.3.4 Fault 7 - C Header Pressure Loss

The selected end-model, given in Table 3.2, detects this fault with 100.0% accuracy with 0% false alarm and false negative rates in 3 min. The diagnosis of this fault is obtained from process variables 45, 7, and 13, which are manipulated process variable - total feed flow rate on Stream 4, measured process variables reactor pressure and product separation pressure, respectively. Here, to compensate the decreased C header pressure, total feed flow rate is increased via the adjustment of the flow valve on Stream 4, which in turn affects the reactor pressure and product separation pressure within the process. Therefore monitoring these three key process variables has enabled accurate detection of the fault 7.

3.3.3.5 Fault 14 - Sticking Reactor Cooling Water Valve

We detect this fault by using the end-model provided in Table 3.4, where we achieve 100.0% accuracy with 0% false alarm and false negative rates in 3 min. The analysis reveals that process variables 51 (manipulated), and 9 (measured) are the two key process variables to identify this fault. Here, if the reactor cooling water temperature is elevated, thus there occurs a lost cooling

effect on the reactor, we observe a direct temperature increase in the reactor (measured process variable 9). The controller then tries to decrease the elevated reactor temperature by increasing the condenser cooling water flow rate (manipulated process variable 51). On the other hand, if the reactor cooling water temperature decreases due to the sticking valve, creating increased cooling effect on the reactor, we notice a decline in the reactor temperature, where the controller would decrease the condenser cooling water flow rate for balance (Figure 3.6). Moreover, we are able to achieve same model performance by observing an additional process variables along with process variables 51, and 9, whereas the model with less number of process variables was favored due to the simplicity. The 3^{rd} ranked key process variable is the measured process variable 21, reactor cooling water outlet temperature, which is directly affected with the sticking reactor cooling water valve. When we consider this 3^{rd} ranked key process variable, and visualize the sampling data, the distinction between normal and faulty operation becomes more evident (Figure 3.7).

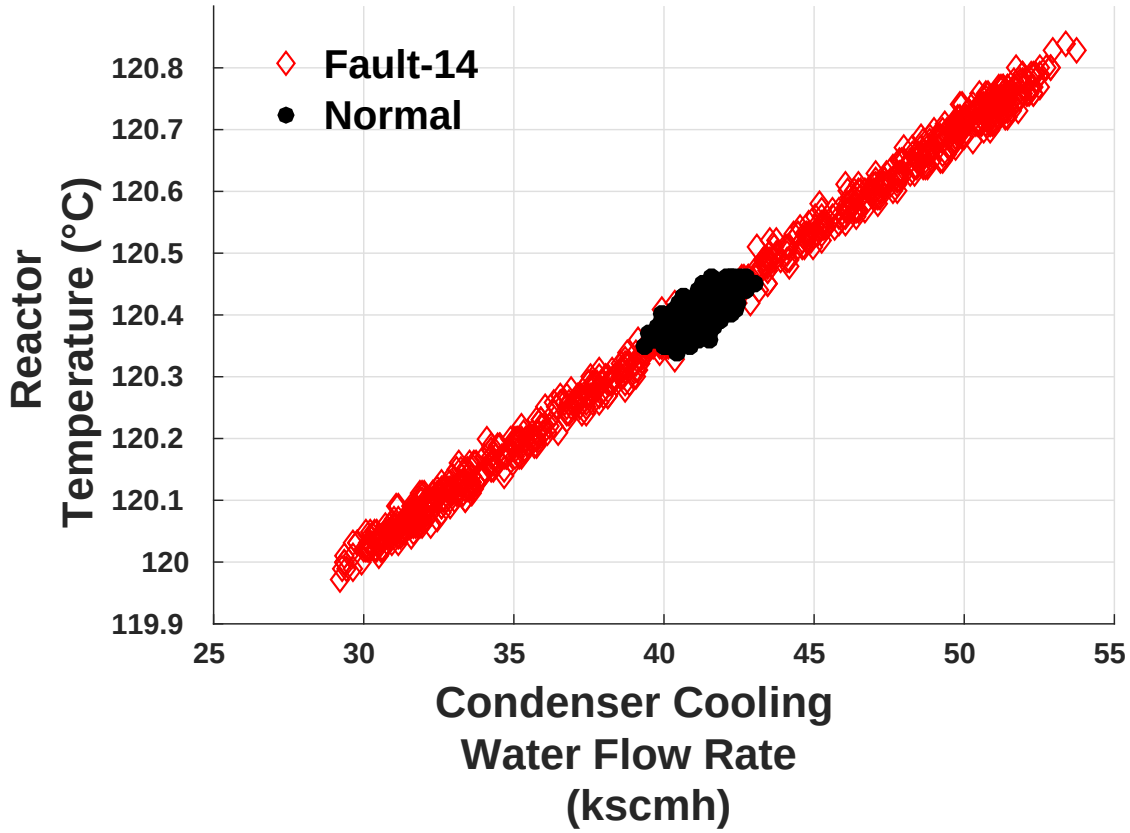


Figure 3.6: Fault 14 diagnosis - plot of the root cause variables.

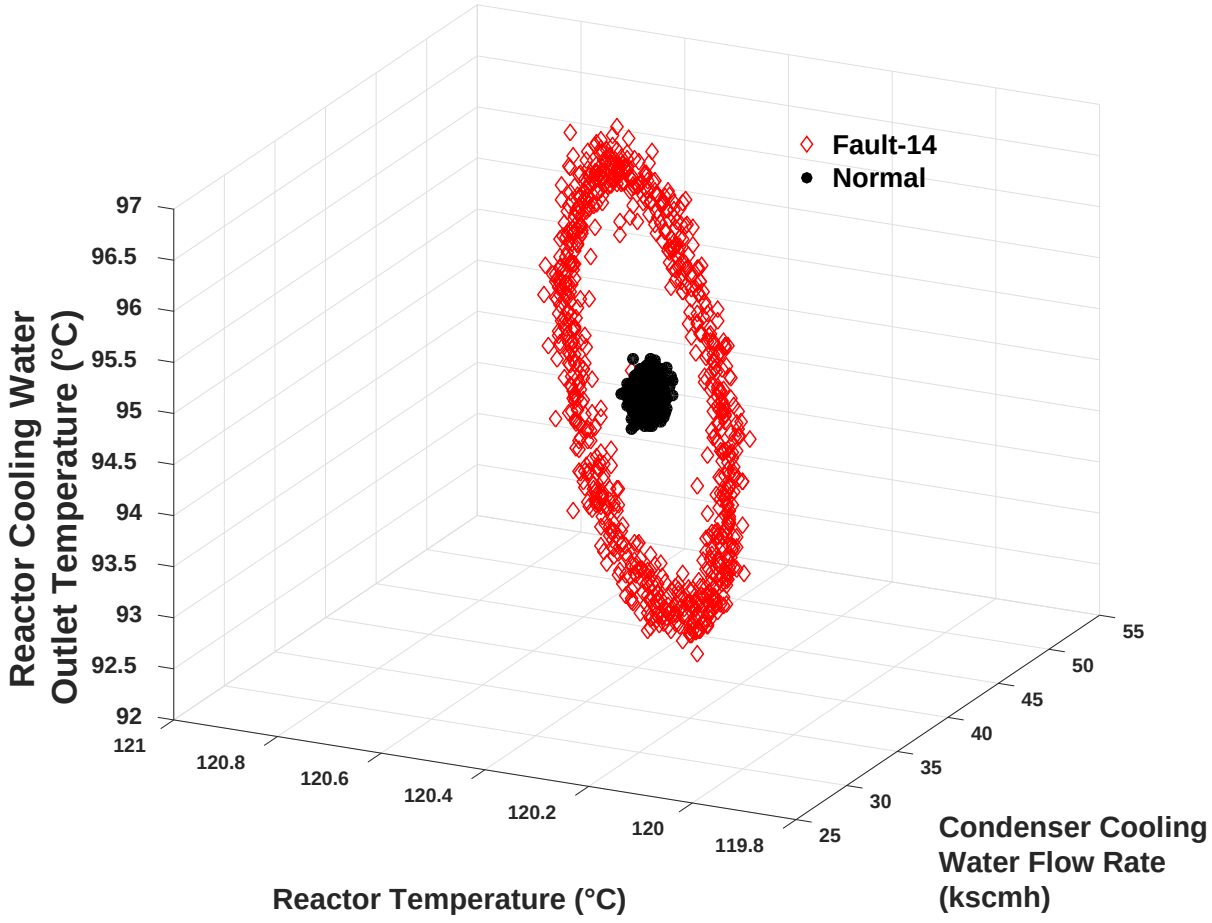


Figure 3.7: Fault 14 diagnosis with top 3 key process variables.

3.3.3.6 Fault 19 - Unknown

The cause of this fault is not provided in Downs and Vogel (176) However, we observe the clear distinction between normal and faulty operation with the optimal set of diagnosed process variables (Figure 3.8). These are measured process variables 13, and 16, as well as a manipulated process variable 46, which are product separation pressure, stripper pressure, and compressor recycle valve, respectively.

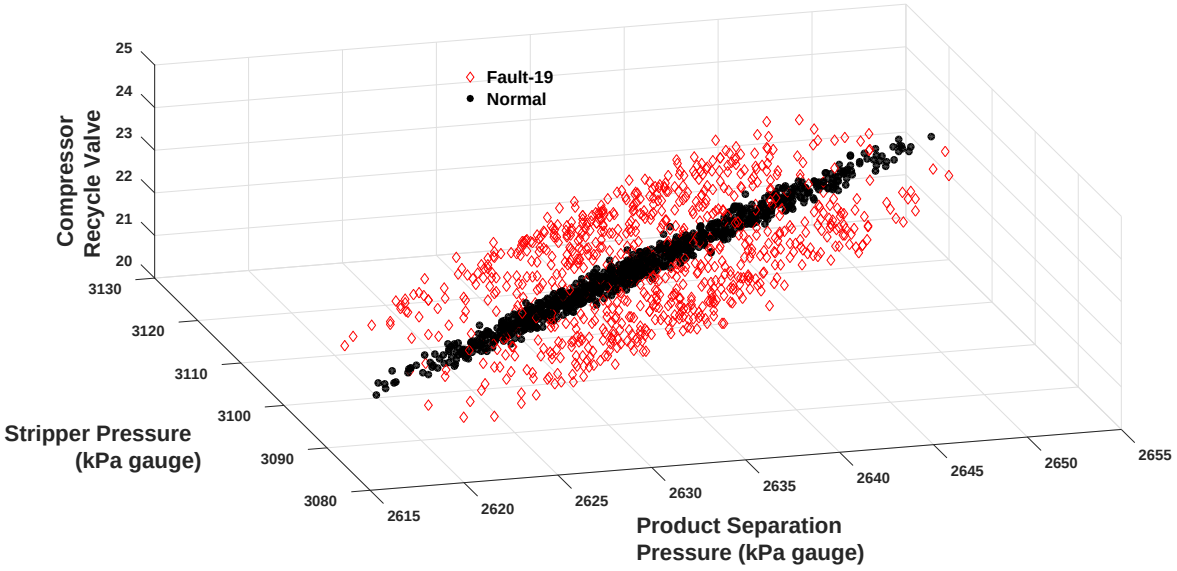


Figure 3.8: Fault 19 diagnosis - plot of the root cause variables.

3.3.4 Fault Identification

By benefiting from hierarchical clustering analysis and the s-FDD framework, we build level-specific classifiers which are used as the decision functions during th online operation. The first level classifier is used for fault detection. Therefore the pairwise classes to separate are "normal operating condition (NOC)" and "faulty". Then, we use the rest of the classifiers to search for the fault occurring in the system. This search is achieved by building a decision-tree type of classification scheme. Particularly, when we detect a fault, and use the second classifier (C-SVM end-model for the level 2), the answer we get directs us to the specific part of the hierarchical tree for further separation. Therefore, we do not need to move further in the arm, thus can be fathomed.

When we consider all faults during the clustering analysis, we obtain a 21-level hierarchy where we identify normal and 21 distinct faults. The hierarchy of separation is extracted from the clustering dendrogram (Figure 3.9). The fault classes separated at each level are provided in Table 3.8 along with the corresponding accuracy of the 21 end-models. Here, we observe that we achieve high fault detection rate and accuracy at each level of the separation. Note that, although the detection rate is high in the first level for identifying the fault occurrence, the accuracy is accuracy is slightly lower. This ise due to the inclusion of process data from faults 3,9, and 15

which are defined as challenging faults to detect, wherein observable change in the process variable behavior is lacking. Yet, this can be further improved with the addition of additional process variable information or extracting informational process descriptors. In order to avoid the adverse impact of these three specific faults, we perform an additional analysis. Here, we exclude faults 3,9, and 15 during the hierarchical analysis and build *C*-SVM models by using the s-FDD framework for the remaining faults. The accuracy of these models are tabulated in Table 3.9, where we also provide the classes of separation at each level.

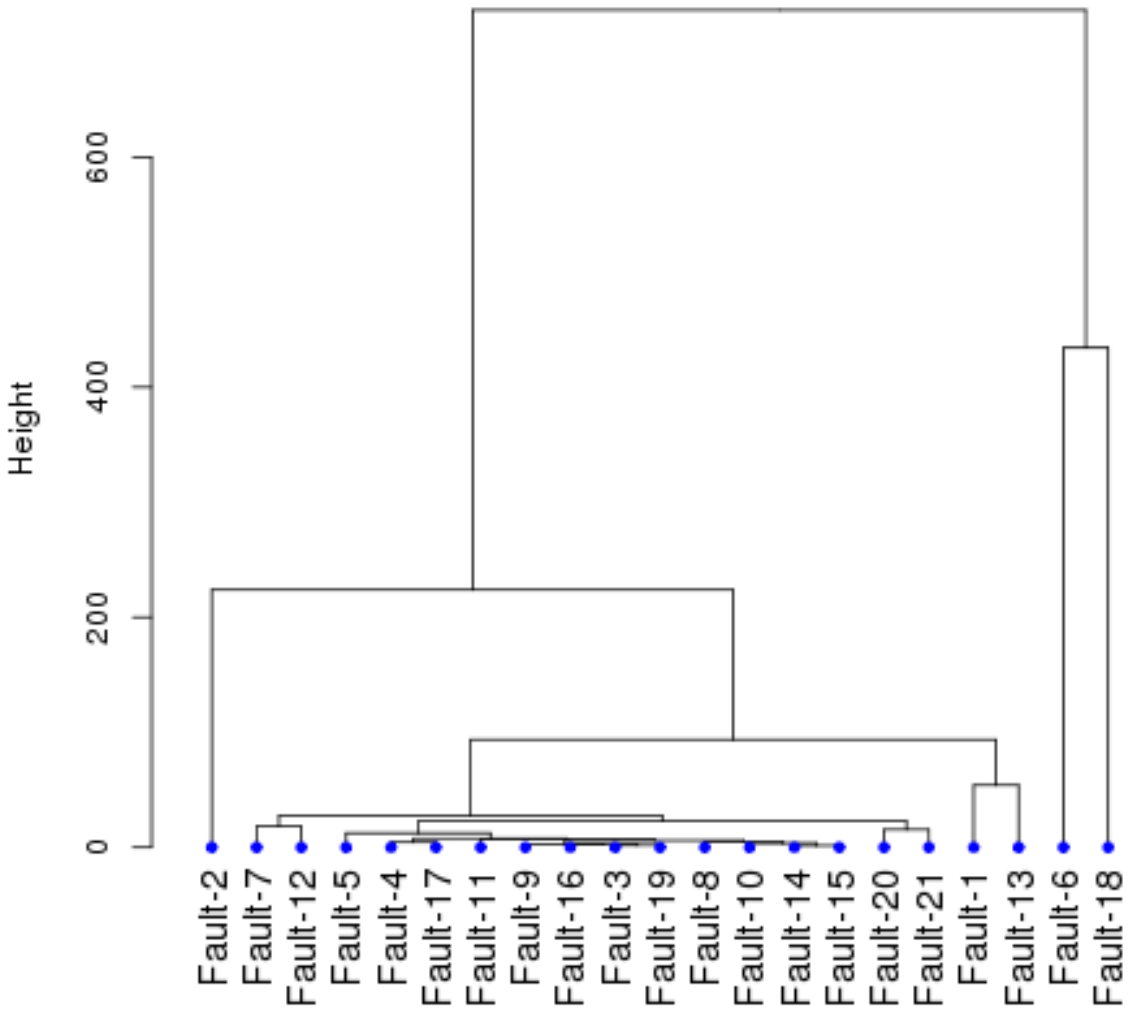


Figure 3.9: Hierarchical clustering dendrogram of faults.

Table 3.8: Performance results of the end-models for fault identification developed with *Chiang et. al* data set. (F: Fault).

Level	Optimal Feature Subset Size	Detection Rate	Accuracy	Class 1	Class 2
1	50	92.00	79.07	Normal	Faulty
2	47	92.13	95.76	F6, F18	Rest
3	2	100.00	100.00	F6	F18
4	20	99.65	99.53	F2	Rest
5	30	98.64	98.88	F1, F13	Rest
6	13	99.96	99.98	F1	F13
7	27	98.14	98.68	F7, F12	Rest
8	52	83.91	82.48	F20, F21	Rest
9	2	100.00	100.00	F7	F12
10	12	98.14	98.89	F20	F21
11	30	99.80	99.90	F5	Rest
12	18	94.64	94.55	F4, F17	Rest
13	14	94.78	95.06	F11	Rest
14	9	97.72	86.99	F8, F10, F14, F15	F3, F9, F16, F19
15	20	97.00	98.49	F8	F10, F14, F15
16	2	100.00	100.00	F17	F4
17	4	99.72	73.04	F14, F15	F10
18	10	98.07	94.01	F9	F3, F16, F19
19	10	96.51	97.23	F16	F3, F19
20	18	94.28	96.36	F19	F3
21	5	100.00	99.66	F15	F14

Table 3.9: Performance results of the end-models for fault identification developed with *Chiang et. al* data set - excluding F3, F9, and F15 (F:Fault).

Level	Optimal Feature Subset Size	Detection Rate	Accuracy	Class 1	Class 2
1	23	98.14	86.22	Normal	Faulty
2	44	92.22	95.70	F6, F18	Rest
3	2	100.00	100.00	F6	F18
4	21	99.54	99.45	F2	Rest
5	26	98.36	98.81	F1, F13	Rest
6	13	99.98	99.99	F1	F13
7	32	98.20	98.68	F7, F12	Rest
8	14	94.86	90.88	F20, F21	Rest
9	2	100.00	100.00	F7	F12
10	12	98.12	98.85	F20	F21
11	52	99.81	99.90	F5	Rest
12	6	86.38	86.40	F11	Rest
13	8	95.15	95.03	F17	Rest
14	10	95.76	97.21	F8, F10	F4, F14, F16, F19
15	14	97.65	98.70	F8	F10
16	4	99.93	99.96	F4	F14, F16, F19
17	13	99.34	99.67	F14	F16, F19
18	12	99.12	98.29	F16	F19

The exclusion of the three challenging faults has improved the fault detection rate and accuracy in the first level of the separation hierarchy. Specifically, the fault detection rate is increased from %92.00 to %98.14, where the accuracy is increased from %79.07 to %86.22. This is significant in order to accurately detect the fault in the monitored system and initiate the fault identification process. The major advantage of this cascaded classification analysis is that its computational efficiency, such that when a fault is identified, the search can be terminated. This enables rapid

fault identification. Furthermore, the optimal feature subset of the final classifier reveals the fault diagnosis simultaneously. In other words, once the fault is identified, operators do not need to further perform analysis to diagnose the fault. Once the fault is identified, we provide its diagnosis instantaneously. A representative diagnosis is shown for the level-3 separation, where we identify between fault 6 and 18 (Figure 3.10).

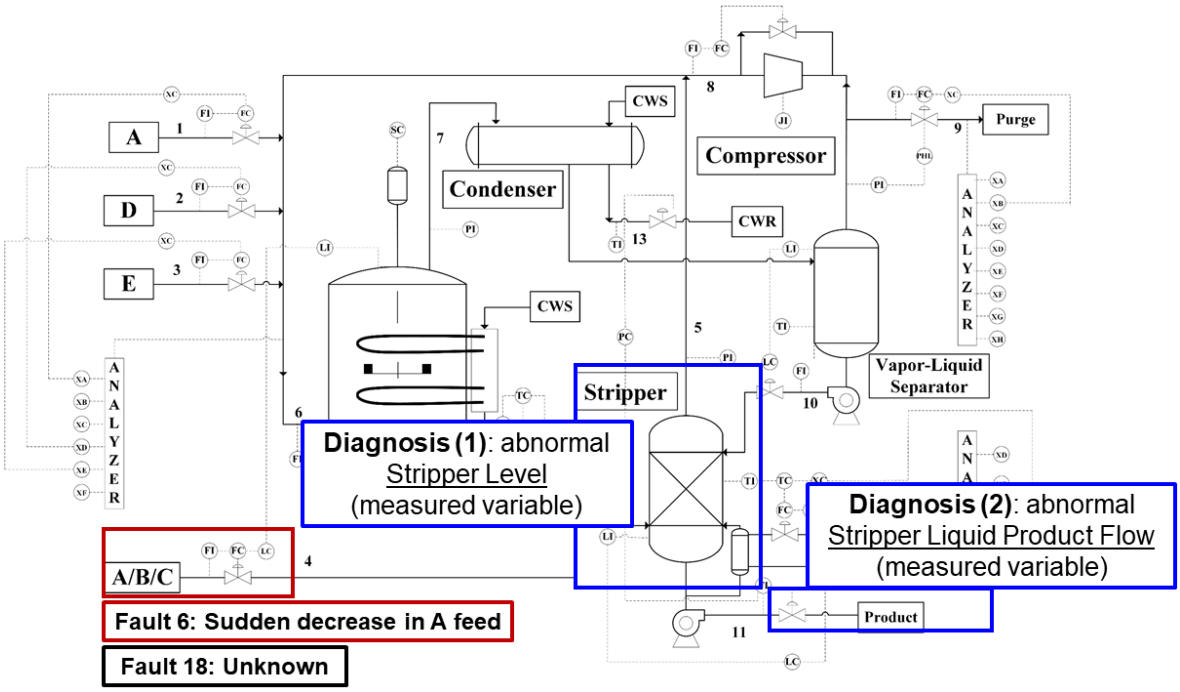


Figure 3.10: Diagnosis from level-3: Separating Fault 6 and 18.

3.4 Conclusions

Dimensionality reduction is a key task in most data-driven applications, in areas such as multi-scale systems engineering, where vast amounts of data must be reduced to an essential subset that is used to provide actionable insights. Process monitoring, specifically fault detection and diagnosis, is one of the major fields in process systems engineering that benefits the advances in data-driven modeling and dimensionality reduction techniques with the increased availability of process data. In this work, we present theoretical advances in the feature selection algorithm based on nonlinear Support Vector Machines, describe a data-driven framework for fault detection and diagnosis

in continuous processes, and finally apply it to the Tennessee Eastman benchmark process. The presented feature selection algorithm is based on nonlinear Kernel-dependent SVM feature rank criteria, which is derived from the sensitivity analysis of the dual *C*-SVM objective function. This enables simultaneous modeling and feature elimination which paves the way for simultaneous fault detection and diagnosis, where feature ranking guides fault diagnosis. Thus, once the implemented fault detection models detect a fault within the process, they are able to instantly report the diagnosed process variables. Moreover, by adopting feature selection techniques which list the most informative features in the original space rather than feature extraction (i.e. PCA, PLS) where features become linear combination of the original features in a transformed space, loss of information is highly minimized and interpretation of the results become more convenient.

In this work, we have demonstrated the application of the s-FDD framework for (i) fault detection and diagnosis, and (ii) fault identification and diagnosis, respectively. For fault detection, we have developed 1092 and 1040 fault-specific *C*-SVM binary classifier models for 52 feature subsets of the 21 and 20 faults simulated in *Chiang et. al* and *Rieth et. al* data sets, respectively. The fault-specific end models that yield highest Area Under the ROC Curve (AUC) along with minimum number of features, false alarm rate, false negative rate and latency (fault detection time) are selected for online implementation. For the cases where we have observed similar model performances for the detection of a fault, we have followed the Occam's razor principle and selected the one that provides diagnosis with minimum process variables, for simplicity. The achieved results with the presented framework are highly promising. Specifically, excluding the faults 3, 9, and 15, the models that we report in this study outperforms the available ones in the literature not only in terms of detection accuracy but also detection latency. The detection latency is attained as low as 35.83 min for the 18 faults (excluding fault 3,9, and 15) analyzed with our framework. Of note, faults 3, 9, and 15 are the ones that are not accurately detected neither in this work nor previous studies. This is due to the fact that available process variable set does not provide a distinct observable change between normal and corresponding faulty operation. However, we also note that consideration of further process variables for certain faults, specifically faults 3, 9, and 15, can highly improve the model performances. Furthermore, for distinct set of faults (i.e. faults 11, 16, 19, and 20), we have benefited from the larger simulation data set (*Rieth et. al*) where added samples have contributed the learning of the developed models.

On the other hand for fault identification, we have developed two-class *C*-SVM classifiers by following a similar approach, yet this time classifiers are utilized for a cascaded search of fault identification. We have calculated the average process trajectories of each operation data and used them in hierarchical clustering analysis to attain the hierarchy of fault class separations. This yields 21 level-specific model development, where the initial model is for fault detection with the detection rate of %92.00 and accuracy of %79.07. The fault detection model performance is significantly improved when we exclude the faults 3,9 and 15 (i.e. %98.14 fault detection rate with %86.22 accuracy). This study demonstrates an application of the s-FDD framework for fault identification. Lastly, we highlight the importance of utilizing additional evaluation metrics (i.e. Area Under the ROC Curve (AUC), accuracy, false alarm rate, and false negative rate) for detailed model performance assessment for all analysis. Commonly used metric in the previous studies is fault detection rate. However, one can end up with a model producing high fault detection rates along with high false alarm and false negative rates, meaning that the model would become very sensitive, thus unreliable. Therefore, collective interpretation of these metrics is critical to avoid such unstable models that would obfuscate the decision-making process.

4. S-FDD FRAMEWORK FOR BATCH PROCESS MONITORING*

Although batch reactor processes are prevalent in chemicals, food, and pharmaceutical industry, most of the data-driven fault detection and diagnosis frameworks presented in the literature focus on continuous processes. Data-driven batch process monitoring is a challenging task due to the challenging characteristics of batch process data citeVanImpe2015,Wang2017 such as (i) involvement of a considerable number of interconnected variables, (ii) inherent non-stationarity, (iii) finite duration, (iv) nonlinear response, and (v) batch-to-batch variability. Moreover, as the process structure becomes more complicated with the changes/enhancements in process technologies, process data becomes even higher in dimension with the addition of further process variables, which further obstructs and complicates the monitoring. Thus, understanding the nonlinear relationship among process variables is becoming more difficult and the need for novel data-driven fault detection and diagnosis frameworks that can meet this demand is growing significantly. In this chapter, application of s-FDD framework is presented for batch process monitoring, where semi-batch penicillin production process is utilized (179) as a benchmark.

4.1 Batch Process Benchmark Model and Data Set

The batch process data is adopted from an extensive simulation dataset(180) based on penicillin production, PenSim benchmark model(179), where the model is expanded with sensor noise (Table 4.1). The process operates in two modes. It starts in the batch mode with high substrate (glucose) concentrations that stimulates biomass growth. Then it switches to fed-batch mode with the depletion of glucose where penicillin production is triggered by biomass due to low glucose content in the bioreactor (179). pH and temperature of the process is monitored via closed-loop PID controllers, where aeration rate, agitator power, feed rate, feed temperature, hot and cold water temperatures are in open loop. Schematic diagram of the fed-batch penicillin production is given in Figure 4.1.

*Reprinted with permission from "Big data approach to batch process monitoring: Simultaneous fault detection and diagnosis using nonlinear support vector machine-based feature selection" by Onel, M. and Kieslich, C.A. and Guzman, Y.A. and Floudas, C.A. and Pistikopoulos, E.N., 2018, *Computers & Chemical Engineering*, Vol. 116, pp 503-520, Elsevier [2018] by Elsevier and Copyright Clearance Center

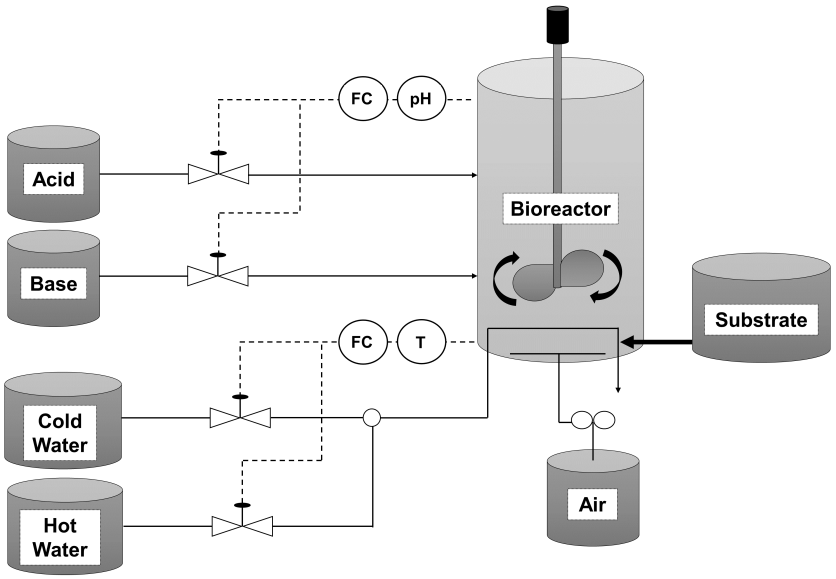


Figure 4.1: Fed-batch penicillin production flow diagram.

Table 4.1: List of online measurements

Online Measured Variables		Measurement Noise (σ)
1.	Fermentation volume [m^3]	0.002
2.	Dissolved O_2 concentration [mg/L]	0.004
3.	Dissolved CO_2 concentration [mg/L]	0.12
4.	Reactor temperature [K]	0.1
5.	pH [-]	0.02
6.	Feed rate [L/h]	1%
7.	Feed temperature [K]	0.1
8.	Agitator power [W]	1%
9.	Cooling/heating medium flow rate [L/h]	1%
10.	Heating medium temperature [K]	0.1
11.	Hot/cold switch [–]	–
12.	Base flow rate [mL/h]	1%
13.	Acid flow rate [mL/h]	1%

In this work, we use the aligned, base case simulation data introduced in *Van Impe et. al* where the initial fermenter volume, biomass concentration, and substrate concentration are independently sampled from normal distributions providing batch-to-batch variability(180). The alignment is done by bringing all simulated batches to equal length via indicator variables as described in *Birol et. al* (179). The data set includes 400 normal and between 1000-2000 batches per 15 different simulated process faults yielding 22,200 faulty batches in total. Table 4.2 tabulates 15 different fault types and corresponding number of batches simulated for each of them.

Table 4.2: Overview of faults and corresponding number of batches in the data set

Fault		No of Batches
1.	Sudden change in feed substrate concentration	2000
2.	Change in coolant temperature	2000
3.	Agitator power drop	1600
4.	Aearation rate drop	1600
5.	Gradual change of feed rate	1200
6.	Gradual dissolved oxygen sensor drift	2000
7.	Feed temperature change	1600
8.	pH sensor drift	1600
9.	Non-functional pH control	1200
10.	Reduced pH control	1200
11.	Reactor temperature sensor bias	1200
12.	Reactor temperature sensor drift	1600
13.	Reduced temperature control	1200
14.	Reduced temperature control - maximal flow not impacted	1200
15.	Contamination	1000

We have considered 13 process variables (5 state and 8 manipulated) that can be measured online (Table 4.1). In this work, we use *cumulative* acid/base flows [mL] instead of instantaneous flow rates as they are suggested to be relatively more informative (180). Each batch is completed in about 460 h where the sensors are sampled in every 0.02 h. When aligned, this corresponds to a total batch length of 1201 samples which results in 3-dimensional (3D) batch process dataset of size 1400-2400 batch x 13 variable x 1201 sample.

4.2 Application of the s-FDD Framework

The proposed framework consists of two phases: (i) Offline phase includes the formulation of the fault and time-specific models for fault detection and diagnosis via historical signal process data where the novel optimization-backed feature selection algorithm is used; (ii) Online phase

monitors ongoing batches in real-time by using the fault and time-specific models. Prior to both phases, data needs to be re-organized and/or processed.

4.2.1 Data Pre-processing

Here, we (i) divide the time horizon into intervals of fixed size which we refer as sample bins, (ii) unfold 3-dimensional batch process data into 2-dimension via batch-wise unfolding (162), (iii) incorporate additional informative features (feature extraction) from sample bins, (iv) normalize the obtained 2-dimensional data, and (v) eliminate the features having less than 10^{-8} standard deviation, respectively.

4.2.1.1 Creating Sample Bins

Batch process data is 3 dimensional. For each batch at each specific time point, it includes a process variable measurement. This initial step of data pre-processing slightly differs for offline and online phase.

In the offline phase, the motivation is to produce an online fault detection and diagnosis decision support tool for end-users, therefore we need to develop time-specific models for each fault. To do this, first, we partition the time horizon into intervals of fixed size and group the process variable measurements of batches at particular time periods. We refer these time periods as sample bins. Next, we train models for fault detection and diagnosis for each sample bin to obtain time-specific models. Here, the size of sample bins is a user-defined parameter. Selection of smaller bin size implicitly increases the number of checks on an ongoing batch, thus accelerates the detection of possible fault occurrences, yet increasing the number of models to be developed. In this study, we have selected the sample bin size as 10 which has resulted in total of 120 sample bins spanning the entire time horizon.

In the online phase, data is continuously collected. In order to analyze the incoming 3-dimensional data with the developed time-specific models, we need to gather the batch process measurements from the relevant time points and form the sample bins for further analysis.

4.2.1.2 Unfolding 3D Batch Process Data Into 2D

In order to train models with 3-dimensional batch process data, we need to unfold it into a 2-dimensional matrix via one of the three possible ways: (i) batches, (ii) process variable mea-

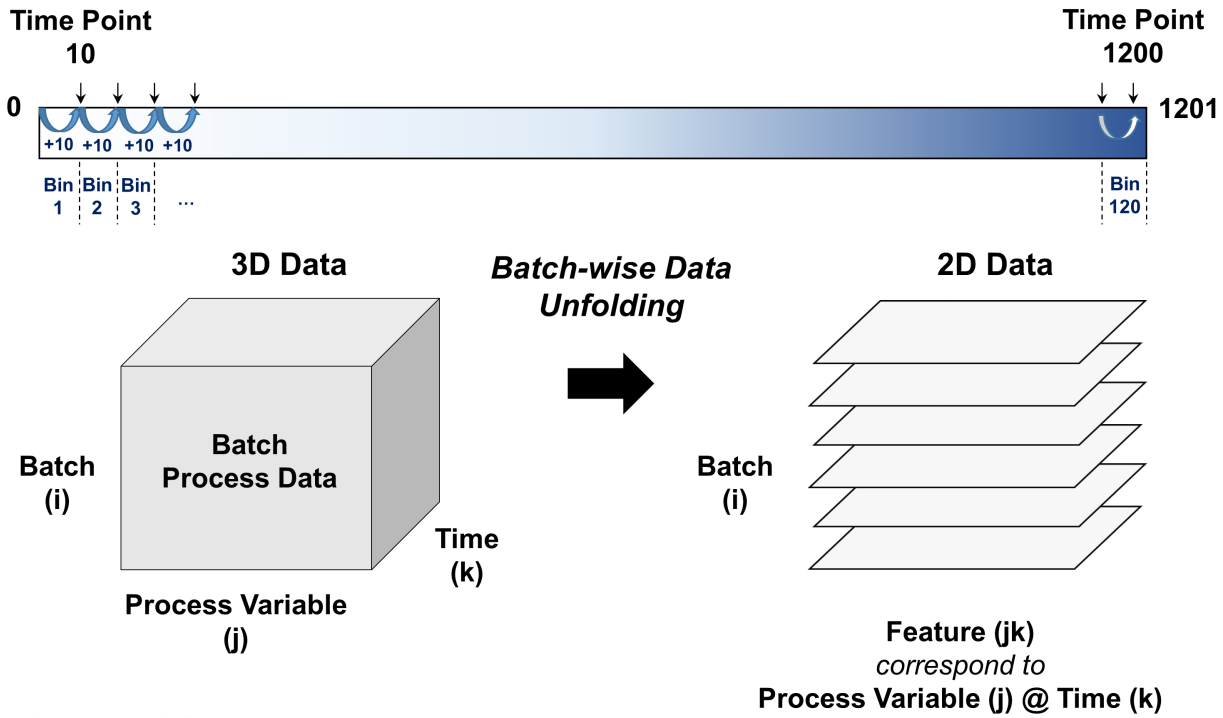
surements, or (iii) time points. In this work, we adopt batch-wise unfolding approach(162). The batch-wise data unfolding yields batches as the incidents (rows), and process variables at specific time points as the features (columns) of the 2-dimensional dataset. Of note, while we are unfolding the 3-dimensional data in each sample bin, we solely include the relevant (observed) process variable measurements. This data re-configuration enables inspection of batches in regular periods via time-specific models that use historical and most recent process signal data to determine whether the batch is performing well or not while the batch is still ongoing. The illustration of data structure pre-processing, that involves Step 1 and 2, is shown in Figure 4.2.

4.2.1.3 Extracting Additional Features

In order to improve the data-driven modeling accuracy, we extract and incorporate additional features that can characterize process behavior of each batch at each sample bin. In this study, in addition to the historical and most recent process variable measurements, we have utilized slope, standard deviation, and mean of 13 process variables (referred as process behavior features) within each sample bin. The case-specific addition of the features is given in Section 4.3. Here, we would like to highlight that the end-users of the presented framework will be allowed to include/exclude any type of process behavior features during the model development phase. In case of highly noisy plant data, one can always smoothen the data by using filters (i.e Kalman filter), and extract features from the filtered data to further use in the model building phase.

4.2.1.4 Data Normalization and Reduction

Finally, we normalize the re-configured and extended 2-dimensional matrix within each sample bin and perform *a priori* dimension reduction by eliminating the features (process variable measurements at specific time points and extracted features describing process behavior) with less than 10^{-8} standard deviation.



At each bin:

	Time Point k			Time Point k+1			...	Time Point k+9		
	Variable 1	...	Variable 13	Variable 1	...	Variable 13		Variable 1	...	Variable 13
Batch 1	<data>						...			
Batch 2										
.										
.										
.										
.										

Figure 4.2: Data pre-processing: Unfolding the batch process data and formation of sample bins.

4.2.2 Offline Phase: Model Building

In offline phase, we build fault and time-specific nonlinear Support Vector Machine classification models by using historical and/or simulation-based batch process signal data. Particularly, we train and test 15 separate C-SVM models per sample bin for detection and diagnosis of 15 distinct faults. To do this, we (i) select the pre-processed data of faulty and normal batches that have been observed within the selected time period (i.e. sample bin), (ii) create balanced training and test sets with 20 runs of 5-fold cross-validation, (iii) tune the hyperparameters C and γ of the (C-SVM) classification models, (iv) perform simultaneous feature selection and model building

via nonlinear Support Vector Machines, and (v) test model accuracies to determine the end-model to be implemented in the online phase, respectively.

4.2.2.1 Collecting Pre-processed Batch Process Data

In the first step of the offline model building, we retrieve the pre-processed 2-dimensional process data of faulty and normal batches that have been observed within the time period of the selected sample bin (Figure 4.3).

4.2.2.2 Generating Train and Test Data Sets

In order to have accurate and robust models, we need to balance the number of batches that belong to different class labels. In other words, we need to have equal amount of faulty and normal batches in training and test sets. Therefore, the next step in offline model building phase is to determine the number of faulty and normal batches within the observed sample bin period separately, include the limiting number of batches of one class label, and randomly select the same number of batches from the pool of batches with the other class label that have been observed within the specified time period. Then, we perform 5-fold cross-validation with 20 runs to create training-testing datasets in order to minimize generalization error and avoid over-fitting. This results in 100 training-testing dataset pair generation.

4.2.2.3 Tuning Hyperparameters for C-SVM Models

Next, we tune the hyperparameters C and γ of the C-SVM (two-class) classification models by using Gaussian radial basis function (RBF),

$$K(x_i, x_j) = \exp\left(-\gamma \left\| x_i - x_j \right\|^2\right), \tag{4.1}$$

as the nonlinear Kernel function. The appropriate selection of hyperparameters is necessary for the sake of generalization error minimization. The density of the data is a critical factor in the selection of hyperparameter γ to avoid over-fitting in the resulted separating decision function model **(cite)**. The default value for RBF kernel hyperparameter, γ, used by LIBSVM is $1/n$, where n is the number of features. Therefore, we tune parameter $\hat{\gamma}$ where

$$\gamma = \frac{2^{\hat{\gamma}}}{n}. \tag{4.2}$$

Similarly, we tune parameter $\hat{C}$, where the relation between $\hat{C}$ and C is:

$$C = 2^{\hat{C}}. \tag{4.3}$$

According to the described iterative feature selection algorithm in Section 2.2, $\hat{\gamma}$ can be updated in each iteration with the available set of features:

$$\gamma = \frac{2^{\hat{\gamma}}}{z^{\top}\mathbf{1}} \tag{4.4}$$

In this work, tuning is performed for parameters $\hat{C}$, and $\hat{\gamma}$ via a grid search for all value combinations between $-10:10$. We train and test models using all features of 100 training-testing dataset pairs with every combination of $\hat{C}$ and $\hat{\gamma}$ parameters. Here, instead of repeating grid search for hyperparameters tuning after each iteration of feature elimination, which would be ideal but inefficient, we perform tuning at first iteration where we include the whole set of features. Then, the hyperparameters with the highest average testing accuracy are chosen for the next steps.

4.2.2.4 Simultaneous Model-informed Feature Selection and Classification

The tuned hyperparameters are incorporated into simultaneous model-informed feature selection and classification algorithm via *C*-SVMs which is described in Section 2.2. Here, we iteratively build *C*-SVM binary classification models with Gaussian radial basis function (RBF) kernel starting from the complete set of features until we are left with the last feature in the data set. In each iteration, we eliminate features based on the Lagrangian sensitivity of the dual objective function of the built *C*-SVM model (objective function of model 2.5) with respect to the feature subset size. This iterative procedure is performed with each of the 100 training and testing data set pairs which creates 100 feature rankings for each fault class at each sample bin. Subsequently, one average feature ranking list is created for each fault and sample bin combination according to the statistical distribution of the feature ranks across 100 individual ranking lists.

4.2.2.5 Building C-SVM Models with Average Feature Rank Lists

At this stage, we train *C*-SVM classification models with Gaussian RBF kernel with the same 100 training and testing datasets by using the fault and time-specific average feature ranking lists. Starting from the entire feature set and eliminating one feature at a time according to the average

fault and time specific feature rank list, we train *C*-SVM classification model with each of the 100 test sets for each feature subsets. Thus, we obtain one fault and time-specific *C*-SVM model per each feature subset. The model performances are then evaluated via several metrics for each test set, and averaged for each feature subset. These metrics are fault detection rate, accuracy, area under the receiver operator curve (AUC), and false alarm rate.

4.2.2.6 Choosing the End-models for Online Phase

In the last stage of the offline phase, we determine the end-models. Among all developed fault and time-specific *C*-SVM models for each feature subset, the ones yielding the highest fault detection rate are selected. These are referred as the end-models. The end-models, that have the optimal feature subset for maximum fault detection rate, are aimed to be further implemented in the online phase.

The overall framework of offline phase is summarized in Figure 4.3.

4.2.3 Online Phase: Fault Detection and Diagnosis in Real Time

In this phase, we implement the fault and time specific end-models to create a decision support tool for online fault detection and diagnosis. These are selected binary classifiers, the optimal decision functions, which will evaluate the incoming pre-processed online batch process data and produce a binary answer for fault occurrence. The fault and time-specific classifier models inherently include the optimal set of features, which are the most informative process measurements/characteristics, to diagnose the detected fault at the time period of interest. Therefore, the end-models are able to provide instantaneous rank-ordered root-cause analysis when a fault occurs, which enables them to be employed for an online simultaneous fault detection and diagnosis tool. The end-user can further interpret and link the rank-ordered fault diagnosis to corrective actions to reverse the fault. The schematic representation of the implementation of the models for online phase is shown in Figure 4.4.

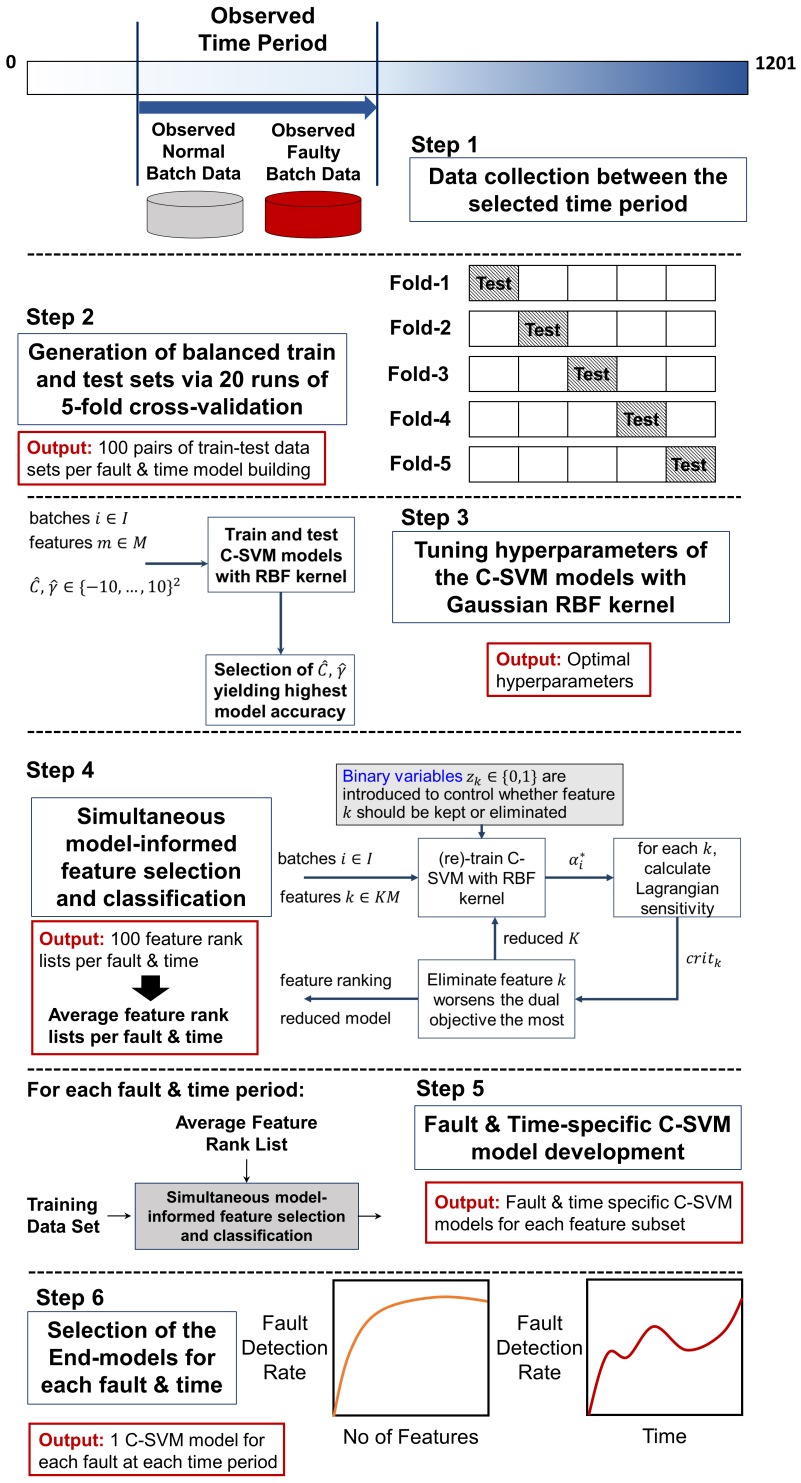


Figure 4.3: Offline phase of the proposed simultaneous fault detection and diagnosis framework.

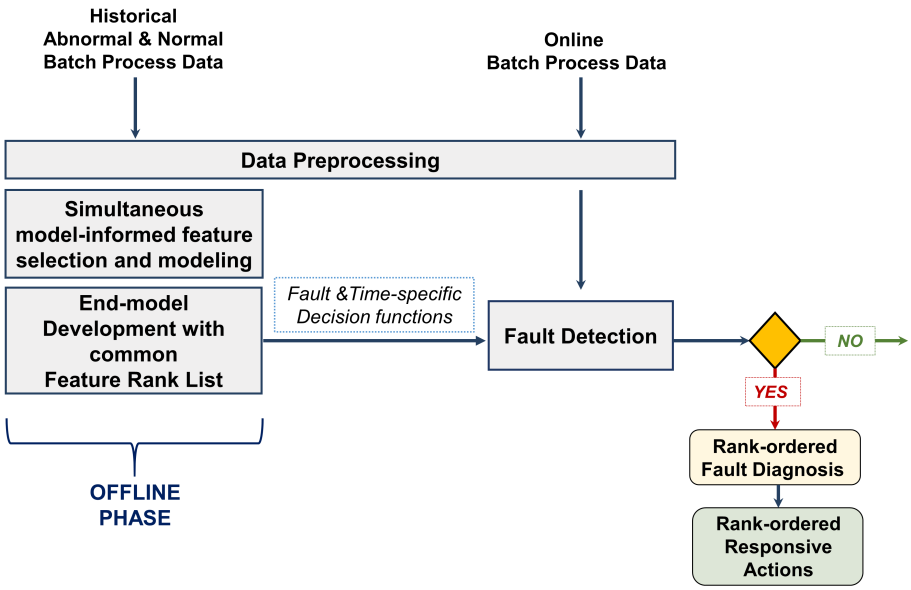


Figure 4.4: Online phase implementation for simultaneous fault detection and diagnosis.

4.3 Case Studies

We have performed three sets of experiments to test the performance of the proposed data-driven framework: (i) one-step rolling time horizon analysis, where fault detection and diagnosis models are built for each individual sample bin; (ii) two-step rolling time horizon analysis, in which models are developed for every two sliding sample bins; and (iii) evolving time analysis, where we build models by using both available data within the selected sample bin and set of informative features from all previous sample bins. Below we describe each of the analysis, and provide corresponding results. The performance of the developed models is assessed via (i) fault detection rate, (ii) accuracy, (iii) false alarm rate, and (iv) Area Under the Receiver Operating Characteristic (ROC) Curve, (AUC). The ideal models would have perfect Area Under the ROC curve, accuracy, as well as fault detection rate being 1 with minimum False Alarm Rate (ideally, 0).

4.3.1 One-Step Rolling Time Horizon Approach

Here, we inspect the status of a given batch with a total of 1800 fault and time-specific end-models, for 15 different faults (Table 4.2) at each of the 120 sample bins. We only consider

the process data within each selected sample bin time period, and exclude any information from previous sample bins. Since, batch monitoring is performed within each sample bin individually, we refer this approach as one-step rolling time horizon approach.

Initial step in the one-step rolling time horizon approach is to gather the pre-processed data sets relevant to each sample bin. Here, the features of the data sets include the process variable measurements as well as slope, standard deviation and mean of each 13 process variables within each sample bin of size 10. This results in 169 features per bin, where 130 of them belong to actual measurements and 39 characterizes the process behavior. On the other hand, the number of instances, which is the number of normal and faulty batches, varies along the time horizon. In each sample bin, we include only the normal and faulty batch data that have been observed between the time period of the selected sample bin. In particular, we have 400 normal and 1000-2000 faulty batches per fault (Table 4.2). When building models with one-step rolling time horizon approach, we select the faulty batches among the batches where the fault has already been introduced before the initial time point of the selected sample bin. Then, as mentioned in Section 4.2.2, we form balanced training and testing datasets including equal amount of normal and faulty batches for each model development. We have less faulty batches in the beginning of the process operation, therefore the dataset used to train a model for an early time period is smaller compared to later. In other words, the data set size changes as we move along the time horizon due to the change in the number of batches with varying fault onset time.

We have demonstrated 3 faults with varying level of difficulty in detection at the selected sample bins to evaluate the performance of the one-step rolling time horizon approach (Table 4.3). Fault 7 (change in feed temperature) is reported to be the easiest fault to detect due to presence of feed temperature measurements in the batch process data set. Fault 8 (pH sensor drift) poses moderate difficulty to be detected, whereas Fault 15 (contamination) is defined as one of the most challenging faults of the adopted dataset (180). Table 4.3 reports the selected fault and time specific models (end-models) with their corresponding optimal feature subset size, fault detection rate, model accuracy, AUC, and false alarm rate. When a fault is detected with these models, the corresponding optimal feature subset reveals the explicit process measurements (and/or features describing process behavior) for diagnosis of the detected fault. For particular models, that are marked with asterisks, we have provided a second alternative model, where we are able to attain

almost equivalent fault detection rate with significantly lower optimal feature subset size (Table 4.3). This consequently facilitates the isolation and correction of the detected fault by ensuing optimal sensor placement (i.e. sensor network design). Furthermore, obtaining the maximum model performance with minimum number of features is favorable in terms of computational efficiency and simplicity. Figure 4.5 shows an example for such cases through Fault 8 at Sample Bin 12.

The models built via one-step rolling time horizon approach reveals perfect detection rate, accuracy for Fault 7 with zero false alarm rate and small optimal feature subset size. As the detection difficulty increases (from Fault 7 to Fault 15), we observe that the early time-specific models (prior to Sample bin 20) have shown relatively high false alarm rates. This may stem from the relatively low number of faulty batches in the early periods of the process simulation, and can be overcome by simulating additional batches with earlier fault onset time. On the other hand, the performance of the models of the later sample bins improve with the inclusion of more batch data. False alarm rates immediately become ideal by approaching to 0 later in the batch process regardless of the difficulty level of the fault detection.

Another observation is that the increase in the fault complexity is reflected with an increase in the average feature subset size across the all time-specific models of each fault.

Figure 4.6 depicts the fault detection rate of all fault and time-specific models built for each feature subset. Among them, the end-model is the one yielding the highest fault detection rate, highlighted with a red line, where corresponding feature subset size, model accuracy and false alarm rate are reported. As mentioned before, the results with asterisks imply the existence of an alternative model with almost equivalent fault detection rate with considerably lower feature subset size, which are listed in Table 4.3.

Table 4.3: One-step rolling time horizon analysis: Selected model performances for Fault 7 (easy), 8 (moderate), and 15 (challenging).

Fault	Sample Bin	Fault Detection Rate	Optimal Feature Subset Size	Accuracy	False Alarm Rate	AUC
Fault-7 Feed Temperature Change	20	0.98	1	0.99	0.00	1.00
	40	0.98	1	0.99	0.00	0.99
	60	1.00	3	1.00	0.00	1.00
	80	1.00	3	1.00	0.00	1.00
	100*	1.00	3	1.00	0.00	1.00
	100	1.00	1	1.00	0.00	1.00
	120*	1.00	9	1.00	0.00	1.00
	120	1.00	1	1.00	0.00	1.00
Fault-8 pH Sensor Drift	20	0.70	16	0.77	0.15	0.84
	40*	0.94	9	0.97	0.00	0.99
	40	0.94	2	0.97	0.00	0.99
	60*	0.96	4	0.98	0.00	1.00
	60	0.96	2	0.98	0.00	1.00
	80*	0.95	34	0.97	0.00	0.99
	80	0.95	2	0.97	0.00	0.99
	100*	0.96	27	0.98	0.00	1.00
	100	0.96	25	0.98	0.00	1.00
	120*	0.97	28	0.98	0.00	1.00
	120	0.97	6	0.98	0.00	1.00
Fault-15 Contamination	20	0.74	99	0.61	0.53	0.65
	40*	0.81	31	0.88	0.04	0.94
	40	0.81	28	0.88	0.04	0.94
	60*	0.97	43	0.98	0.01	0.99
	60	0.97	29	0.98	0.01	0.99
	80*	0.98	68	0.99	0.00	1.00
	80	0.98	10	0.99	0.00	1.00
	100*	0.98	56	0.99	0.00	1.00

Table 4.3 – Continued

Fault	Sample Bin	Fault Detection Rate	Optimal Feature Subset Size	Accuracy	False Alarm Rate	AUC
	100	0.98	35	0.99	0.00	1.00
	120*	1.00	10	1.00	0.00	1.00
	120	1.00	8	1.00	0.00	1.00

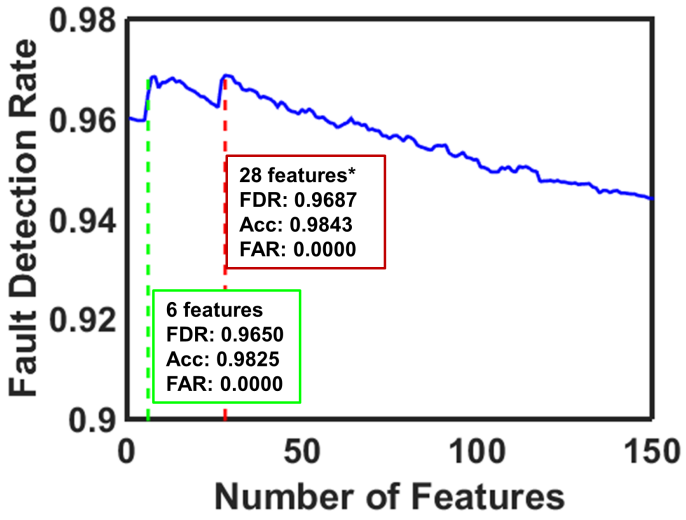


Figure 4.5: Selecting the optimal feature subset for detection and diagnosis of Fault 8 at Sample Bin 120. (FDR: Fault Detection Rate, Acc: Accuracy, FAR: False Alarm Rate)

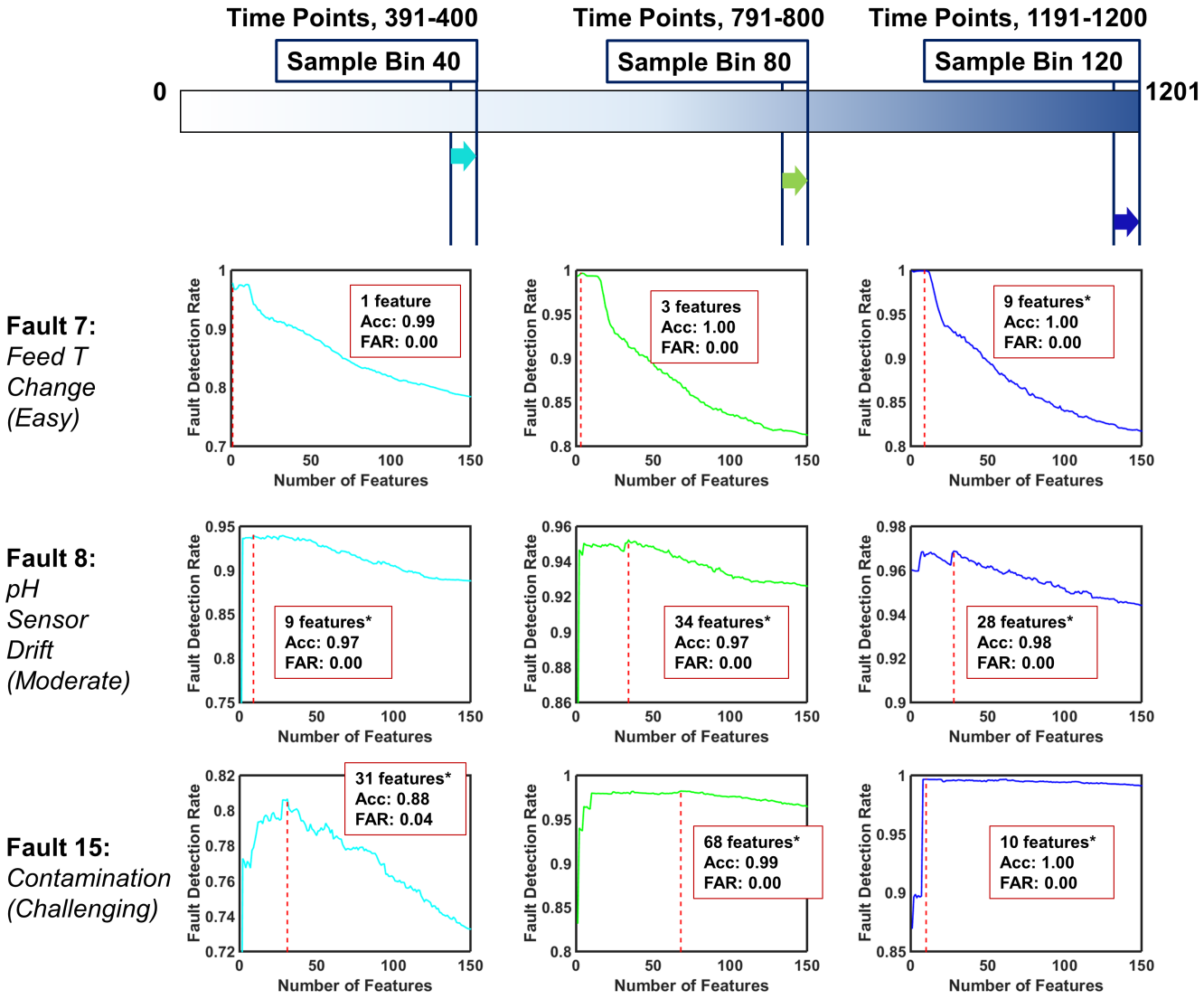


Figure 4.6: One-step rolling time horizon analysis: Detection rates of 3 faults with increasing level of difficulty in detection; Fault 7,8, and 15 respectively. The optimal number of features, detection accuracy and false alarm rate are provided at the highest detection rate. The results with asterisks indicate the existence of an alternative model producing almost equivalent performance with less features. Alternative models are provided in Table 4.3.

4.3.2 Multi-Step Rolling Time Horizon Approach

In this approach, we monitor a given batch with a total of 1785 fault and time-specific end-models throughout the process. The models examine for 15 separate faults at every two successive sample bins in a sliding manner which corresponds to 119 checks. In addition to the process variable measurements of the later sample bin, we consider set of features from the former sample bin that characterize process behavior. In this case, we have exploited data of one previous sample bin

in addition to the selected sample bin for each time-specific model development. This has enabled us to analyze the 3-dimensional batch process data via two-step rolling time horizon approach. Particularly, this approach can be extended with addition of further previous sample bins, therefore we call it as multi-step rolling time horizon approach.

In this case, similar to one-step rolling time horizon approach, initial step is to arrange two-step sample bins and collect the pre-processed data sets. We have two-fold feature subsets in this approach, one set identifying the former sample bin, and later defining the recent process progress. The former feature subset contains solely the process behavior characterizing features; slope, standard deviation and mean of 13 process variable measurements within that previous sample bin, resulting in 39 features. The later feature subset consists of the process measurements as well as slope, standard deviation and mean of each 13 process variables within the latest sample bin, resulting in an additional 169 features per bin. As a result, we incorporate 208 features in each time-specific model development with two-step rolling time horizon approach. Here, similar to the one-step rolling time horizon approach, the number of batches involved during model development changes according to time due to the variation in fault onset time.

Table 4.4 tabulates the selected fault and time specific models (end-models) with the corresponding performance. Similarly, the alternative model results are listed in Table 4.4.

Similar to the previous approach, we observe that fault detection rates improve with time for all faults, and the optimal feature subset size increases as the faults become more challenging to detect. Finally, we notice a significant improvement in the fault detection rate of Fault 15 time-specific models after Sample Bins 18-20 compared to the models obtained via one-step rolling time horizon approach, which indicates that the added features have helped models to gain more insight to detect this challenging fault.

Figure 4.7 demonstrates the fault detection rate of the entire fault and time specific models built for each feature subset. The end-models are shown with red-dashed lines.

Table 4.4: Two-step rolling time horizon analysis: Selected model performances for Fault 7 (easy), 8 (moderate), and 15 (challenging).

Fault	Sample Bin	Fault Detection Rate	Optimal Feature Subset Size	Accuracy	False Alarm Rate	AUC
Fault-7 Feed Temperature Change	19 – 20*	0.83	21	0.92	0.00	1.00
	19-20	0.83	10	0.91	0.00	1.00
	39-40	1.00	1	1.00	0.00	1.00
	59-60	1.00	1	1.00	0.00	1.00
	79 – 80*	1.00	2	1.00	0.00	1.00
	79-80	1.00	1	1.00	0.00	1.00
	99 – 100*	1.00	6	1.00	0.00	1.00
	99-100	1.00	1	1.00	0.00	1.00
	119-120	0.75	1	0.88	0.00	1.00
Fault-8 pH Sensor Drift	19 – 20*	0.83	52	0.85	0.13	0.93
	19-20	0.83	49	0.85	0.12	0.93
	39-40	0.98	4	0.99	0.00	1.00
	59 – 60*	0.97	3	0.99	0.00	1.00
	59-60	0.97	2	0.98	0.00	1.00
	79 – 80*	0.98	32	0.99	0.00	1.00
	79-80	0.98	9	0.99	0.00	1.00
	99 – 100*	0.98	66	0.99	0.00	1.00
	99-100	0.98	3	0.99	0.00	1.00
	119 – 120*	0.99	70	1.00	0.00	1.00
	119-120	0.99	1	0.99	0.00	1.00
Fault-15 Contamination	19 – 20*	0.57	204	0.58	0.41	0.61
	19-20	0.57	201	0.58	0.41	0.61
	39-40	0.94	58	0.96	0.02	0.99
	59 – 60*	0.99	57	1.00	0.00	1.00
	59-60	0.99	18	0.99	0.01	1.00
	79 – 80*	0.99	72	1.00	0.00	1.00
	79-80	0.99	42	0.99	0.00	1.00

Table 4.4 – Continued

Fault	Sample Bin	Fault Detection Rate	Optimal Feature Subset Size	Accuracy	False Alarm Rate	AUC
	99 – 100*	1.00	77	1.00	0.00	1.00
	99-100	1.00	5	1.00	0.00	1.00
	119 – 120*	1.00	145	1.00	0.00	1.00
	119-120	1.00	8	1.00	0.00	1.00

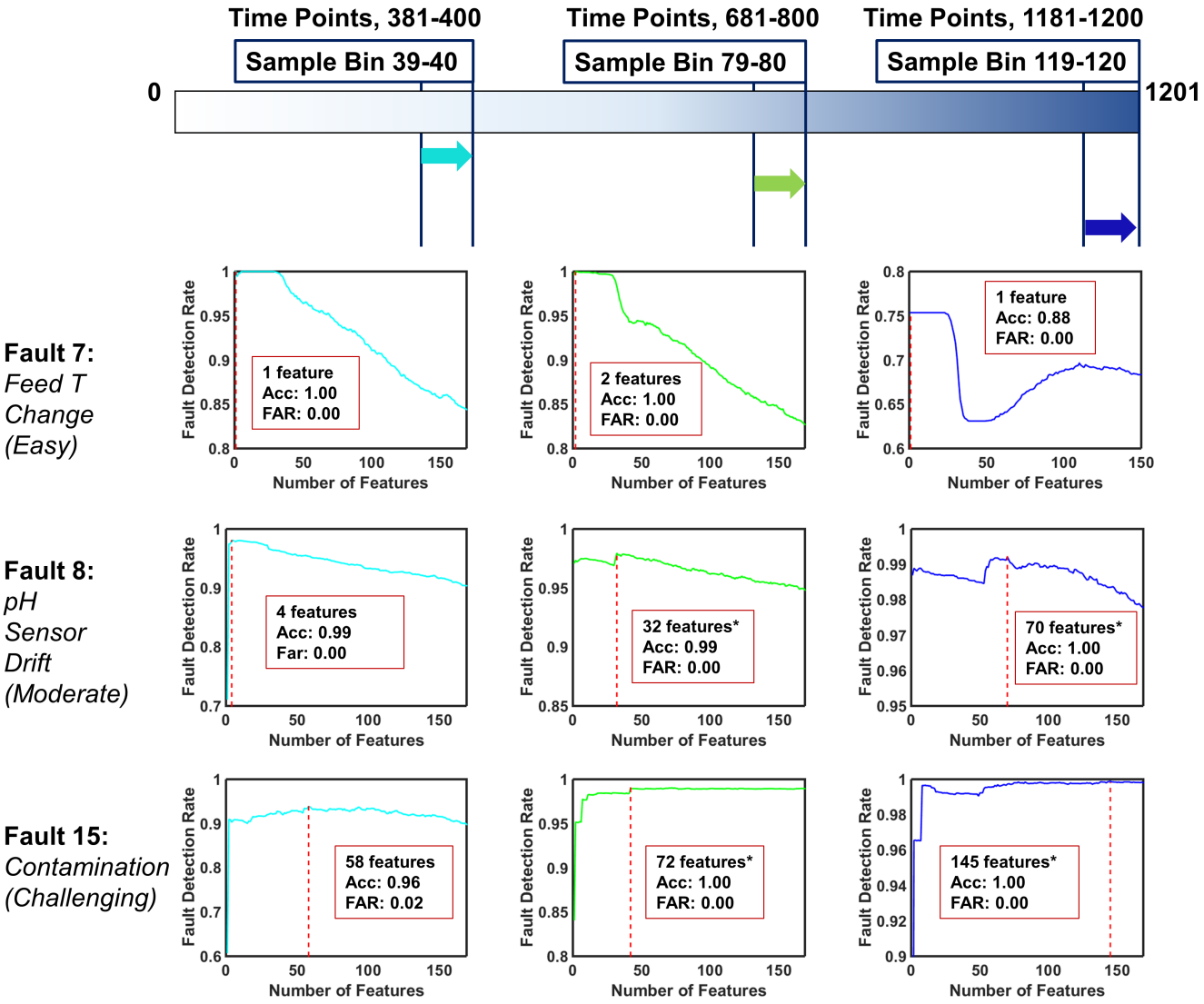


Figure 4.7: Two-step rolling time horizon analysis: Detection rates of Fault 7,8, and 15. The optimal number of features, detection accuracy and false alarm rate are provided at the highest detection rate. The results with asterisks indicate the existence of an alternative model producing almost equivalent performance with less features. Alternative models are provided in Table 4.4.

4.3.3 Evolving Time Horizon Approach

As a third and final case study, we adopt evolving time horizon based approach, where we build models by making use of the entire historical process data until the last time point of the selected sample bin for analysis. Here, we monitor a given batch with a total of 1800 fault and time-specific end-models where the given batch is scanned for 15 different faults (Table 4.2) at each of the 120 sample bins. We train models by exploiting the entire historical batch process data at each sample

bin, therefore we refer this approach as evolving time horizon approach.

This approach is equivalent to the multi-step rolling time horizon approach, when one incorporates the process behavior features from the entire previous sample bins, instead of only one. This corresponds to the collection of slope, standard deviation and mean of 13 process variable measurements from all previous and the most recent sample bins as well as the actual process measurements observed within the latest sample bin. Consequently, the data set size grows significantly as we move along the time horizon, where the number of features range between $\sim 170 - 4400$.

Table 4.5 exhibits the chosen fault and time specific models for the online phase (end-models) obtained via evolving time horizon based analysis. Compared to the other time horizon approaches, the fault detection rates attained with this approach for Fault 8 and 15 either remain same or slightly improve. On the other hand, the model performances for Fault 7 time-specific models significantly deteriorate. This may be an indicator of the increased noise level in the data. In other words, aggregation of the entire historic process data has possibly elevated the amount of redundant features, consequently deteriorated the performance of the models for Fault 7 (the easiest fault to detect). Another remarkable observation is the significant increase in the optimal feature subset sizes for Fault 7 and 15, which makes the evolving time horizon approach not preferable compared to the other two approaches.

Figure 4.8 shows the fault detection rate of all fault and time specific models built for each feature subset. The end-models are highlighted with red dashed lines.

Table 4.5: Evolving time horizon analysis: Selected model performances for Fault 7 (easy), 8 (moderate), and 15 (challenging).

Fault	Sample Bin	Fault Detection Rate	Optimal Feature Subset Size	Accuracy	False Alarm Rate	AUC
Fault-7 Feed Temperature Change	20	0.80	143	0.89	0.02	0.97
	40*	0.81	407	0.88	0.05	0.95
	40	0.81	406	0.88	0.04	0.95
	60*	0.77	348	0.87	0.02	0.95
	60	0.77	320	0.87	0.03	0.95
	80*	0.79	401	0.88	0.04	0.95
	80	0.79	233	0.88	0.03	0.94
	100	0.80	721	0.87	0.06	0.94
	120*	0.83	433	0.90	0.03	0.96
	120	0.83	427	0.89	0.04	0.96
Fault-8 pH Sensor Drift	20	0.79	93	0.83	0.12	0.91
	40*	0.90	20	0.95	0.00	0.99
	40	0.90	12	0.95	0.00	0.99
	60*	0.96	89	0.98	0.00	1.00
	60	0.96	82	0.98	0.00	1.00
	80*	0.97	87	0.99	0.00	1.00
	80	0.97	4	0.98	0.00	0.99
	100*	0.98	13	0.99	0.00	0.99
	100	0.98	7	0.99	0.00	0.99
	120*	1.00	17	1.00	0.00	1.00
	120	1.00	5	1.00	0.00	1.00
Fault-15 Contamination	20	0.66	1	0.50	1.00	1.00
	40*	0.92	226	0.95	0.02	0.99
	40	0.92	172	0.95	0.02	0.99
	60*	0.99	523	0.99	0.00	1.00
	60	0.99	268	0.99	0.00	1.00
	80*	0.99	496	1.00	0.00	1.00

Table 4.5 – Continued

Fault	Sample Bin	Fault Detection Rate	Optimal Feature Subset Size	Accuracy	False Alarm Rate	AUC
	80	0.99	323	0.99	0.00	1.00
	100*	0.99	684	0.99	0.00	1.00
	100	0.99	84	0.99	0.00	1.00
	120*	1.00	193	1.00	0.00	1.00
	120	1.00	55	1.00	0.00	1.00

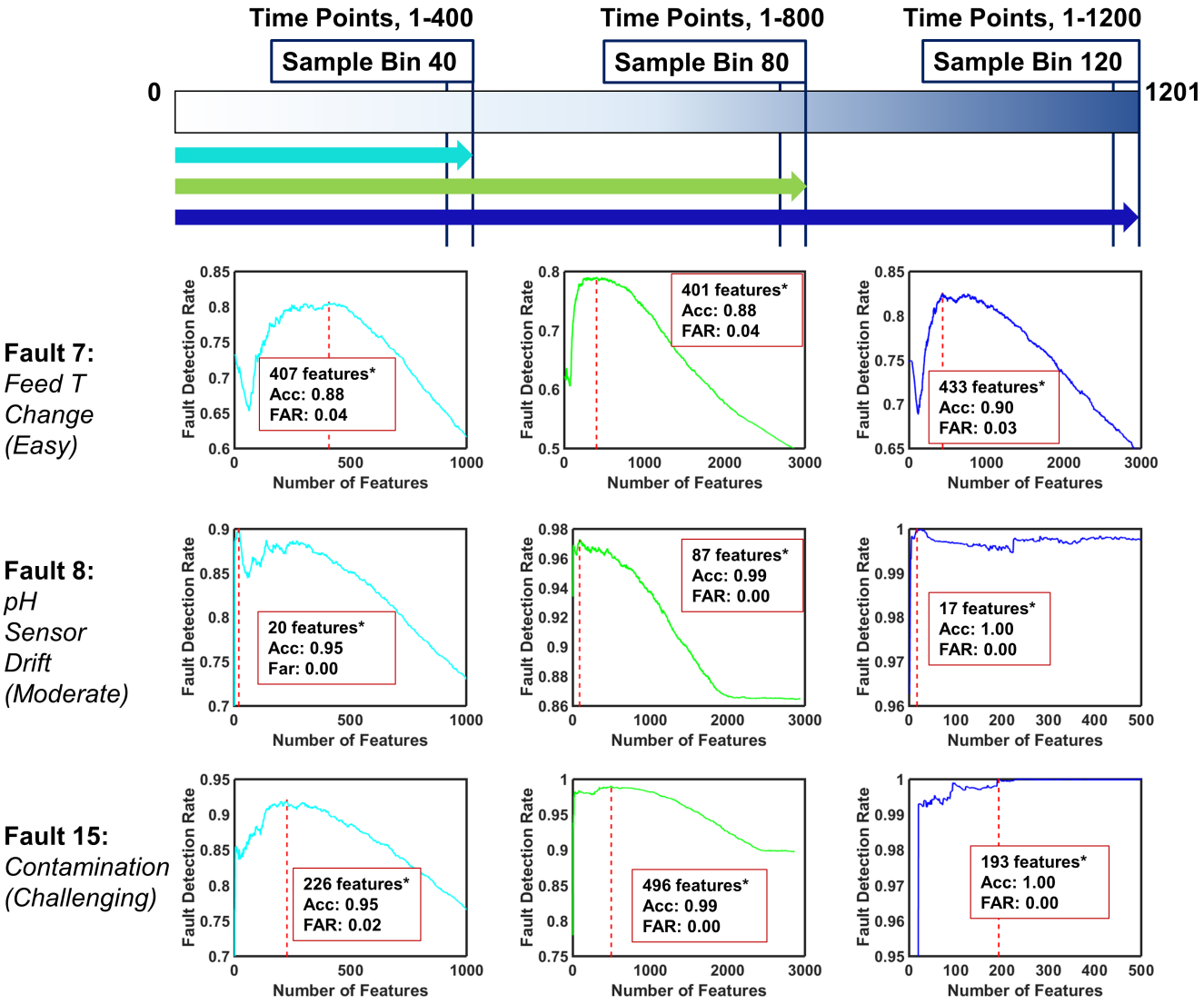


Figure 4.8: Evolving time horizon analysis: Detection rates of Fault 7,8, and 15. The optimal number of features, detection accuracy and false alarm rate are provided at the highest detection rate. The results with asterisks indicate the existence of an alternative model producing almost equivalent performance with less features. Alternative models are provided in Table 4.5.

4.3.4 Comparison

Here, we compare the performance of the fault and time-specific models built via three distinct time horizon approaches through Fault 7,8, and 15 (three faults with varying level of difficulty to detect). Figure 4.9 depicts the variation of fault detection rate with respect to time (i.e. sample bin) for one-step rolling, two-step rolling, and evolving time horizon approaches. The fault detection rate versus time plots for all faults are provided in Figure 4.10.

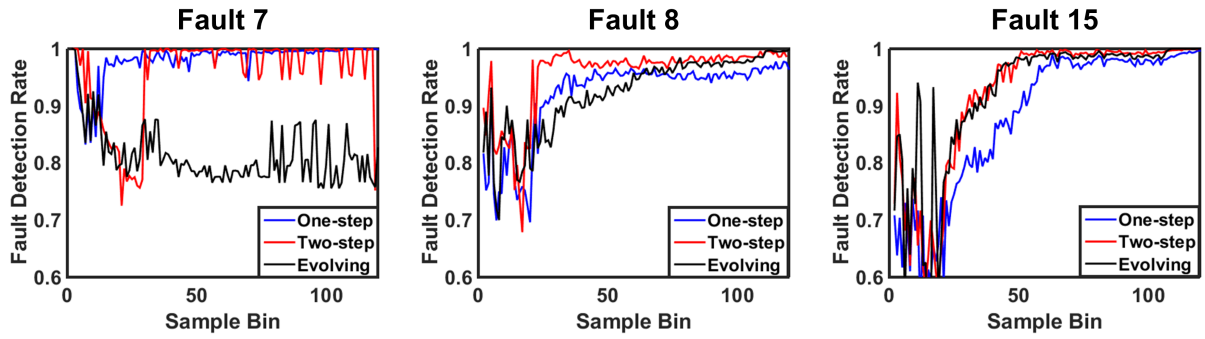


Figure 4.9: Comparison of the fault detection rates for Fault 7,8, and 15 along the batch process with respect to one-step rolling, two-step Rolling, and evolving time horizon approaches.

As one can notice, performance of different time horizon approaches are not consistent across the all faults; they are fault-specific. However, it is evident that as the detection difficulty level increases, two-step rolling and evolving time horizon analysis performs better than the other approach, which prevails the increasing significance of historical data usage for more challenging faults.

One major observation is the relatively low fault detection rate and accuracy in early time models among all time horizon approaches. This is mainly due to the faulty batch data scarcity in early time models (until sample bin 30), which implies low number of simulated batches with fault in early periods of the process due to varying fault onset times. Such scarcity affects the number of batches included during the model development in the offline phase. In early time models, the number of faulty batches is limiting case. However, the situation is reversed as we move along the time horizon where the number of simulated batches, in which fault has been introduced by then, exceed the number of simulated batches under normal conditions. In order to be consistent during model development and learn both class equally, we equate the number of normal and faulty batches before we train our fault detection and diagnosis models. This leads us to use of datasets with varying sizes along the process for different time-specific models. The size of the adopted dataset changes in two ways: (i) number of batches involved, and (ii) number of extracted features. The afore-mentioned fluctuation in the number of simulated faulty batches along the simulation alters the dataset dimension by changing the number of batches involved. On the other hand, the number of features changes with the adopted time horizon approach. Particularly, the smallest

dataset used for model development belongs to the first sample bin of one-step rolling time horizon analysis (Sample Bin 2), where we train our fault-specific models with 12 faulty and 400 normal batches, and 142 features (after eliminating the ones with less than 10^{-8} standard deviation among 169 features). Whereas the largest dataset is observed at the final sample bin of the evolving time horizon approach, where we have 2000 faulty and 400 normal batches, and 4381 features.

Another possible reason for relatively lower performance of the models in early time of the process is the change in the operation mode. The process begins in a batch mode and switches to fed-batch mode once the substrate (i.e. glucose) concentration is nearly depleted. Since we have no flows into the bioreactor, thus no information of feed flow rate in batch mode as we do in fed-batch mode, the number of features contributing model development during that period decreases. This may have also caused the decrease in model performance in early time of the process.

During the detection of a moderate fault, Fault 8, we observe that all three time horizon approaches perform similar in early time models, where two-step rolling time horizon technique becomes superior in later stages. Whereas for the detection of a challenging fault, Fault 15, evolving time horizon approach competes with two-step rolling time horizon, and indeed dominates it in early time models in terms of the fault detection rate. Yet, for the sake of computational efficiency and simplicity, selection of the end-models via two-step rolling time horizon approach would be favorable over the end-models assessed by the evolving time horizon approach. Regardless of the fault and the time horizon approach, fault detection rates of early time models need to be improved and this can be achieved with further set of simulations where faults can be introduced earlier.

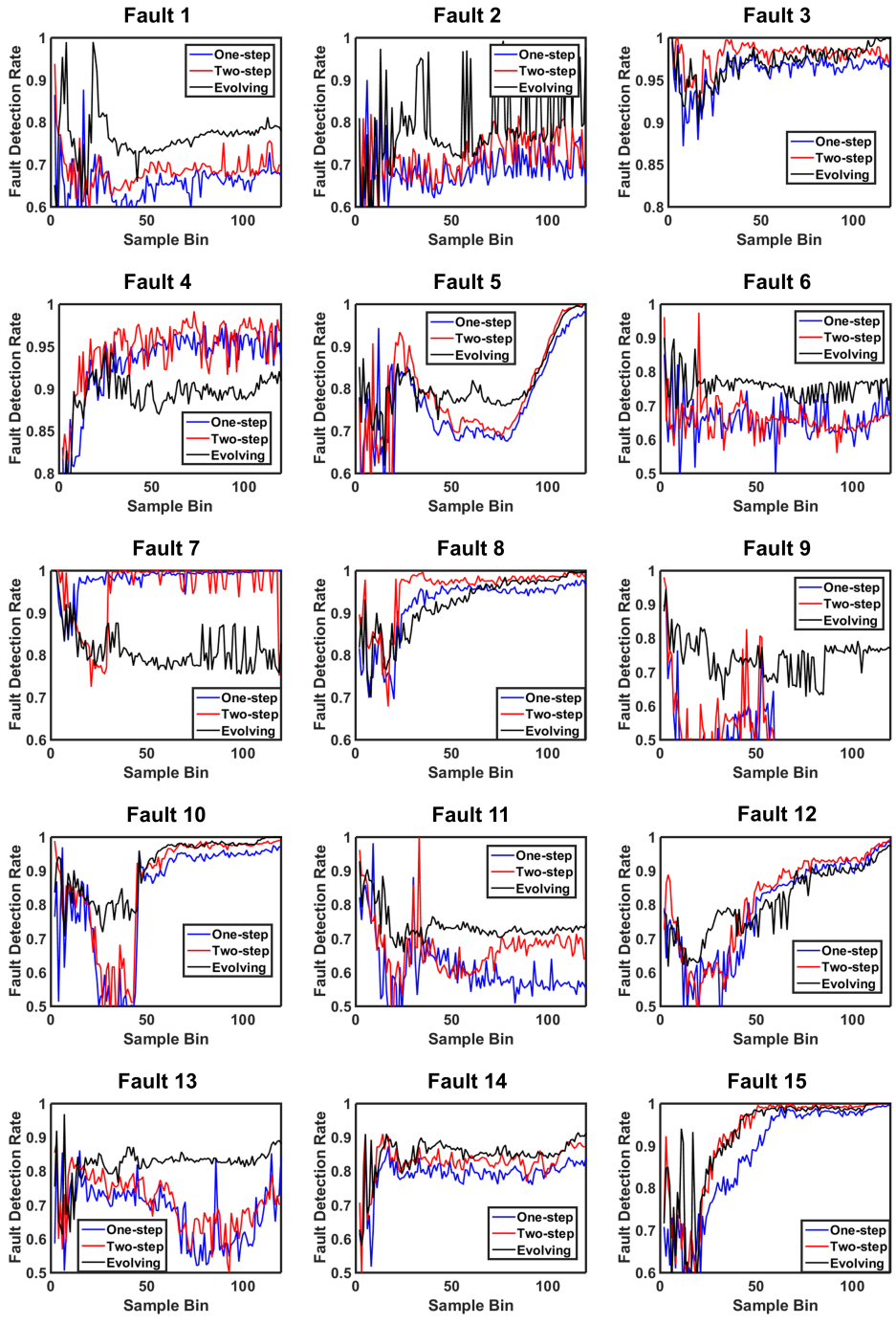


Figure 4.10: Comparison of the fault detection rates along the batch process with respect to one-step Rolling, two-step Rolling, and evolving time horizon approaches.

Finally, we present a comparative analysis for fault diagnosis through Fault 8. The diagnosis is provided for the Sample Bin 40, 80, and 120 models formed via one-step rolling (4.11), two-step rolling (4.12), and evolving (4.13) time horizon approaches. The color codes represent the Sample Bin index in Figures 4.6, 4.7, and 4.8, where cyan, green and blue represents Sample Bin 40, 80, and 120 models, respectively. Of note, for the cases where we have superior alternative models, we have adopted their optimal feature subset for fault diagnosis. The explicit list of features (process measurements and/or process behavior describing features) for fault diagnosis with the three time horizon approaches are given in Tables 4.6,4.7,and 4.8, respectively.

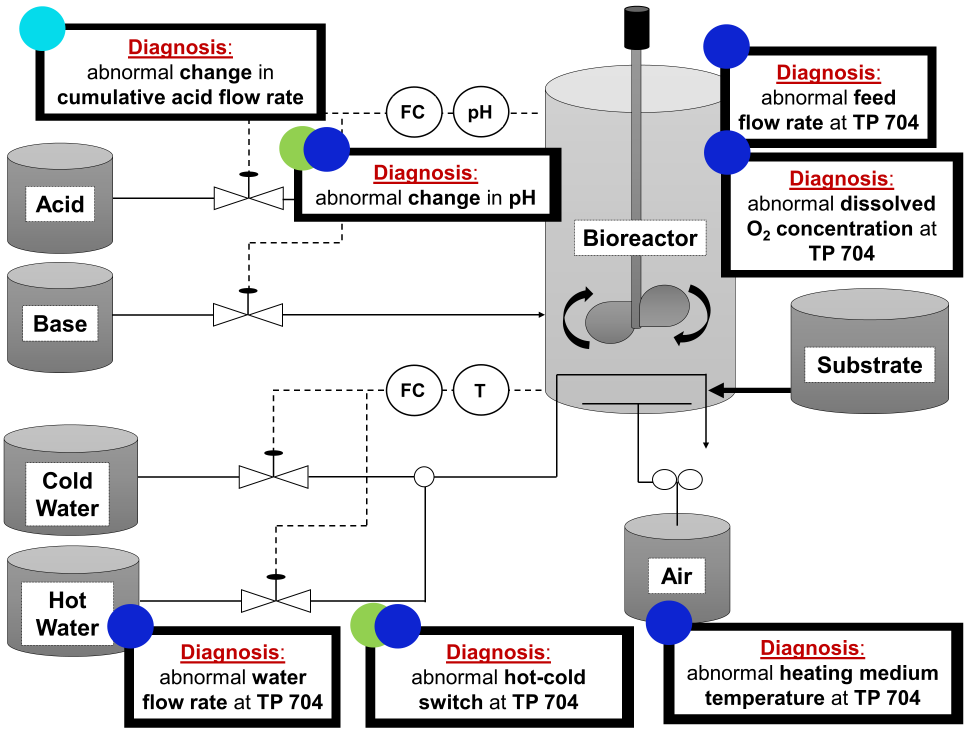


Figure 4.11: Diagnosis of Fault 8 (pH sensor drift) with one-step rolling time horizon based approach at selected sample bins. The color codes indicate the sample bin index which is compatible with the Figure 4.6.

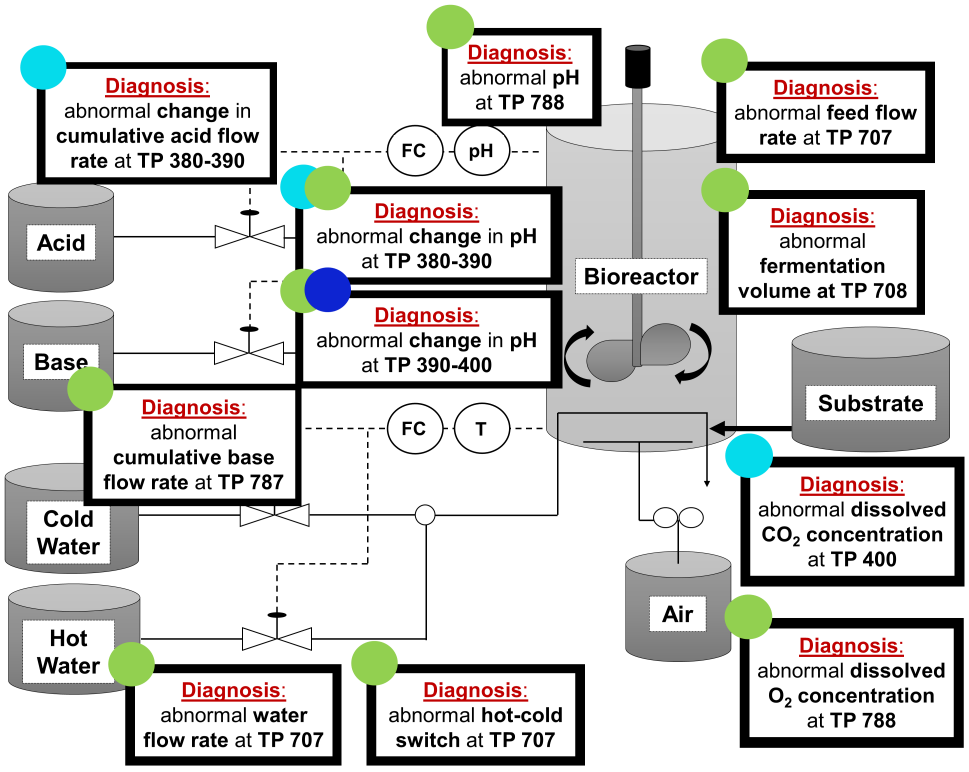


Figure 4.12: Diagnosis of Fault 8 (pH sensor drift) with two-step rolling time horizon based approach at selected sample bins. The color codes indicate the sample bin index which is compatible with the Figure 4.7.

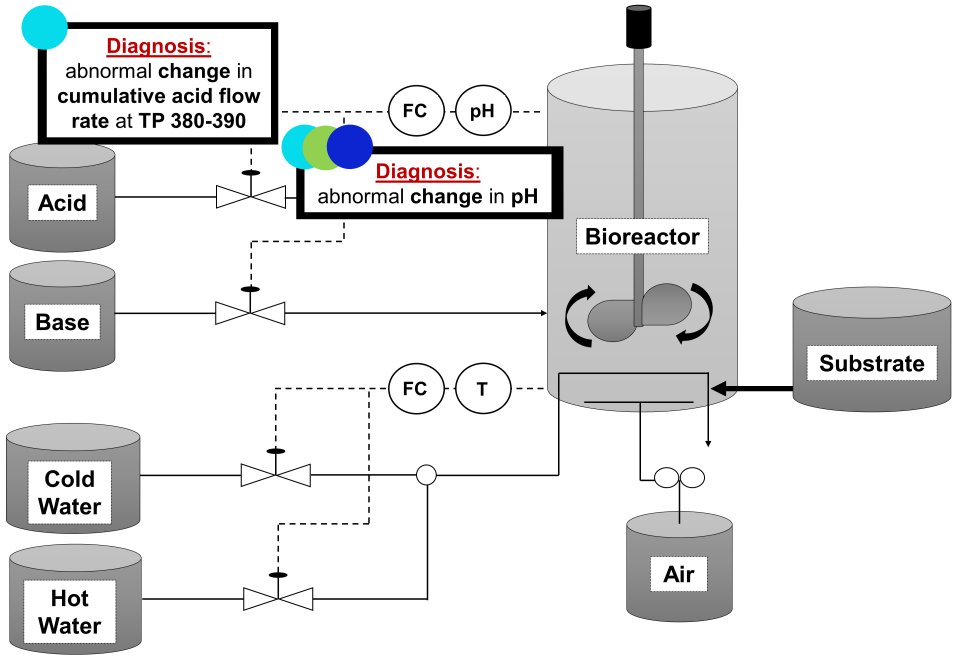


Figure 4.13: Diagnosis of Fault 8 (pH sensor drift) with evolving time horizon based approach at selected sample bins. The color codes indicate the sample bin index which is compatible with the Figure 4.8.

4.4 Conclusions

In this work, we have presented the application of a feature selection algorithm based on nonlinear Support Vector Machine formulations and applied it for fault detection and diagnosis in batch processes. The proposed framework can easily be implemented as an online decision support tool. We have performed 3 sets of experiment to assess the performance of the proposed framework: (i) one-step rolling time horizon basis analysis, (ii) two-step rolling time horizon basis analysis, and (iii) evolving time horizon analysis, in which we change the amount of historical data incorporation during the offline (model development) phase. The results show that the selection of time horizon approach is specific to fault characteristics; whereas, for moderate and challenging faults, two-step rolling or evolving time horizon based analysis is favorable. Specifically, evolving time horizon based analysis has degraded the detection rates of Fault 7, which is a fault reported to be the easiest to detect. This shows that the inclusion of additional historical process information during model development does not necessarily improve accuracy of the fault detection models. On

Table 4.6: Diagnosis of Fault 8 at Sample Bin 40 via one-step rolling, two-step rolling, and evolving time horizon based analysis.

Approach	**Sample Bin 40**			
	Feature Rank	Feature Type	Corresponding Time Point	Feature Name
One-step Rolling Time Horizon	1	HPB	391-400	Slope of Cumulative Acid Flow Rate
	2	HPB	391-400	Standard deviation in Cumulative Acid Flow Rate
Two-step Rolling Time Horizon	1	HPB	381-390	Slope of Cumulative Acid Flow Rate
	2	HPB	381-390	Standard deviation in Cumulative Acid Flow Rate
	3	HPB	381-390	Mean of Cumulative Acid Flow Rate
	4	CPV	400	Dissolved CO_2 Concentration
Evolving Time Horizon	1	HPB	51-60	Mean of pH
	2	HPB	31-40	Mean of pH
	3	HPB	61-70	Mean of pH
	4	HPB	71-80	Slope of Cumulative Acid Flow Rate
	5	HPB	71-80	Standard deviation in Cumulative Acid Flow Rate
	6	HPB	141-150	Mean of Cumulative Acid Flow Rate
	7	HPB	151-160	Mean of Cumulative Acid Flow Rate
	8	HPB	81-90	Slope of Cumulative Acid Flow Rate
	9	HPB	161-170	Mean of Cumulative Acid Flow Rate
	10	HPB	171-180	Mean of Cumulative Acid Flow Rate
	11	HPB	131-140	Mean of Cumulative Acid Flow Rate
	12	HPB	61-70	Slope of Cumulative Acid Flow Rate

Table 4.7: Diagnosis of Fault 8 at Sample Bin 80 via one-step rolling, two-step rolling, and evolving time horizon based analysis.

Approach	**Sample Bin 80**			
	Feature Rank	Feature Type	Corresponding Time Point	Feature Name
One-step Rolling Time Horizon	1	HPB	791-800	Mean of pH
	2	HPV	794	Hot/cold Switch
Two-step Rolling Time Horizon	1	HPB	781-790	Mean of pH
	2	HPB	791-800	Mean of pH
	3	HPV	787	Hot/cold Switch
	4	HPV	787	Feed Rate
	5	HPV	788	Fermentation Volume
	6	HPV	788	Dissolved O_2 Concentration
	7	HPV	788	pH
	8	HPV	787	Cumulative Base Flow Rate
	9	HPV	787	Cooling/heating Medium Flow Rate
Evolving Time Horizon	1	HPB	300-310	Standard deviation in Cumulative Acid Flow Rate
	2	HPB	300-310	Slope of Cumulative Acid Flow Rate
	3	HPB	290-300	Standard deviation in Cumulative Acid Flow Rate
	4	HPB	290-300	Slope of Cumulative Acid Flow Rate

Table 4.8: Diagnosis of Fault 8 at Sample Bin 120 via one-step rolling, two-step rolling, and evolving time horizon based analysis.

Approach	**Sample Bin 120**			
	Feature Rank	Feature Type	Corresponding Time Point	Feature Name
One-step Rolling Time Horizon	1	HPB	1190-1200	Mean of pH
	2	HPV	1194	Feed Rate
	3	HPV	1194	Cooling/heating Medium Flow Rate
	4	HPV	1194	Hot/cold Switch
	5	HPV	1194	Dissolved O_2 Concentration
	6	HPV	1194	Heating Medium Temperature
Two-step Rolling Time Horizon	1	HPB	1191-1200	Mean of pH
Evolving Time Horizon	1	HPB	300-310	Standard deviation in Cumulative Acid Flow Rate
	2	HPB	300-310	Slope of Cumulative Acid Flow Rate
	3	HPB	290-300	Standard deviation in Cumulative Acid Flow Rate
	4	HPB	290-300	Slope of Cumulative Acid Flow Rate
	5	HPB	280-290	Standard deviation in Cumulative Acid Flow Rate

the contrary, the additional features may become redundant and increase the amount of noisy data, which subsequently deteriorate the model performance. Furthermore, even evolving time horizon based analysis have produced slightly better performance for Fault 15, one of the most challenging faults to detect, at specific sample bins, the increase in the optimal feature (process measurement) subset size may render them unfavorable for the sake of simplicity. Nevertheless, one can select and implement the fault and time-specific models from any of the three time horizon approaches that yields the highest fault detection rate with optimal number of features for simultaneous fault detection and diagnosis.

The major contribution of the proposed framework is the establishment of accurate and simultaneous fault detection and diagnosis in batch processes by virtue of a feature selection algorithm based on nonlinear SVM formulation backed by global optimization theory. Most of the existing prevalent data-driven methods for fault detection are based on feature extraction techniques which do not explicitly reveal the explicit process variables for fault diagnosis. With the proposed framework, we produce fault and time specific end-models which enable not only accurate detection of the faults but also simultaneously provide diagnosis of the detected fault by listing the most informative process measurements. The implementation of the end-models as an online decision support tool can (i) enable early intervention to the process to reverse the detected fault, (ii) significantly reduce the number of sensor measurements to diagnose the detected fault, and (iii) possibly

guide for the optimal sensor placement (i.e sensor network design). This subsequently may increase process efficiency, safety, and profitability, which is the ultimate goal of the modern process industry.

Finally, in this work, we have focused on training 2-class models where one can access historical and/or simulation-based process data. In future work, we aim to extend the presented framework by developing one-class and multi-class classification techniques with SVM formulations for simultaneous fault detection, and diagnosis. Specifically, one-class classification techniques with SVM formulations with the presented feature selection algorithm is expected to handle cases where the historical industry data is unbalanced with normal and faulty cases, and process simulations are computationally expensive.

5. INTEGRATING MAINTENANCE OPTIMIZATION AND CONTROL WITH S-FDD FRAMEWORK

Once a fault is detected and diagnosed in a process, operators need to decide on ensuing actions rapidly to maintain a safe and profitable operation. To do this, the common practice in industry is to adopt several different maintenance strategies to delay if possible prevent the upcoming failures, as well as to react to recover the process from faulty operation. With the emergence of Industry 4.0 (12) and Smart Manufacturing (89) initiatives, mechanization and automation in industry has become prevalent, which consequently has reduced the number of production personnel and has increased the capital employed in production equipment and civil structures (90). This has resulted in an increase in the fraction of employees working in maintenance area along with an increase in the fraction of maintenance spending on the total operational costs. Today, the numbers show that maintenance spending can be the second largest part of the operational budget following after energy costs (90). Knowing the significance and prevalence of use of maintenance strategies in industry, the goal for this section of the dissertation is to study different maintenance strategies (i.e corrective and preventive) using a benchmark chemical process and then integrate them to simultaneous fault detection and diagnosis framework (s-FDD framework). The premise is that with the integration of corrective and preventive maintenance strategies to s-FDD framework one can maximize the profit by minimizing the time spent under faulty operation.

Faults occur when a change/disturbance occurs in a system and system cannot reverse it by itself via existing controllers. Whereas failure is defined as the inability of the process or the equipment to perform its required functions within the specified performance requirements to counteract these disturbances. Failures may occur due to several different reasons. In general, they are classified under two categories: (i) process-based, and (ii) mechanical-based failures (Figure 5.1). Process-based failures occur when a disturbance changes the process dynamics of an operation, where the existing control scheme understanding the old process dynamics cannot handle it anymore. On the other hand, mechanical-based failures generally occur due to equipment age or wear, where the failure probability of equipment increases over time (i.e. batch cycles in batch/semi-batch processes). In order to tackle the two different classes of failures, one may need to adopt

different maintenance strategies (88).

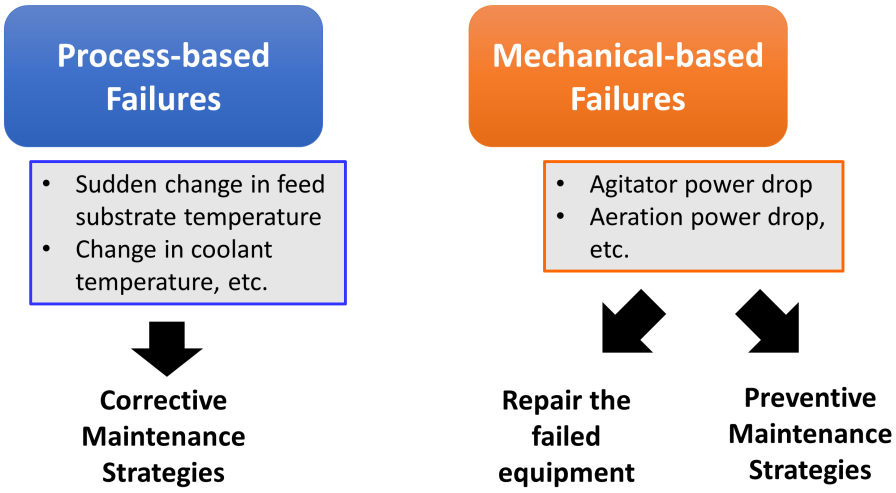


Figure 5.1: Different maintenance strategies for different failure types.

5.1 Corrective Maintenance Strategies

Corrective maintenance includes the actions toward failed equipment or highly disturbed system in order to restore the operation to its nominal condition. As discussed in Section 1.2 of Chapter 1, corrective maintenance strategies can be categorized into two groups based on the extremity of the process abnormality: (i) "run-to-failure" reactive category, where complete equipment failure occurs, and (ii) corrective category, where only individual/minor faults (high disturbances where controller cannot handle) occur but the system does not fail immediately. The first one is the most extreme case, where the only solution to recover the process is to repair or replace the failed equipment (i.e. renewal of a sensor, replacement of a malfunctioning equipment etc.). The source of such failures are mechanical. To handle these types of failures, we incorporate reactive maintenance strategies, where we repair the failed equipment within the s-FDD framework.

Moreover, another source of operation failures are process-based. This is the case where significant change in process conditions arise and controllers cannot continue to operate to keep the process in nominal condition. If this type of failure cannot be reversed on time, it may further affect

the units of the system and cause equipment failures. Therefore corrective maintenance strategies are required to be applied to recover the process. These strategies describe the actions that are taken toward the process-based failures which change the process dynamics and lead existing controllers inoperable. One of the common actions taken for performing corrective maintenance is re-tuning of the existing controllers within the system, while the process continues to operate under faulty condition. Here, the pitfall is the process time spent under faulty operation which prevents the process to achieve its goals, therefore can be referred as process downtime. Therefore, in order to prevent process downtime, our main goal is to (i) design and implement a multi-parametric Model Predictive Controller (mp-MPC) (181) via PAROC framework(182) to have an offline, *a priori*, map of optimal fault-aware actions to a chemical process under normal as well as faulty operation, (ii) integrate the designed mp-MPC with s-FDD framework, where s-FDD framework will provide process information to the mp-MPC, which in turn makes mp-MPC to switch between normal and faulty operation. This corrective maintenance strategy formulates a novel "parametric fault-tolerant control" concept which is described in detail in Section 5.3. In the first part of this chapter, we integrate reactive and preventive maintenance strategies to the s-FDD framework to assess the economic impact of each element on the profit.

5.2 Preventive Maintenance Strategies

Preventive maintenance strategies are the maintenance strategies that are scheduled within the process in order to delay and if possible completely prevent any possible failures occurring due to equipment (mechanical-based failures). The activities under this maintenance strategy include inspection and replacement of process equipment (87). Here, the idea is by performing regular inspections and preventive tasks, one may sustain the failure probability as low as possible depending on the age and usage of the equipment.

In this section, the goal is to implement maintenance strategies into chemical processes while monitoring them via s-FDD framework for early detection and diagnosis of the faults. The hypothesis is that integrated s-FDD framework with corrective and preventive maintenance tasks would perform superior than using either the s-FDD framework or maintenance tasks alone. To test this hypothesis, penicillin benchmark process, a semi-batch process, is used. Particular to batch processes, the premise is that (i) in the intra-batch time scale, s-FDD will detect and diagnose faults

early, and (ii) in the inter-batch time scale, preventive maintenance scheduling will delay fault occurrence within the process. The ultimate goal is the maximization of profit by minimizing process downtime due to mechanical-based failures.

Testing of the proposed hypothesis is done by considering a previously introduced faulty batch process data (Section 4.1). Fault 3 - agitator power drop (Table 4.2), which occurs due to mechanical-based failure, is adopted, and 4 different scenarios are considered as shown in Table 5.1. Failure rates are assumed to increase linearly and based on Weibull distribution (87). In order to show a proof of the proposed concept, the following question is postulated: In how many batch cycles, we can obtain 100 successful batches? Here, batches are running sequentially, therefore the optimal preventive maintenance frequency is calculated by minimizing the expected cost as shown in Equation 5.1 and 5.2 for linear and Weibull distribution based failure rate increase. The parameter/variable definitions used in Equations 5.1 and 5.2 are given in Table 5.2, and adopted parameter values are tabulated in Table 5.3, respectively.

Table 5.1: 4 considered scenarios for the proof of concept

Scenario	Use of s-FDD	Scheduling Preventive Maintenance	Repairing Failed Equipment
1	NO	NO	YES
2	YES	NO	YES
3	NO	YES	YES
4	YES	YES	YES

$$\begin{aligned} \min_{x} \quad & \frac{\sum_{i=1}^{x} C_{op}(1-f_i) + C_m - R(1-f_i)}{x} \\ \text{s.t.} \quad & f_i = f_{i-1} + f_{base} \\ & f_i \leq 0.5 \end{aligned} \tag{5.1}$$

$$\begin{aligned} \min_{x} \quad & \frac{\sum_{i=1}^{x} C_{op}(1-f_i) + C_m - R(1-f_i)}{x} \\ \text{s.t.} \quad & f_i = 1 - e^{-(\frac{i}{b})^a} \\ & f_i \leq 0.5 \end{aligned} \tag{5.2}$$

Table 5.2: Parameter and variable definitions used in Equation 5.1

Parameter/Variable	Definition
f_i	Failure Rate of the Equipment during Batch Cycle i
C_m	Preventive Maintenance Cost
C_{op}	Operating Cost per Batch
R	Revenue from each Successful Batch
x	Optimal Preventive Maintenance Pattern
f_{base}	Base Failure Rate
a	Shape parameter of Weibull distribution
b	Scale parameter of Weibull distribution

Table 5.3: Adopted parameters from *Peters et. al* (6)

Parameter	Value
Production Scale	30 tonnes/batch
Revenue	12,000,000 $/year
Operating Cost	1,000,000 $/year
Repair Cost	1,000,000 $/year
Preventive Maintenance Cost	500,000 $/year
Target Batch Number	100
Base Failure Rate (f_i)	0.05
a	3
b	120
Fault Detection & Diagnosis Policy	2 Sequential Alarm Raise

By solving the Equation 5.1, the optimal preventive maintenance frequency is obtained as 2. Whereas by solving the Equation 5.1, the optimal preventive maintenance frequency is obtained as 6.This means that, at every 6 batch cycles, a preventive maintenance task is performed for optimal operation under given conditions where failure rate increase follows a Weibull distribution where k is 3 and α is 120 (Figure 5.2, where x is 6 in this case). This information is used for the 4 different considered scenarios as shown in Figure 5.3.

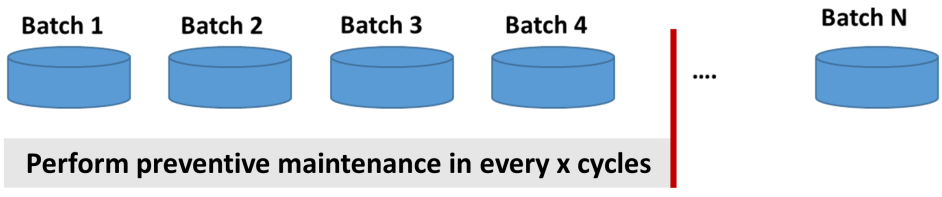


Figure 5.2: Scheduling scheme of preventive maintenance.

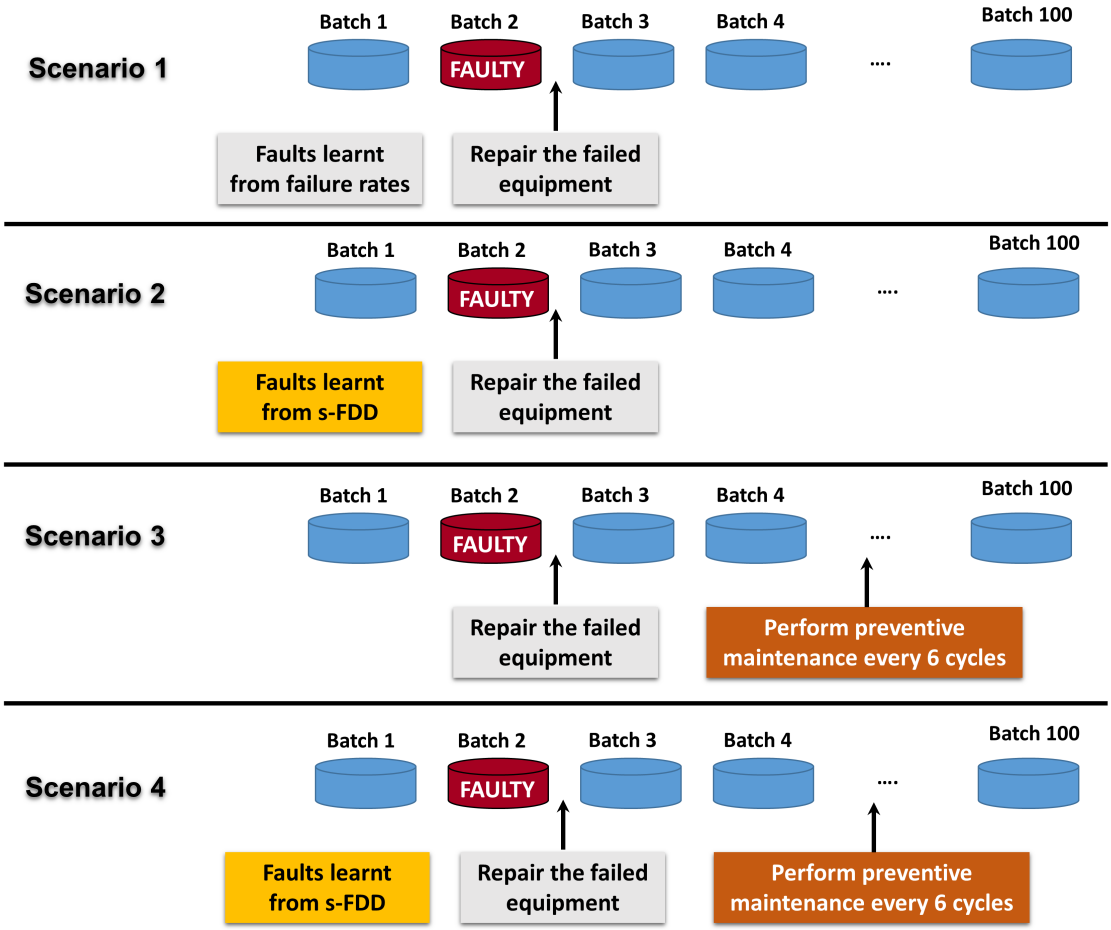


Figure 5.3: Understanding the effect of maintenance task integration to the s-FDD Framework.

The obtained results have validated the proposed hypothesis. The results that are obtained with Weibull distributed failure rates show that the process downtime is minimized by 9.17% between the two extreme cases: (i) the one where s-FDD is not used and preventive maintenance tasks are not scheduled, and (ii) the one wherein integrated s-FDD framework and preventive maintenance scheduling approach is adopted 5.4. This number increases to 43.23% when the failure rate is assumed to increase linearly. The profit increase between these two extreme cases is observed as 120.68% and 9.05% for failure rates that are assumed to increase linearly and based on the specified Weibull distribution, respectively. If one compares the scenarios 1 and 2, where preventive maintenance tasks are not scheduled in both scenarios but s-FDD is used in the latter

one, a 50.02% (linear failure rate model) and 6% (Weibull distributed failure rate model) profit increase are still observed with the use of s-FDD framework. Finally, if scenarios 3 and 4 are compared, where preventive maintenance tasks are scheduled in both scenarios but s-FDD is used only in scenario 4, a 2.5% (linear failure rate model) and 1.99% (Weibull distributed failure rate model) profit increase are obtained.

Table 5.4: Results for the proof of concept - Failure Rate from Weibull Distribution

Scenario	Use of s-FDD	Scheduling Preventive Maintenance	Repairing Failed Equipment	Cycles	Profit($)
1	NO	NO	YES	192	700,000,000
2	YES	NO	YES	192	741,966,101
3	NO	YES	YES	109	748,500,000
4	YES	YES	YES	109	763,381,355

Table 5.5: Results for the proof of concept - Linear Increase in Failure Rate

Scenario	Use of s-FDD	Scheduling Preventive Maintenance	Repairing Failed Equipment	Cycles	Profit($)
1	NO	NO	YES	192	340,000,000
2	YES	NO	YES	192	510,101,694
3	NO	YES	YES	109	732,000,000
4	YES	YES	YES	109	750,305,080

5.3 Integrating Multi-Parametric Model Predictive Control with s-FDD Framework: Parametric Fault-Tolerant Control Systems Design

In the second part of this chapter, we integrate multi-parametric model predictive control and the s-FDD framework to introduce a novel corrective maintenance strategy, parametric fault-tolerant control (FTC) systems design. By using multi-parametric programming, we are able to establish the control actions for the faulty state explicitly and generate *a priori*, offline, map to be implemented in the online phase. This is an active fault-tolerant control strategy, where we need to use online fault detection and identification (FDI) mechanism to monitor the process and get information on faults for further fault accommodation. In order to have a precise representation of the process condition for detecting any abnormalities, we need to have accurate and robust (i) fault detection and diagnosis scheme, and (ii) fault direction and magnitude estimation scheme, known as fault reconstruction mechanism. These two schemes collect required information on the process condition and pass them to the fault-tolerant mp-MPC in order to draw the necessary corrective actions to return the process to the nominal stage. Therefore, the accuracy of these two schemes play a significant role in the reliability of the proposed active FTC strategy.

Below, we present the parametric fault-tolerant control framework as a novel corrective maintenance strategy. The framework consists of three main steps: (i) offline design of fault-tolerant multi-parametric model predictive controller (mp-MPC) by using PAROC framework (182), (ii) offline design of an accurate fault detection and reconstruction mechanism by using the s-FDD framework, and (iii) implementation for online monitoring.

5.4 Parametric Fault-Tolerant Control Framework

The first step in building parametric fault-tolerant mp-MPC system is data acquisition. This can be achieved via either historical operation data or process data simulations based on the dynamic model of the system, which is often readily available in industrial applications. For the offline design of the fault-tolerant mp-MPC, we only need normal operation data. We collect both normal and faulty operation data for building fault detection and diagnosis models by using two-class classification models developed by following the s-FDD framework. Similarly, we use both normal and faulty operation data during fault magnitude regressor development.

In this work, we adopt fed-batch penicillin production process introduced in Chapter 4. We

simulate process data for fed-batch penicillin production by using the RAYMOND simulation package (183). We produce 25 simulations for each fault magnitude and onset combinations in addition to the 200 simulations of normal operating condition (NOC) by using the RAYMOND software. Of note, fault direction is defined as $measuredvalue - realvalue$ within the RAYMOND simulator. In this work, a nominal feed rate of 0.06 L/h is chosen for the fed-batch phase of the simulations. A batch is terminated after a total of 30 L of substrate have been added. This corresponds to a total batch duration of approximately 549 h. The initial fermenter volume V_0, biomass concentration $C_{x,0}$, and substrate concentration $C_{s,0}$ are all independently sampled from normal distributions with mean μ and standard deviation σ. Values are limited to $\mu \pm 2.5\sigma$ in order to avoid outliers in the initial conditions. Measurements are collected from 20 process variables, where white noise is included into each of them 5.6. Sensors are sampled every 0.2 h which has generated an average of 2745 sample points per batch.

In particular, we control the reactor temperature via fault-tolerant mp-MPC by manipulating water flow rate. We have studied two distinct fault types: (i) sensor fault, which introduces a bias in reactor temperature measurements, and (ii) actuator fault, which creates bias in water flow rate.

Table 5.6: List of process variables

Variable Name		Initial condition	Measurement Noise (σ)	Type
1.	Substrate concentration [mg/L]	17.5 ± 1	0.01	State variable
2.	Dissolved O_2 concentration [mg/L]	1.1601	0.004	State variable
3.	Biomass concentration [g/L]	0.1250 ± 0.030	0.5	State variable
4.	Pencillin concentration [g/L]	0	0.02	State variable
5.	Fermentation volume [m^3]	102.5 ± 5	0.002	State variable
6.	Dissolved CO_2 concentration [mg/L]	0.4487	0.12	State variable
7.	pH [-]	5	0.02	State variable
8.	Reactor temperature [K]	298	0.01	State variable (controlled)
9.	Reaction heat [cal]	0	–	State variable
10.	Feed rate [L/h]	–	0.01	Input variable
11.	Aeration rate [L/h]	–	0.01	Input variable
12.	Agitator power [W]	–	0.01	Input variable
13.	Feed temperature [K]	–	0.1	Input variable
14.	Water flow rate [L/h]	–	0.01	Input variable (manipulated)
15.	Hot/cold switch [–]	–	–	Input variable
16.	Base flow rate [mL/h]	–	0.01	Input variable
17.	Acid flow rate [mL/h]	–	0.01	Input variable
18.	Feed substrate concentration [mg/L]	–	–	Input variable
19.	Cooling medium temperature [K]	–	0.1	Input variable
20.	Heating medium temperature [K]	–	0.05	Input variable

5.4.1 Offline Fault-Tolerant mp-MPC Design via PAROC Framework

In this work, we build fault-tolerant mp-MPC by using the PARametric Optimization and Control (PAROC) framework (182) (Figure 5.4), which provides a systematic methodology to design advanced model-based controllers via multiparametric programming. The PAROC framework presents an extensive environment for the design of chemical processes, building controllers, and performing parameter estimation based on high-fidelity models while benefiting from the most recent advances in the field of multiparametric programming. Numerous applications of the PAROC framework are demonstrated in the literature for the integration of (i) process design and control (184; 185; 186), (ii) process scheduling and control (187), and (iii) process design, control, and

scheduling (188).

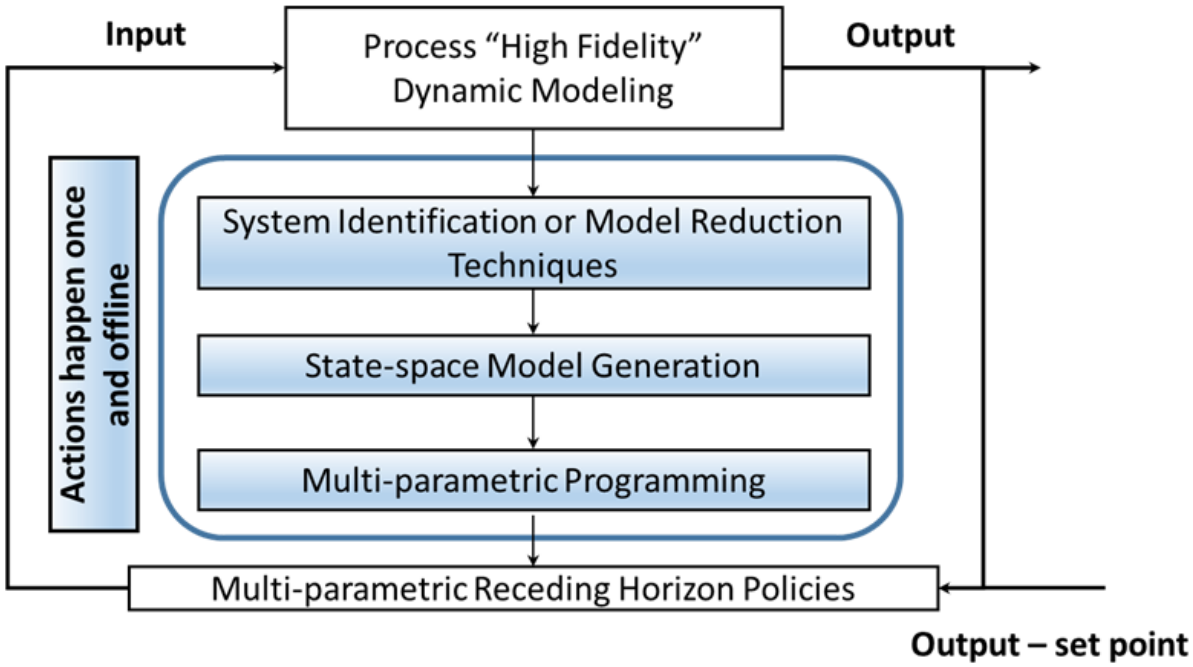


Figure 5.4: The PAROC framework.

The initial step of the PAROC framework is high fidelity modeling and analysis in order to acquire a mathematical model that can describe the system of interest accurately. However, often times the derived mathematical models are high-dimensional with large number of variables and/or complex in nature posing a significant challenge during their optimization in terms of computational expense. This further hinders their direct use of these models for the development of model-based strategies, and necessitates model approximation or reduction steps prior to the controller design. The reduced model is then used to build a model predictive control (MPC) scheme, and solved via multi-parametric programming to obtain mp-MPC, which produces the offline map of optimal control actions under both normal and faulty operations. Here, the fault-tolerant control scheme is achieved by introducing the mismatch (fault) information as an additional dimension during the mp-MPC design. Final step is "closed-loop validation", where we implement the ex-

tracted offline map of fault-tolerant control actions to the original mathematical model of the process to observe the closed-loop behavior of the system. The PAROC framework has been applied to numerous fields sucessfully (188; 187; 186; 185; 184).The details of each step are provided below.

5.4.1.1 High Fidelity Dynamic Modeling

A detailed and accurate representation of the system dynamics based on first principle dynamics and empirical correlations is used to simulate the open loop characteristics of the fed-batch penicillin production process. In this work, we employ the differential algebraic model (DAE) model presented by Birol et al. (2002) (179), as generalized below.

$$\dot{x} = f(x,u) \tag{5.3}$$

where x are the states of the system, u are the manipulated variables given in Table 5.6, and f is a generic function.

5.4.1.2 Model Approximation

The detailed model represented by Eq. 5.3 features complex and highly nonlinear dynamics among the manipulated variables and the observed outputs, rendering it inappropriate to develop advanced parametric controllers. Therefore, we develop an affine approximate model that accurately represents the high fidelity dynamics by model reduction or subspace identification techniques. In this work, we use MATLAB System Identification Toolbox to capture the dynamics of Eq. 5.3 by the discrete time state space model, given by the equation below.

$$\begin{aligned} x_{t+1} &= Ax_t + Bu_t + Cd_t \\ \hat{y}_t &= Dx_t + Eu_t + Fd_t \end{aligned} \tag{5.4}$$

where subscript t denotes the discretized time step, and $\hat{y}$ is the output prediction. The state space matrices are developed based on the simulated process outputs y under randomized input profiles for u and d. Note that the identified states x do not represent the real system states.

Acquiring satisfactory closed-loop performance relies heavily on developing accurate approximate models. Katz et al. (2018) (189) investigated the effects of approximating the high fidelity

models by simpler models in the context of multiparametric programming, and introduced novel error metrics to evaluate open and closed-loop performances. In this work, we use the open and closed loop metrics introduced in Katz et. al to increase the confidence of the developed approximate models. The details are omitted here for brevity and to focus on the fault-tolerant explicit control scheme.

5.4.1.3 Designing the MP-MPC

The offline control strategy is designed to (i) track the output set points determined prior to the operation, (ii) acquire smooth control actions to maintain the longevity of the processing equipment. Therefore, the objective function of the control problem is formulated is given by the following equation.

$$\sum_{t=1}^{N} \|y_t - y_t^{sp}\|_{QR}^2 + \sum_{t=0}^{M} \|\Delta u_t - \theta^a\|_{R1}^2 \tag{5.5}$$

where N is the prediction horizon, M is the control horizon, θ^a is the magnitude of the fault acting on the corresponding actuator, $\|\cdot\|_{\psi}$ denotes weighted vector norm with a weight matrix ψ, QR and $R1$ are the corresponding weight matrices, and the superscript sp denotes the set point. Hence, the quadratic objective function is minimized only if the process outputs track the designated set points y^{sp}, and the consecutive control actions are smooth in the existence of faulty actuators θ^a.

The developed objective function is subjected to the approximated process model, given by Eq. 5.4. However, using an approximate model to achieve closed-loop control creates a mismatch between the real process outputs, y, and the predicted output values, $\hat{y}$. We address this mismatch by including Eq. 5.6 in the mp-MPC formulation.

$$e = y_t - \hat{y}_t, t = 0 \tag{5.6}$$

where the error term e denotes the mismatch magnitude between the real and predicted output values at the time of measurement, $t = 0$. The error term is carried over the entire prediction horizon, as given by the equation below.

$$y_t = \hat{y}_t + e - \theta^s, \forall t \in \{1, 2, ..., N\} \tag{5.7}$$

Note that apart from the mismatch term, we also incorporate a sensor bias term θ^s to account for the sensor faults in the mp-MPC. The path constraints are formulated as box constraints for the process variables to maintain certain product specifications, as presented by the equation shown below.

$$\begin{aligned} \underline{x} \leq x_t \leq \overline{x} \\ \underline{y} \leq y_t \leq \overline{y} \\ \underline{u} \leq u_t \leq \overline{u} \\ \underline{\Delta u} \leq \Delta u_t - \theta^a \leq \overline{\Delta u} \end{aligned} \tag{5.8}$$

Lastly, we define the set of parameters in the control problem as following:

$$\theta := [x_{t=0}^T, u_{t=-1}^T, y_{t=0}^T, (y_t^{sp})^T, d_{t=0}^T, \theta^a, \theta^s]^T \tag{5.9}$$

where θ is the vector of parameters. Therefore, we postulate the explicit control strategy described by Eq. 5.10.

$$\begin{aligned} u_t(\theta) = \arg\min \quad & \text{Equation 5.5} \\ s.t. \quad & \text{Equations 5.4, 5.6-5.9} \end{aligned} \tag{5.10}$$

Note that the control strategy formulated by Eq. 5.10 is a multiparametric optimization problem with a quadratic objective function and a set of linear constraints. This class of problems can be solved exactly by using the Parametric OPtimization (POP) toolbox (190), and the solution to these problems are expressed as a single piece-wise affine function of the parameters. Therefore, the explicit control law is derived as given by the equation below.

$$\begin{aligned} u_t(\theta) = K_i\theta + r_i, \forall t \in \{1,2,...,M\}, \forall \theta \in CR_i \\ CR_i := \{\theta \in \Theta \mid CR_i^A\theta \leq CR_i^b\} \end{aligned} \tag{5.11}$$

where CR denotes a polyhedral partition of the feasible parameter space, and Θ is a closed and bounded set.

Remark 1. *Equation 5.11 explicitly maps the exact optimal control actions for any parameter realization in Θ, if a feasible solution exists. Therefore, inclusion of the monitored faults as pa-*

rameters in the explicit control law identifies the range of recovery in the existence of faulty sensors and/or actuators prior to the operation.

5.4.1.4 Closed-loop Validation

The proposed control problem is developed based on an approximate model, described in Step 2. Therefore, the closed-loop strategy should be validated against the high-fidelity model by observing the set point tracking performance and path constraint violation by exhaustive simulations under numerous uncertainty scenarios. Note that due to the explicit nature of the closed-loop strategy, the control law can be embedded in the high-fidelity model exactly. Therefore, the closed-loop profiles can be simulated without the necessity of solving any online optimization problems.

In the case of insufficient or poor closed-loop performance, one can (i) adjust the weight matrices QR and $R1$ in the objective function given by Eq. 5.5, (ii) develop a new approximate model using a different technique, or (iii) develop multiple discrete time state space models that are used to govern different operating regions.

5.4.2 Offline Fault Detection and Reconstruction Mechanism Development

The fault detection and reconstruction mechanism is responsible from two main tasks: (i) precise and rapid fault detection and diagnosis, and (ii) accurate fault direction and magnitude estimation (a.k.a fault reconstruction). We follow the main steps of the s-FDD framework ((Section 2.2 of Chapter 2)) to build fault and time-specific classification models for fault detection and diagnosis. Additionally, in order to predict the magnitude of the detected fault, we develop regression models by adopting Random Forest algorithm (191). Specifically, we regress the water flow rate measurements for the actuator fault, and reactor temperature measurements for the sensor fault. The modeling procedure for both analysis is summarized in 3 main steps. The initial step is data pre-processing which includes targeted data collection, unfolding of 3-dimensional (3D) batch process data into 2D, extracting additional process descriptors when necessary, and data scaling, respectively. This is followed by parameter tuning, and model building steps.

5.4.2.1 Data Pre-processing

Data pre-processing steps are necessary prior to model building in order to prevent bias and improve the performance of the model. Generally, data pre-processing comprises of data formatting, scaling and cleaning steps, where data cleaning includes both outlier removal and missing

data handling. Below, we describe these three main pillars of data pre-processing in four steps. Step-1.1 to 1.3 describe data formatting, where we collect targeted process data, unfold the 3D data into 2D, and extract further features when necessary to enrich the data set. We address data scaling and cleaning steps in Step-1.4.

Targeted Data Collection: We are building (i) fault and time specific two-class *C*-SVM classification models for fault detection and diagnosis, and (ii) regression models for fault magnitude estimation after fault onset time. Therefore we need to gather process data around the fault onset time for both modeling. In this work, we have selected four different fault onset times, 100, 200, 300, and 400 h, where we introduce two different faults in various magnitudes. The details on the fault types and their magnitude are provided in Section 5.5. In each batch, we consider the time periods that encompass the fault onset time and 10 h (50 sensor samples) afterwards.

During fault detection classifier building, we extract process data by following a sliding window approach. At each sensor sample, we collect historical data in 10 h blocks. For instance, to build a classifier around 100 h, we consider the time period of $100-110$ h of a batch. Next, starting from the fault onset time 100 h until 110^{th} h, we obtain process data in 10 h blocks: At hour 100, we collect data from $90-100^{th}$ h. Similarly at next sensor sample time, 100.2 h, we collect data from $90.2-100.2$ h. We obtain the process data iteratively until the end of the considered time period, 110 h. The schematic representation of the targeted data collection is presented in Figure 5.5, wherein the gray boxes mark the fault onset time classification models are built. Blue line indicates the first, red line indicates the last 10 h data block extracted from the 90-110 h time period for 100 h fault detection classifier.. Each data collection from the selected window adds a new instance in the data set. This approach yields a 3-dimensional (3D) data with size of 2500 X 20 X 50. The first dimension of the data set is obtained with 50 sliding window iterations in 50 batches (25 faulty and 25 normal operating). Furthermore, we observe 20 process variables that includes both state and manipulated variables 5.6 in 50 sensor sample periods (i.e. $100-110$ h for 100 h classifier building). The data set size is consistent for each fault and time-specific fault detection model building.

On the other hand, during fault magnitude regression development, we consider solely the process variables and do not extract any further process descriptors. Here, we collect 10 h block for actuator fault, and 1 h for sensor fault magnitude estimation model development. We also

combine all faulty operation data with varying fault magnitudes. Specifically, we have simulated 6 distinct fault magnitudes for sensor, and 8 for actuator faults. For each magnitude, we have simulated 25 batches. This yields 150 faulty batches with sensor fault, and 200 faulty batches with actuator fault. Next, we include equal amount of normal operating batches to our data sets. Thus, the size of the obtained data set is 300 X 20 X 5 for regression model development for sensor fault magnitude estimation. Whereas the data set size for regression model development for actuator fault magnitude estimation becomes 400 X 20 X 50. Here, the first dimension belongs to total number of batches (with equal number of faulty and normal operating batches), second dimension is the 20 process variable measurements, and the last dimension indicates the 10 h (50 sensor sample) block, and 1 h (5 sensor sample) block examined after the fault onset time of actuator, and sensor faults, respectively.

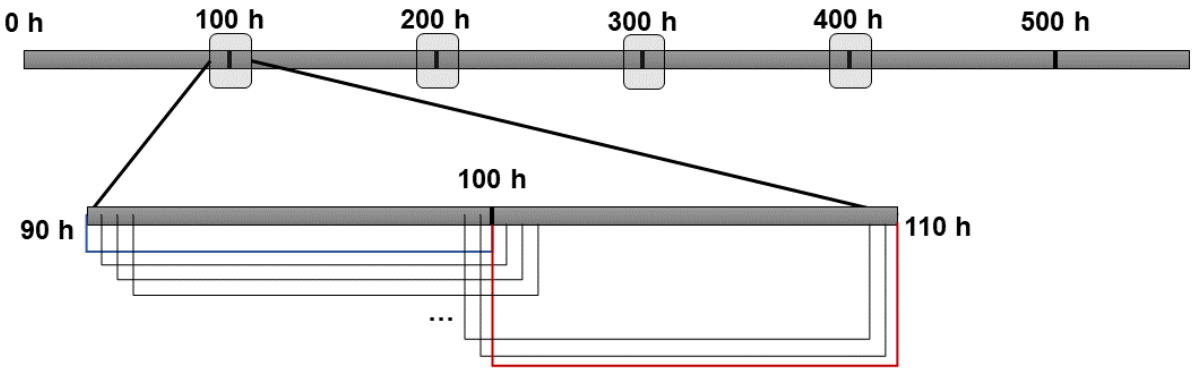


Figure 5.5: Schematic representation of targeted data collection for fault detection and diagnosis classifier development.

Unfolding 3D Batch Process Data into 2D: The collected 3D data needs to be unfolded into 2D prior to model building steps. The 3D data set can be unfolded in three ways by placing one out of three dimensions as rows, and grouping the other two dimensions as columns. In this work, we perform batch-wise unfolding for classification and measurement-wise unfolding for regression analysis (162). In bath-wise unfolding, batches are the instances which are provided in the rows of the 2D data set. Whereas in measurement-wise unfolding, we keep the process variable

measurements as the features and place them to the columns of the 2D data set for regression analysis. The features that constitute the columns of the 2D data sets are time-specific process variable measurements for classification, and time-specific-batch ID for regression models. After the unfolding step, the data set size becomes 2500 X 1000 for classification analysis. On the other hand, the unfolded data set size becomes 20000 X 20 for actuator fault, and 1500 X 20 for sensor fault magnitude estimation.

Extracting Additional Features: This step is optional. We apply this step only during classification analysis. The aim of this step is to enrich the data set by including additional process descriptors to capture the process nonlinearity, which can then improve classification model performances. To do this, we calculate the (i) mean, (ii) standard deviation, and (iii) slope of 20 process measurements within each sliding time window and incorporate them into the unfolded data set. This increases the classification data set sizes to 2500 X 1060.

Data Normalization and Reduction: Final data-preprocessing step is scaling of the re-configured data set and *a priori* dimensionality reduction to remove redundant features. This procedure is common to both classification and regression analysis. Each column of the 2D data set is scaled by z-score calculation, where mean of the column is extracted from each value and then divided into the standard deviation of the column. Redundant features, where the standard deviation is less than 10^{-8}, are removed in order to decrease the computational cost during offline model building phase.

5.4.2.2 Parameter Tuning

We are training (i) C-SVM (two-class) classification models by using the Gaussian radial basis function (RBF) as the nonlinear Kernel function for fault detection and diagnosis, and (ii) regression models via Random Forest algorithm for fault magnitude estimation after fault detection. Additionally, we have trained C-SVR models by using the introduced feature selection algorithm in Onel et al (192) and Onel et al (193) as described in Chapter 2. Table5.8 tabulates the results for actuator and Table 5.7 for sensor fault. Here, we prefer to keep the entire process variables instead of selecting a subset among them since the ratio of the number of features to the number of instances is significantly low. The results provided in Tables 5.8 and 5.7 validate this reasoning, where dimensionality reduction does not necessarily improve the model R^2s. Therefore, we use

the entire process variables that remain after the data pre-processing steps, and benefit the bagging technique of the Random Forest algorithm during model training to build accurate regressors.

Table 5.7: SVR model performances for the sensor fault.

Fault Onset Time	R^2	Number of Features
100 h	0.806	16
100 h	0.775	15
100 h	0.776	14
100 h	0.777	13
100 h	0.777	12
100 h	0.778	11
100 h	0.779	10
100 h	0.780	9
100 h	0.781	8
100 h	0.783	7
100 h	0.784	6
100 h	0.787	5
100 h	0.788	4
100 h	0.777	3
100 h	0.765	2
100 h	0.714	1
200 h	0.842	16
200 h	0.799	15
200 h	0.801	14
200 h	0.802	13
200 h	0.804	12
200 h	0.806	11
200 h	0.809	10
200 h	0.811	9
200 h	0.812	8

Table 5.7 – Continued

Fault Onset Time	R^2	Number of Features
200 h	0.814	7
200 h	0.816	6
200 h	0.817	5
200 h	0.818	4
200 h	0.818	3
200 h	0.815	2
200 h	0.792	1
300 h	0.850	16
300 h	0.846	15
300 h	0.847	14
300 h	0.848	13
300 h	0.849	12
300 h	0.850	11
300 h	0.852	10
300 h	0.853	9
300 h	0.854	8
300 h	0.854	7
300 h	0.790	6
300 h	0.793	5
300 h	0.797	4
300 h	0.798	3
300 h	0.795	2
300 h	0.795	1
400 h	0.852	16
400 h	0.852	15
400 h	0.853	14
400 h	0.854	13
400 h	0.855	12

Table 5.7 – Continued

Fault Onset Time	R^2	Number of Features
400 h	0.856	11
400 h	0.853	10
400 h	0.854	9
400 h	0.855	8
400 h	0.856	7
400 h	0.795	6
400 h	0.798	5
400 h	0.799	4
400 h	0.803	3
400 h	0.801	2
400 h	0.799	1

Table 5.8: SVR model performances for the actuator fault.

Fault Onset Time	R^2	Number of Features
100 h	0.999	1 to 16
200 h	0.999	4 to 16
200 h	0.998	1 to 3
300 h	0.999	2 to 16
300 h	0.998	1
400 h	0.999	3 to 16
400 h	0.998	1 to 2

Regardless of the analysis, parameter tuning is required to achieve the optimal model performance:

Parameter Tuning for C-SVM Models: Here, we have two parameters to tune: (i) C (cost) parameter of C-SVM, and (ii) γ parameter of the Gaussian Radial Basis kernel function. While the first

parameter acts as a regularization parameter that controls the trade-off between low training error and low test error. In other words, this parameter regulates the balance between model complexity and model generalization. Lower the training error, higher the model complexity and lower the model generalizability. On the other hand, lower the testing error, lower the model complexity and higher model generalizability but with higher training error. Finding an optimal balance is crucial to develop an accurate classifier. Furthermore, the γ parameter determines the complexity of the Gaussian RBF kernel and affects the radius of influence of the samples selected as support vectors by the model.

In LIBSVM, the default value for RBF kernel parameter, γ, is $1/n$, where n is the number of features. Thus, we tune parameter $\hat{\gamma}$ where

$$\gamma = \frac{2^{\hat{\gamma}}}{n}. \tag{5.12}$$

Moreover, we tune parameter $\hat{C}$, where the relation between $\hat{C}$ and C is:

$$C = 2^{\hat{C}}. \tag{5.13}$$

According to the described iterative feature selection algorithm in our previous papers (192; 193) and in Section 2.2 of Chapter 2, $\hat{\gamma}$ can be re-tuned after each feature elimination step with the available set of features:

$$\gamma = \frac{2^{\hat{\gamma}}}{z^{\mathsf{T}}\mathbf{1}} \tag{5.14}$$

We have performed the parameter tuning via grid search and 10-fold cross-validation. In particular, we have used the values between $-1:1$ for $\hat{C}$, and $-10:10$ for $\hat{\gamma}$. We have performed the parameter tuning once in the beginning where we have the entire features in the data set. Although repeating grid search for parameters tuning after each feature elimination would be ideal, we avoid the computational cost since the attained model performance has been observed to be satisfactory. If one obtains a poor-performing model, tuning can be repeated with each available feature subsets. Finally, the parameters that produce the highest average testing accuracy are chosen for the next steps. The optimal parameters for the fault-and-time specific C-SVM models are provided in Table 5.9.

Table 5.9: Optimal C and γ parameters of the C-SVM classifiers.

Fault Type	Fault Onset Time	Optimal $\hat{C}$	Optimal $\hat{\gamma}$
Actuator	100 h	1	0
Actuator	200 h	1	0
Actuator	300 h	1	0
Actuator	400 h	1	0
Sensor	100 h	1	-2
Sensor	200 h	-1	0
Sensor	300 h	1	-2
Sensor	400 h	1	-2

Parameter Tuning for Random Forest Regression Models: In regression analysis, we have one parameter to tune, which is the number of features that can be used in training of each decision trees of the random forest model, $mtry$. This is performed via grid search among the total number of features until 1 while training Random Forest models via 10-fold cross-validation. The optimal $mtry$ parameters are obtained by using the "trainControl" function of the "caret" package of the R statistical software. The optimal $mtry$ values for each time-specific regressors are provided in Table 5.10.

Table 5.10: Optimal *mtry* parameters of the Random Forest regressors.

Fault Type	Fault Onset Time	Optimal *mtry*
Actuator	100 h	12
Actuator	200 h	11
Actuator	300 h	11
Actuator	400 h	11
Sensor	100 h	16
Sensor	200 h	16
Sensor	300 h	12
Sensor	400 h	14

5.4.2.3 Model Building

Here, we address the model building steps separately for classification and regression analysis. We follow the s-FDD framework (192; 193) to build the *C*-SVM classifiers for fault detection and diagnosis. The application of the framework and data-specific details are provided in Step 3.1. Furthermore, we describe the model building steps for regression analysis via Random Forest algorithm in Step 3.2.

5.4.2.3.1 Training *C*-SVM Models

The overall procedure for fault detection model development is summarized in Figure 5.6.

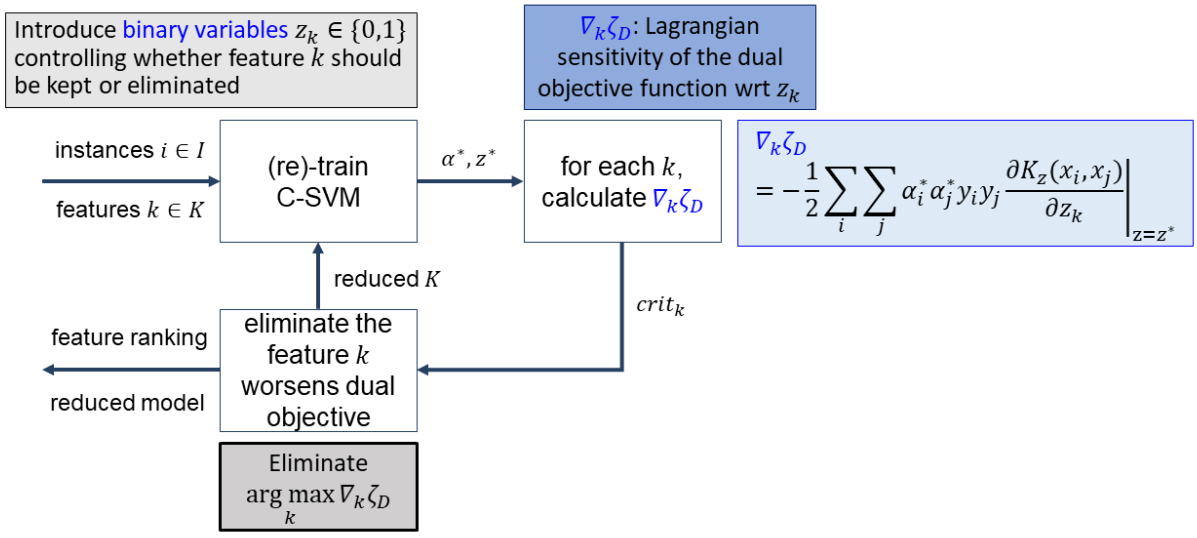


Figure 5.6: Algorithmic solution procedure for simultaneous Support Vector Machine-based feature selection and modeling.

Step-1. Feature Ranking via C-SVM Modeling:

The tuned parameters are incorporated into simultaneous model-informed feature selection and classification algorithm via *C*-SVMs (128; 192; 193). *C*-SVM binary classification models with Gaussian radial basis function (RBF) kernel are trained iteratively with each feature subset as features are eliminated one by one. Features are eliminated based on the Lagrangian sensitivity of the dual objective function of the built *C*-SVM model with respect to the feature subset size at each iteration. This iterative process is performed with each of the 10 train-test data set pairs which produces 10 separate feature ranking lists. Next, we create an average feature rank list based on the statistical distribution of the feature ranks among the 10 ranking lists.

Step-2. Developing C-SVM Models for each Feature Subset:

Here, we re-build *C*-SVM models by using the optimal parameters determined in Step 2 and 10-fold cross validation, where we use the average feature rank list to guide the iterative feature elimination process. We start with the whole set of features and eliminate them one by one based on this final ranking list. This process produces 10 *C*-SVM classifiers for each feature subset due to the 10-fold cross-validation. The performance of each model is assessed via accuracy, area under

the curve (AUC), fault detection rate, and false alarm rate. We average the performance of 10 classifiers and obtain one *C*-SVM model performance per feature subset. At the end of this step, we tabulate the performance of *C*-SVM models with each feature subset. Specifically, in this work, we have generated 1060 *C*-SVM models.

Step-3. Choosing the C-SVM Model with Optimal Feature Subset:

This step determines the final *C*-SVM models to be implemented in the online phase for process monitoring. Here, we select the classifier that has provided the highest model accuracy and area under the curve with minimum number of features among the 1060 *C*-SVM models produced in Step 4. The selected feature subset is used in analyzing the root-cause of the detected fault. Therefore, selecting minimum number of features is significant in order to facilitate the interpretation of the fault diagnosis. The performance of the selected fault-and-time specific *C*-SVM models are tabulated in Table 5.11.

Table 5.11: *C*-SVM model performances. (FDR: Fault Detection Rate, FAR: False Alarm Rate)

Fault Type	Fault Onset Time	Accuracy (%)	AUC	FDR	FAR (%)	Optimal Feature Subset Size
Actuator	100 h	98.29	99.84	97.35	0.77	30
Actuator	200 h	98.34	99.86	97.55	0.87	35
Actuator	300 h	98.37	99.83	97.52	0.77	32
Actuator	400 h	98.77	99.05	98.29	0.75	45
Sensor	100 h	94.92	97.34	89.85	0.00	33
Sensor	200 h	98.84	99.21	97.68	0.00	59
Sensor	300 h	98.03	99.39	96.06	0.00	45
Sensor	400 h	99.00	99.38	98.00	0.00	7

5.4.2.3.2 Training Random Forest Models

By using the optimal *mtry* parameters, we train Random Forest models with 500 decision trees. Training is performed via the "randomForest" function of the "randomForest" package of R statistical software. The performance of the fault-and-time specific Random Forest models are

tabulated in Table 5.12.

Table 5.12: Random Forest regressor performances (RMSE: Root Mean Square)

Fault Type	Fault Onset Time	R^2	RMSE
Actuator	100 h	0.999	0.179
Actuator	200 h	0.999	0.262
Actuator	300 h	0.999	0.248
Actuator	400 h	0.999	0.260
Sensor	100 h	0.964	0.202
Sensor	200 h	0.911	0.321
Sensor	300 h	0.985	0.129
Sensor	400 h	0.973	0.176

5.4.3 Closed-loop Validation and Online Implementation

Prior to the online implementation, we have implemented the developed fault-tolerant mp-MPC, and fault detection and reconstruction mechanism to the RAYMOND simulator separately in order to validate their individual performance. The performance of fault-tolerant mp-MPC is assessed by providing the fault onset time and magnitude information to the controller. We have observed that the controller adapts to the faulty condition once it is provided with accurate information on the fault type, onset time and magnitude. The accuracy of the fault detection and reconstruction mechanism is also tested and validated separately, where we have simulated a process with known fault onset and magnitude without incorporating any fault-tolerant control actions during the simulation. Finally, we integrate the fault-tolerant mp-MPC, and fault detection and reconstruction mechanism within the RAYMOND simulator simultaneously.

5.5 Results

In this work, we control reactor temperature by manipulating water flow rate during penicillin production. We build fault-tolerant control scheme that can tolerate for both actuator and sen-

sor fault. We introduce sensor bias in water flow rate measurements for actuator fault, whereas we introduce sensor bias in reactor temperature measurements to induce sensor fault. Numerous fault magnitudes and onset time are simulated for each fault type. Particularly, we select $-2.5, -2.0, -1.5, +1.5, +2.0, +2.5$, and $-2.0, -1.5, -1.0, -0.5, +0.5, +1.0, +1.5, +2.0$ for actuator and sensor fault magnitudes during the simulations, respectively. We have developed highly accurate fault and time-specific fault detection models and regression models for fault magnitude estimation (Table 5.11 and 5.12) and implemented them for the fault detection and reconstruction mechanism of the established parametric fault-tolerant control system.

Figure 5.7 provides a comparison of the open and closed (via fault-tolerant mp-MPC) loop simulation, which signifies the importance of having an accurate control actions on the reactor temperature by manipulating the water flow rate. The mp-MPC yields an offline, *a priori*, map of optimal control actions for the process. Figure 5.8 deliniates the distinct control laws for various magnitudes of sensor and actuator faults at the fixed parameters. The major advantage of the built fault-tolerant system is to gain *a priori* knowledge on the control actions for different fault magnitudes of actuator and sensor fault separately, as well as for different combinations of the two distinct fault types simultaneously. This map further draws the limits of the fault tolerance for each fault types. These limits indicate specific fault magnitudes for each fault type until where the designed fault-tolerant mp-MPC can recover the process back to the normal condition.

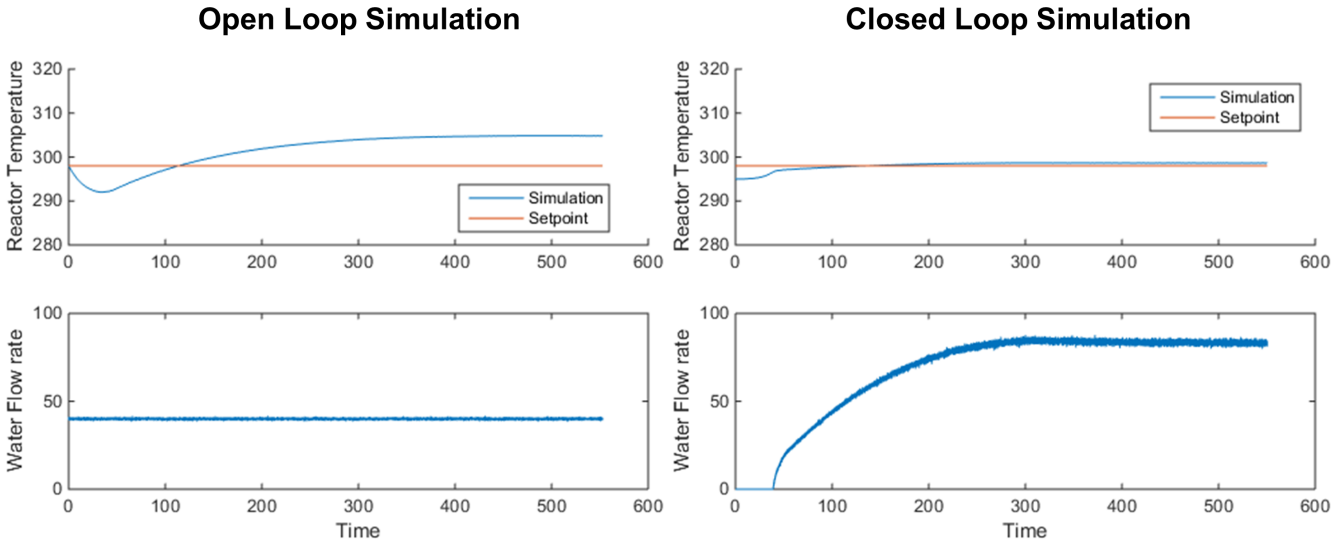


Figure 5.7: Simulated reactor temperature (controlled) and water flow rate (manipulated) profiles in open and closed loop (via mp-MPC).

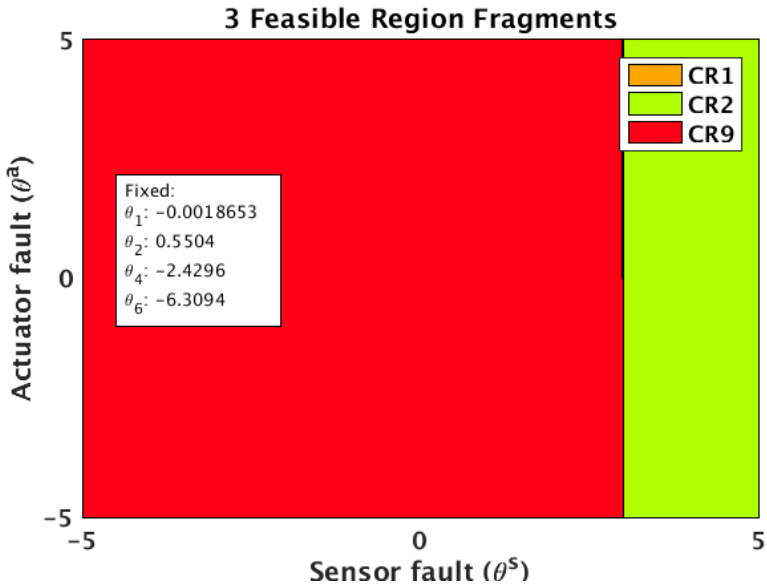


Figure 5.8: A demonstration of the offline map of the fault-tolerant mpMPC strategy projected to the actuator and sensor fault magnitudes at an arbitrary time in a closed-loop simulation. Each color contains a different explicit control law as a function of the parameters. The parameters θ_1 denotes the identified state, θ_2 is the normalized process output (reacture temperature), θ_4 is the output (reacture temperature) set point, and θ_6 denotes the previous control action.

From the beginning, we monitor the process with the fault tolerant mp-MPC and acquire information on the existence of any fault within the system from the fault detection classifier models. In this work, the adopted alarm policy is the generation of 3 consecutive alarms. In other words, we conclude on the fault existence when we obtain 3 consecutive positive response from the fault detection classifiers. Once the fault is detected, we initiate to regress the magnitude and direction of the fault. The random forest models use the online process variable measurements to estimate the amount of deviation from the reactor temperature of the normal operating condition. Here, early detection of the faults is crucial to initiate the fault estimation process. If the fault detection latency is high, that is when fault is detected late during the operation, the controller may not be able to return the process back to the normal condition. The reason can be two-fold: (i) the validity of the regressor may expire, thus accuracy of the fault estimation deteriorates, and (ii) there may be significant damage on the process which is irreparable. Table 5.13 presents the average fault detection latency of each fault and time specific classifier among the entire simulations with varying fault magnitudes.

Table 5.13: Average fault detection latency of the fault&time-specific *C*-SVM models.

Fault Type	Fault Onset Time	Average Latency (h)
Actuator	100 h	0.5
Actuator	200 h	1.5
Actuator	300 h	0.04
Actuator	400 h	0.16
Sensor	100 h	0.38
Sensor	200 h	1.27
Sensor	300 h	0.64
Sensor	400 h	6.17

Achieving low latency with the fault and time-specific *C*-SVM models indicates early fault detection. When we compare the two different fault types, the average latency is lower for the

actuator fault models. The main reason for this can be the fact that changes in water flow rate may affect the other process variables in a more definite way. This may lead to sudden changes not only in one but numerous process variables, thus facilitates the fault detection. Furthermore, the process nonlinearity affects the detection latency in distinct ways for different fault types. Specifically, we observe that we detect the actuator fault more rapidly in the later stages of the batch process, namely 300 h and 400 h models. On the other hand, the separation in the average latency is not that clear among the sensor fault detection models. Here, the high latency can be linked to the low number of process variables used in the fault magnitude estimator models, which may not be adequate to capture the process behavior in the specific process time.

We are building fault and time-specific regression models for fault magnitude estimation. Therefore, it is crucial to assess the accuracy of the fault reconstruction performance after the fault onset time. During the online operation, we use the regressors that are trained around the simulated fault onset time. As the operation progresses after the fault onset time, where the process is kept under normal condition thanks to the fault-tolerant mp-MPC, the regressor model continues to use the online process data at every new sampling point. However, as the sampling time moves away from the fault onset time, the process data characteristics can significantly change, which renders the regressor inaccurate for fault estimation. Fault estimation may not be performed as accurate as it is done near the fault onset time, which hinders the controller's learning about the process condition. This, in turn, may lead to insufficient control actions to recover the process back to the normal condition. Note that the extended validity of the regressor accuracy heavily depends on the amount of deviation of the process data characteristics. As a result, it is significant and necessary to assess the time-sensitivity of the fault estimators and identify when we need new models for accurate fault reconstruction. Furthermore, the limit of each regressor determines the targeted data collection location for the next regression model training.

Tables 5.14 and 5.15 tabulate the extent of the validity of the time-speicifc fault detection classifiers and magnitude estimation regressors for two set of thresholds around the reactor temperature set point being $\pm$ 0.5 and $\pm$ 0.75 K. The complete set of reactor temperature and water flow rate profiles with $\pm$ 0.5 K threshold on the set point for each time-specific models are provided in the Appendix B. In particular, we assess the extent of the validity of each time-specific model until the next time-specific model territory (i.e. 100^{th} h models are tested until 200^{th} h etc.). The re-

sults for the actuator fault case show that the models that are built at 200^{th} and 300^{th} h have been sucessfully provided necessary control actions until the target process time being 300^{th} and 400^{th} h respectively. Similarly, models built at 400^{th} h have enabled satisfactory control actions until the end of the operation. The results for the models built for 100^{th} show that the models are valid on average for the next 73.5 h and 75.4 h for $\pm$ 0.5 and 0.75 K thresholds around the reacture temperature set point, respectively. This highlights that we need to have additional models for accurate fault detection and magnitude estimation between $100^{th} - 200^{th}$ h of the batch operation.

On the other hand, for the sensor fault case, we note that the models built for 200^{th} and 400^{th} h are not valid for an extended process time when negative fault magnitudes are observed. On average, the models are valid for another 1.3 and 1.5 h after fault is introduced in 200 and 400 h, respectively when the reactor temperature deviation threshold is set to 0.5 K. When we increase the threshold to 0.75 K around the set point, we observe that the models built at 200^{th} can maintain a smooth operation for its entire targeted operation range, which is the next 100 h, because the latency in fault detection has caused a deviation that is higher than 0.5 but lower than 0.75 K. However this does not apply to the models for 400^{th} h. The threshold increase does not extend the validity of 400^{th} h models since the maximum deviation observed is as high as 2.1 K (Figure A49). The reason behind the limited model validity for the two time-specific models at 200 and 400 h is due to the high fault detection latency. In other words, by the time we detect the fault occuring at 200^{th} and 400^{th} h, the deviation from the reactor temperature set point already exceeds the predetermined thresholds (Figure A33 - Figure A36 for 200^{th} models and Figure A49 - Figure A52 for 400^{th} models). Ttherefore, required control actions are not provided by the controller as it has not been notified with the existence and magnitude of the fault. In order to overcome this problem, fault detection latency is required to be improved. This can be achieved by increasing the frequency of the fault detection classifiers between 200 and 400 h of the batch operation.

Table 5.14: Extent of time-specific fault detection and magnitude estimation model validity for actuator fault

Fault Onset Time (h)	Fault Magnitude	Validity Limit for 0.5 K threshold (h)	Validity Limit for 0.75 K threshold (h)
100	-2.5	143.2	146.6
100	-2	147.3	150.7
100	-1.5	151.2	155.0
100	1.5	199.4	200.0
100	2	200.0	200.0
100	2.5	200.0	200.0
200	-2.5	200.0	200.0
200	-2	300.0	300.0
200	-1.5	300.0	300.0
200	1.5	300.0	300.0
200	2	300.0	300.0
200	2.5	300.0	300.0
300	-2.5	400.0	400.0
300	-2	400.0	400.0
300	-1.5	400.0	400.0
300	1.5	400.0	400.0
300	2	400.0	400.0
300	2.5	400.0	400.0
400	-2.5	Through the end	Through the end
400	-2	Through the end	Through the end
400	-1.5	Through the end	Through the end
400	1.5	Through the end	Through the end
400	2	Through the end	Through the end
400	2.5	Through the end	Through the end

Table 5.15: Extent of time-specific fault detection and magnitude estimation model validity for sensor fault

Fault Onset Time (h)	Fault Magnitude	Validity Limit for 0.5 K threshold (h)	Validity Limit for 0.75 K threshold (h)
100	-2.0	187.9	200.0
100	-1.5	187.4	200.0
100	-1.0	187.2	200.0
100	-0.5	187.2	200.0
100	0.5	187.6	200.0
100	1.0	187.6	200.0
100	1.5	187.6	200.0
100	2.0	187.4	200.0
200	-2.0	201.3	300.0
200	-1.5	201.3	300.0
200	-1.0	201.3	300.0
200	-0.5	201.3	300.0
200	0.5	300.0	300.0
200	1.0	300.0	300.0
200	1.5	300.0	300.0
200	2.0	300.0	300.0
300	-2.0	358.1	358.1
300	-1.5	356.4	356.4
300	-1.0	359.9	359.9
300	-0.5	358.0	358.0
300	0.5	369.9	369.9
300	1.0	360.5	360.5
300	1.5	360.4	360.4
300	2.0	363.1	363.1
400	-2.0	401.4	401.4
400	-1.5	401.4	401.4
400	-1.0	401.4	401.4
400	-0.5	401.5	401.5
400	0.5	Through the end	Through the end
400	1.0	Through the end	Through the end
400	1.5	Through the end	Through the end
400	2.0	Through the end	Through the end

In order to provide a comparison between the two fault types, we provide the reactor temperature and water flow rate profiles for 100 h models. Particularly, we display the profiles of the simulations where we introduce actuator fault with -2.5 and $+2.5$ fault magnitude in Figures 5.9 and 5.10, respectively. Additionally, Figures 5.11 and 5.12 demonstrate the profiles of the simulations where we introduce sensor fault with -2 and $+2$ fault magnitude. The profiles with actuator fault show that once the regressor model validity expires with the altering dynamics of the batch process, the correction in the faulty water flow rate disrupts and deteriorates. This leads to a significant increase in the reactor temperature that leads to a possible system failure. On the other hand, the early capture of the sensor fault leads to rapid and necessary changes in the water flow rate which enables fast process recovery back to the normal condition. Of note, in order to ensure smooth control actions, one needs to switch to the next valid model once the validity of the previous model expires. This is necessary in order to capture dynamic process characteristics and detect any possible faults. The presented simulation profiles with actuator and sensor faults prove that the designed fault-tolerant mp-MPC provides smooth control actions succesfully.

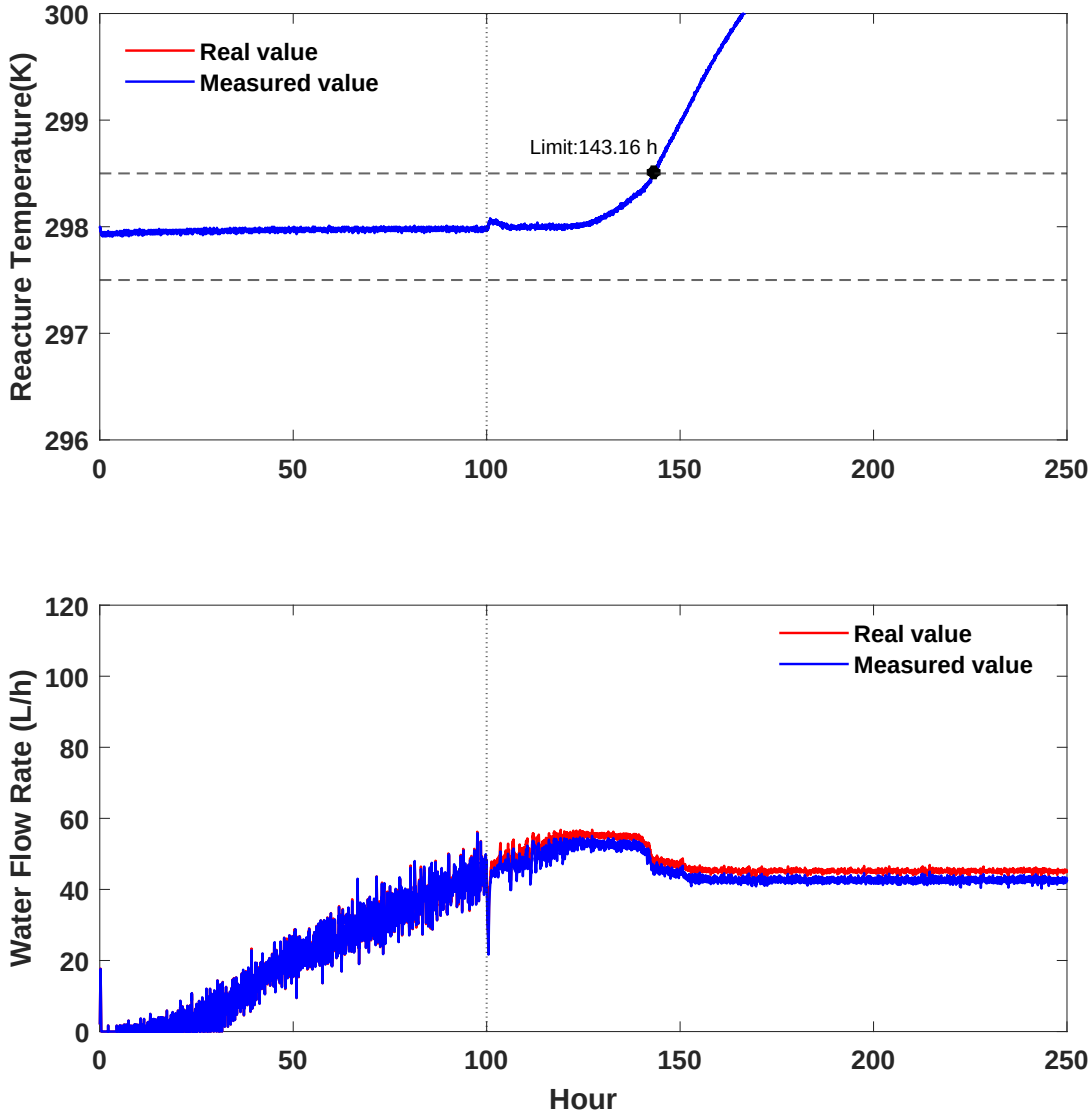


Figure 5.9: Reactor temperature and water flow rate profiles for process with actuator fault. Fault Onset: 100 h, Fault Magnitude: -2.5.

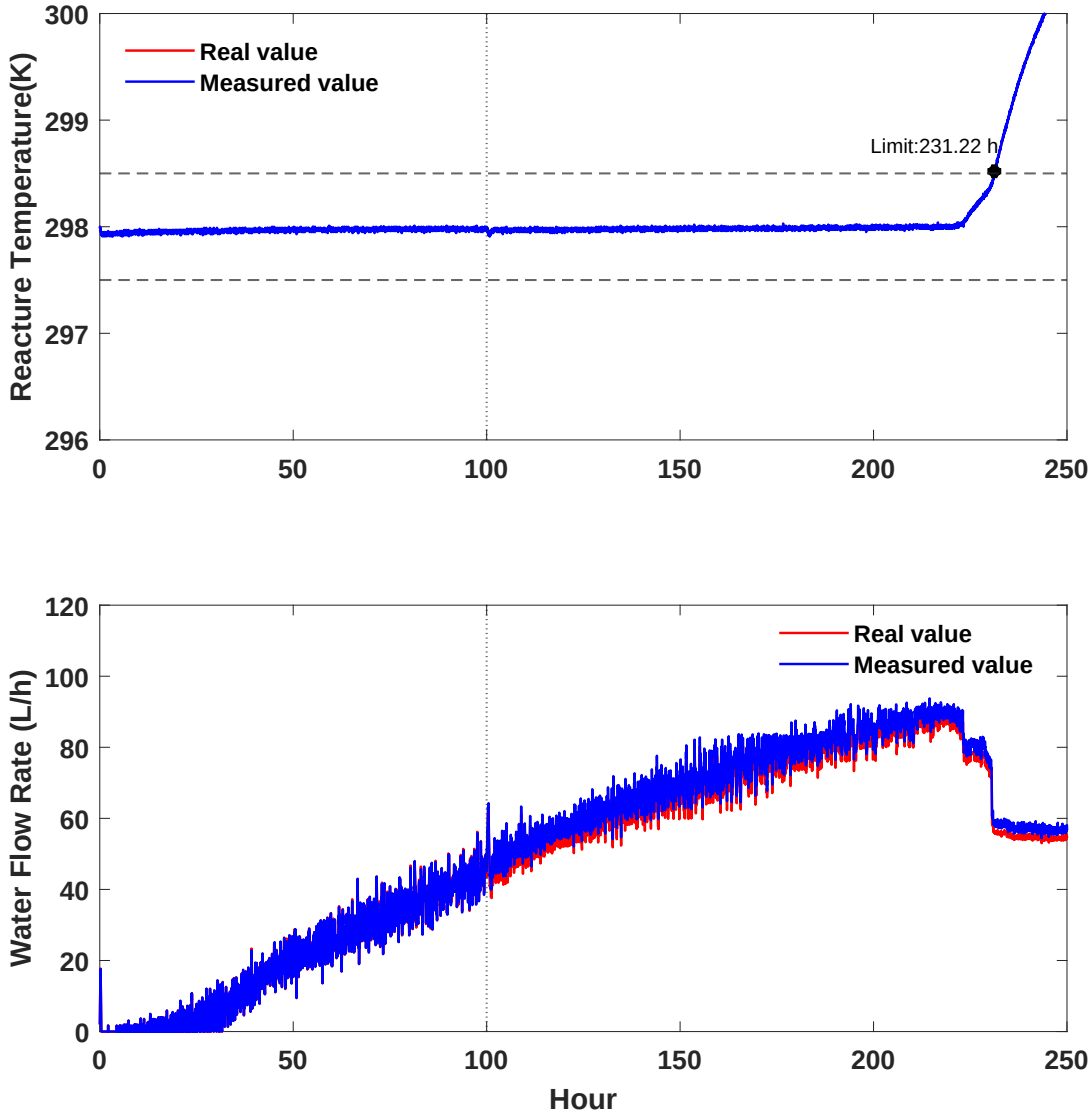


Figure 5.10: Reactor temperature and water flow rate profiles for process with actuator fault. Fault Onset: 100 h, Fault Magnitude: +2.5.

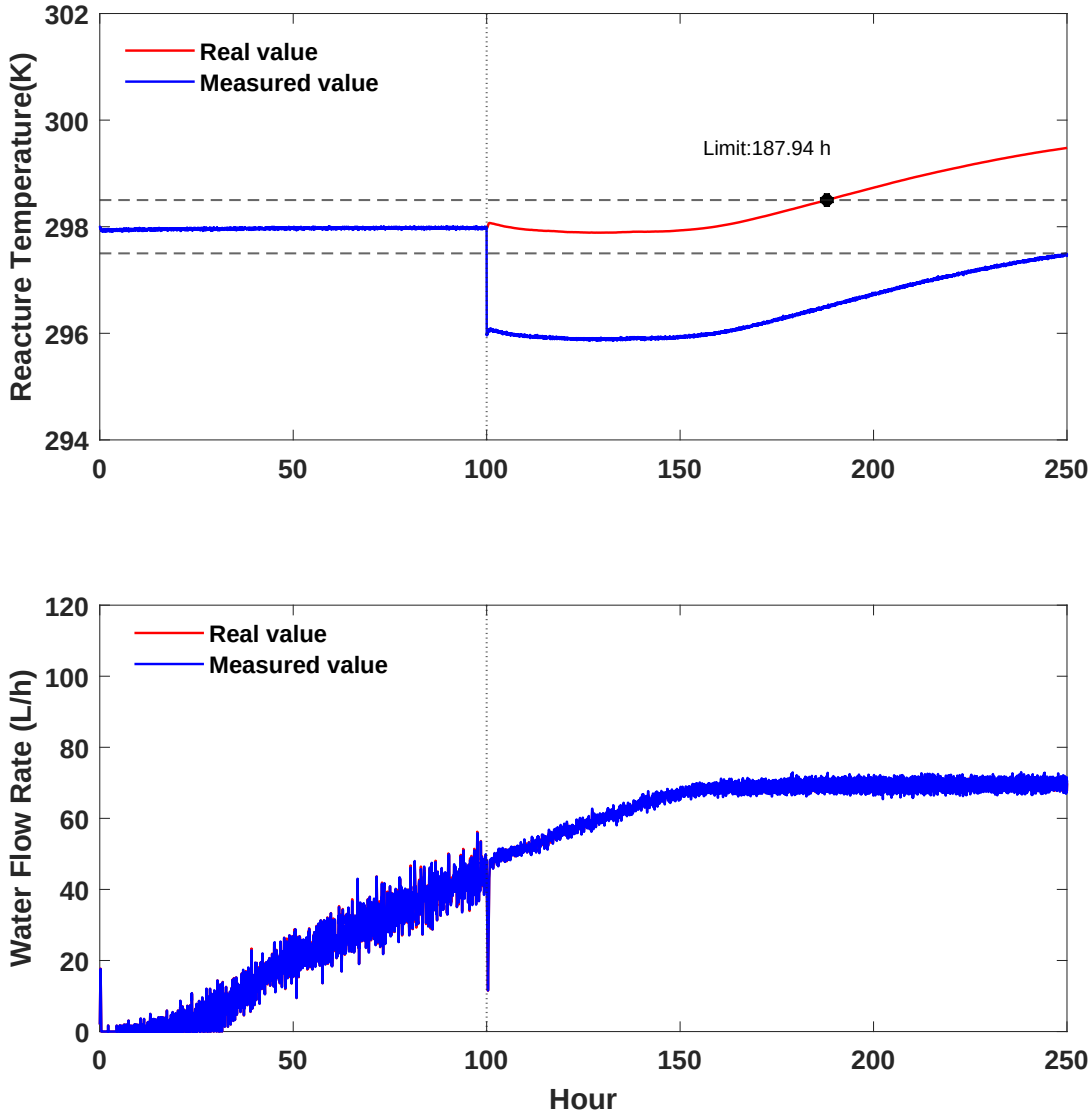


Figure 5.11: Reactor temperature and water flow rate profiles for process with sensor fault. Fault Onset: 100 h, Fault Magnitude: -2.0.

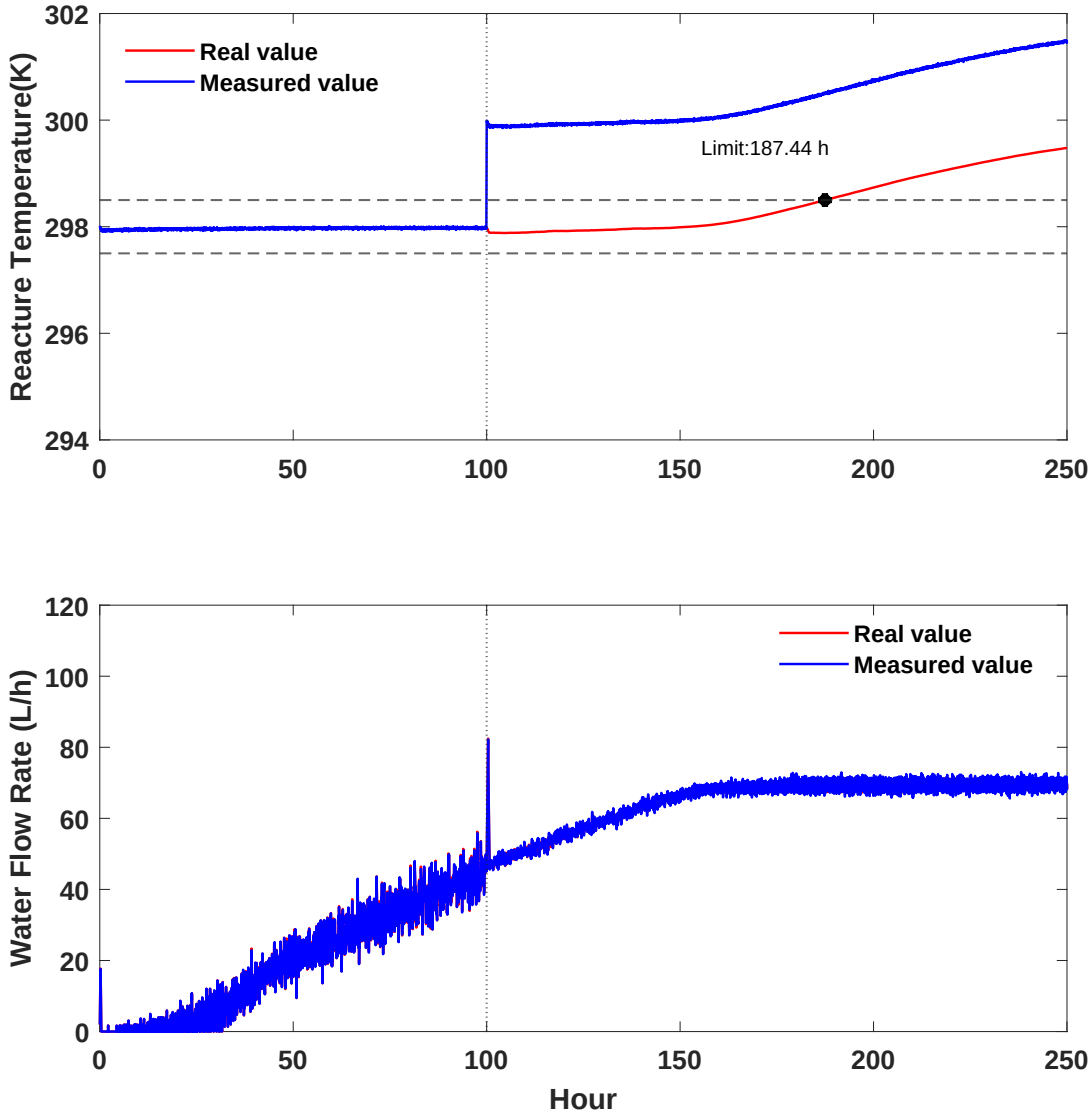


Figure 5.12: Reactor temperature and water flow rate profiles for process with sensor fault. Fault Onset: 100 h, Fault Magnitude: +2.0.

Finally, we perform a sensitivity analysis with the time-specific fault detection and magnitude estimation models built at 100^{th} and 200^{th} h in order to determine the perimeter of the model effectiveness. To this end, we use the time-specific models for $\pm$ 30 h perimeter of their corresponding process time. Particularly, the *C*-SVM model for fault detection and random forest model for the fault magnitude estimation are utilized for every 5 h fault onsets between 70^{th} and 130^{th} h with the models built at 100^{th} h and between 170^{th} and 230^{th} h with the models built at 200^{th} h (Figure

5.13). We adopt the $\pm$ 0.5 K threshold around the reactor temperature set point and only utilize the extreme negative and positive fault magnitudes simulated in this work (-2.5 and +2.5 for actuator and -2 and +2 for sensor fault) for the analysis. The results reveal that actuator fault models have more limited range compared to sensor fault models. In particular, the models built at 100^{th} have successfully been used between $90-100^{th}$ h of the batch operation. The validity range for the models built at 200^{th} reaches to 15 and 20 h for negative and positive fault magnitudes, respectively. On the other hand, the analysis yield that the models built at 100^{th} and 200^{th} h for sensor fault have been able to perform required control actions for the analyzed 30 h perimeter except the analysis performed with negative fault magnitude with models built at 200^{th} h. This is again due to the fact that by the time fault is detected the raise in the reactor temperature exceeds the allowed region (Figure A33). When the deviation threshold is raised to $\pm$ 0.75 K, the time-specific models are shown to be valid for the entire analyzed 30 h perimeter (Figure 5.14). This analysis is significant to elucidate the effectiveness limit of the time-specific models which is required to determine the model switching frequency during online monitoring. Overall, the results demonstrate the need for additional models during 100-200 h of the operation if a strict deviation threshold (i.e. 0.5 K) is preferred during the operation. Yet for 0.75 K devaition threshold, the presented time-specific models have successfully provided satisfactory control actions under faulty condition.

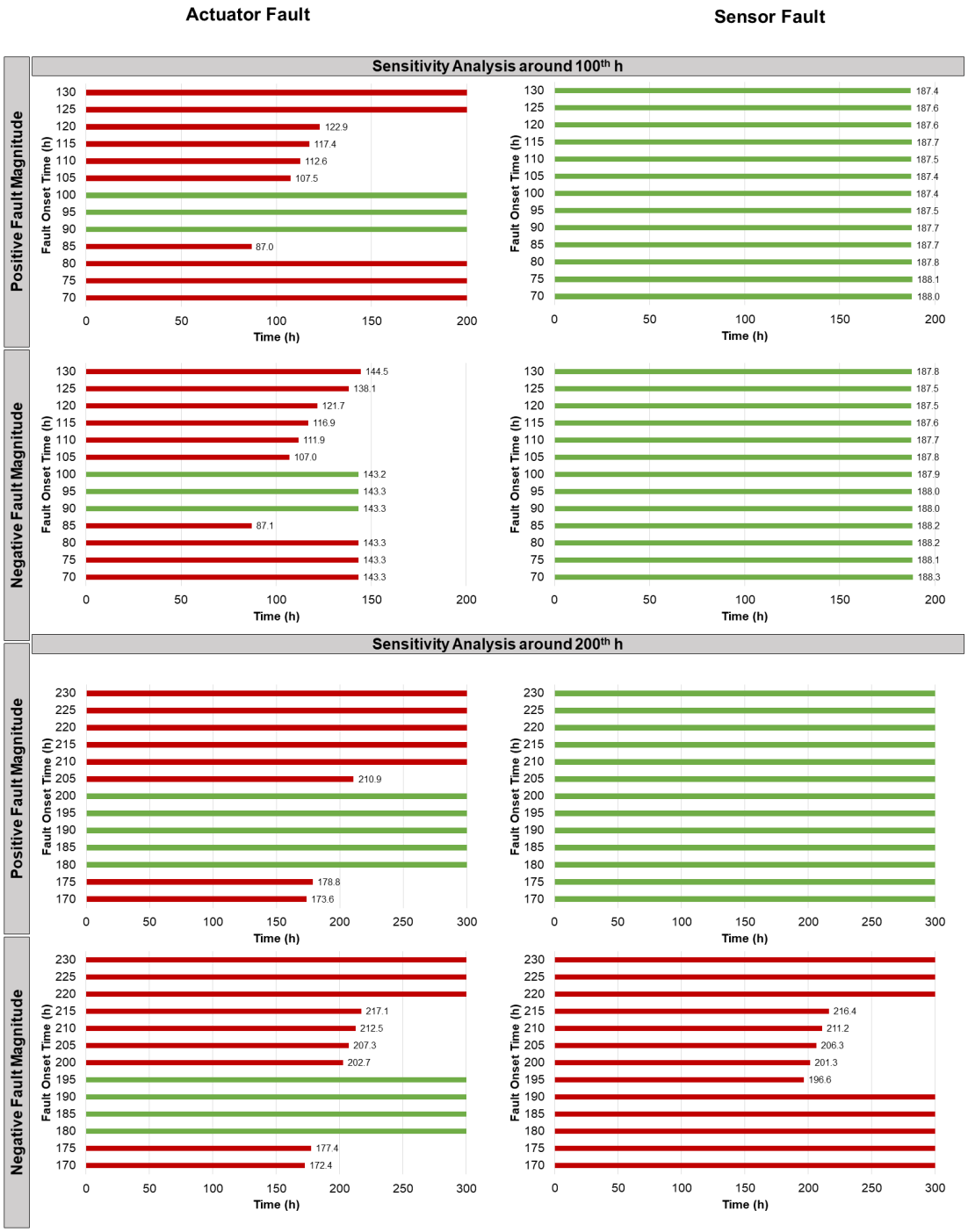


Figure 5.13: Sensitivity analysis of time-specific models built at 100^{th} and 200^{th} h for actuator and sensor faults. The set point deviation threshold is $\pm$ 0.5 K. The green bars highlight that the model is satisfactorily valid. The red bars belong to limited time model validity cases. Note that once a red bar is assigned, the further hours are automatically assigned with red.

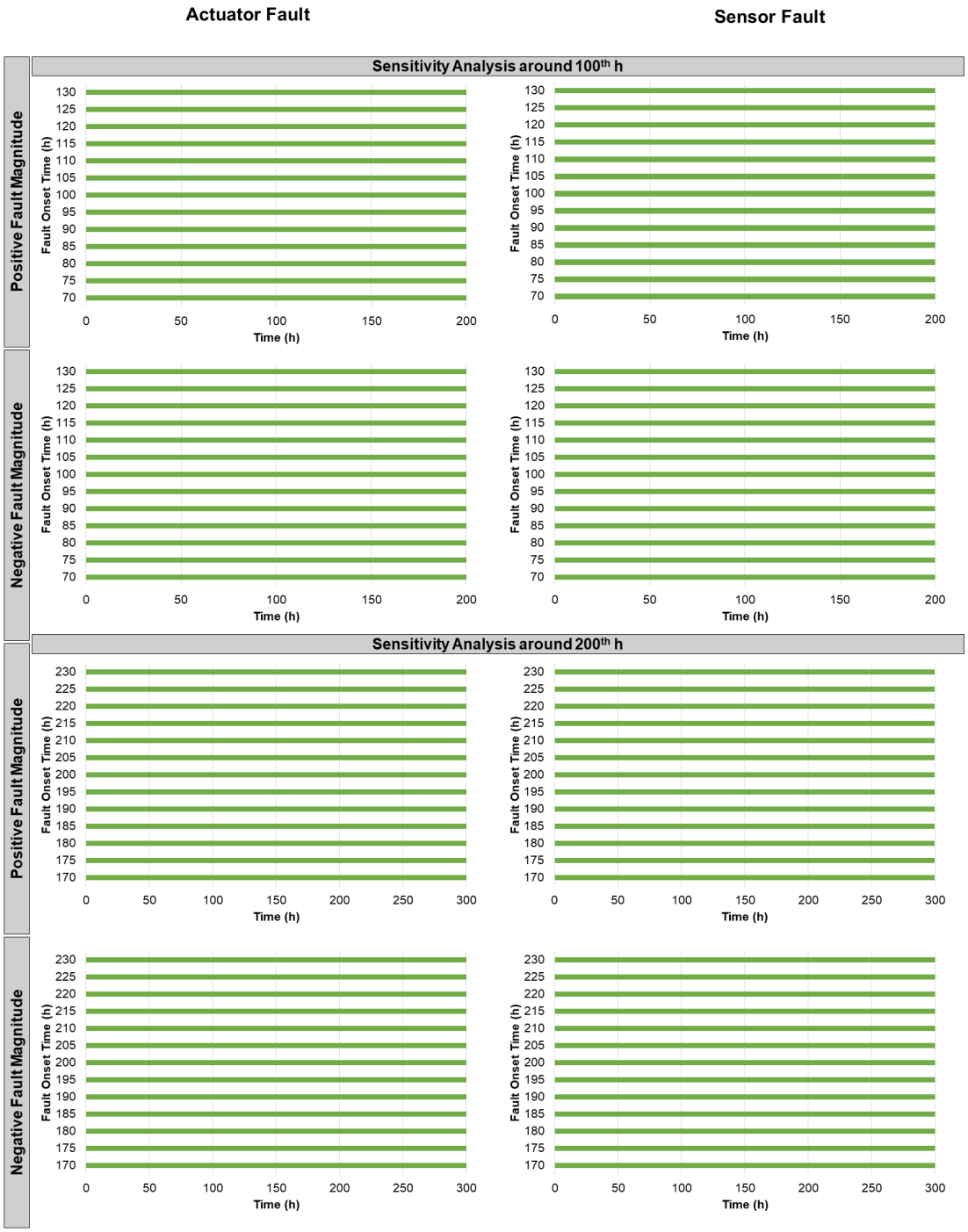


Figure 5.14: Sensitivity analysis of time-specific models built at 100^{th} and 200^{th} h for actuator and sensor faults. The set point deviation threshold is $\pm$ 0.75 K. The green bars highlight that the model is satisfactorily valid. The red bars belong to limited time model validity cases. Note that once a red bar is assigned, the further hours are automatically assigned with red.

5.6 Conclusions

As the effect of smart manufacturing revolution propagates and influences the vision of numerous industrial operations, development of fault-tolerant control system becomes one of the major factors in achieving high process resilience. Traditional corrective maintenance strategies include controller re-tuning which leads to longer process downtime that may adverse the end-product quality and cause higher operation cost. This work proposes a novel parametric fault-tolerant control framework that enables rapid and accurate switches within the offline map of control actions to eliminate process downtime, and maximize process reliability. This further enables attaining higher product quality which leads to higher profit from the operation.

In this work, we present a novel active fault-tolerant strategy and corrective maintenance strategy which benefits from multi-parametric programming and machine learning-based process monitoring. Particularly, we have designed multi-parametric model predictive controller by following the PAROC framework (182) and the s-FDD framework (Chapter 2). The s-FDD framework is used to formulate the fault detection and reconstruction mechanism of the fault-tolerant system, where the built classifiers provide the information on fault existence and regressors yield the fault magnitude and direction estimation. The trained C-SVM models with the optimal feature subset further enables the rapid diagnosis of the detected of fault. The average accuracy of the classifiers is %98.44, and %97.70 for the actuator and sensor faults, respectively. Moreover, the average R^2 of the trained regressors is 0.999 and 0.958 for the actuator and sensor faults, respectively. The presented approach formulates as a novel active fault-tolerant strategy in which an accurate and robust fault detection and reconstruction mechanism is ensured via the s-FDD framework and multi-parametric MPC enables rapid switches between fault-tolerant control actions. Finally, we note that the design of the fault-tolerant mp-MPC can further enable the handling of simultaneous faults as it includes the deviation in both process variables (i.e. reactor temperature and water flow rate) as additional parameters.

6. GROUPING OF COMPLEX SUBSTANCES FOR FACILITATED DECISION-MAKING IN ENVIRONMENTAL HEALTH REGULATIONS

Climate change has become one of the major risk factors for chemical contamination events. Therefore, precise and rapid examination of the complexity of the hazardous chemical exposures is essential to identify the potential adverse health impacts, and subsequently to provide immediate solutions and/or prevent further catastrophic events. At Texas A&M Superfund Research Program (TAMU SRP), we aim to develop comprehensive tools and models for addressing exposure to unknown chemical mixtures, and accordingly design solutions for the community during environmental emergency-related contamination events (TAMU Superfund Research Center, 2017). In this work, we present two applications where data analysis, modeling and dimensionality reduction techniques guide experimental design and decision-making in biomedical and environmental areas.

Under TAMU SRP, we aim to design a framework for optimal grouping of unknown chemical mixtures based on their multi-dimensional analytical chemistry and bioactivity profiles. Detailed characterization of the chemical composition of complex substances, such as products of petroleum refining and environmental mixtures, is an unmet need in both exposure assessment and manufacturing. The inherent complexity and variability in the composition of complex substances obfuscate the choices for their detailed analytical characterization; however, it has been postulated that evaluation of the degree of similarity among substances, rather than their exact chemical composition, is a sensible path towards decision-making. Grouping of sufficiently similar complex substances is a challenge that can be addressed by more informative analytical methods and by streamlined data analysis and visualization that enable communication of the complex data.

In this work, we develop a framework to optimally group complex substances based on their analytical features. The framework includes both unsupervised and supervised analyses. Two datasets of complex oil-derived substances were used. First dataset is from gas chromatography-mass spectrometry (GC-MS) analysis of 20 Standard Reference Materials representing crude oils and oil refining products. Second dataset constitutes of 15 samples of various gas oils that have been analyzed using three analytical techniques: GC-MS, GC×GC-flame ionization detection

(FID), and ion mobility spectrometry-mass spectrometry (IM-MS). We have tested hierarchical clustering using the Pearson correlation as a similarity metric for unsupervised analysis, and built classification models using the Random Forest algorithm for supervised analysis. We also tested the effects of dimensionality reduction for the input data sets by selecting the most informative features. We present quantitative comparative assessment of clustering-based groupings via Fowlkes–Mallows index, and report classification model accuracies in predicting the group of an unknown complex substance. We demonstrate the effect of (i) different grouping methodologies, (ii) data size, and (iii) dimensionality reduction on the grouping quality, and (iv) different analytical techniques on the characterization of the complex substances. While the complexity and variability in chemical composition are an inherent nature of complex substances, this work demonstrates how the choices of the data analysis and visualization methods can impact the communication of their characteristics to delineate sufficient similarity.

The optimal grouping information is then expected to be used in determination of optimal enterosobent material design for different group of complex substances. The environmental chemical contaminants can easily get mobilized, subsequently contaminate soil, and threaten the safety of the municipal water and food supply during environmental emergencies. In order to minimize the adverse health effects of chemical exposures, Texas A&M Superfund Research Program aims to identify and develop novel broad-acting, high-capacity sorbents, enterosorbents, which can be implemented in diets to reduce the bioavailability of chemical mixtures. Therefore, it is essential to interpret the optimal grouping information of complex substances so that the optimal enterosorbent material design can be selected for further mitigation of adverse environmental health impacts.

6.1 Materials and Methods

6.1.1 Materials

In this study, we use two different sets of benchmark analytical chemistry data of: (i) 3 replicates of 20 Standard Reference Materials (SRM) 6.1, and (ii) several recent examples of UVCBs, which are supplied by the European Petroleum Refiners Association AISBL, Concawe division (Brussels, Belgium) and thus referred to as Concawe samples from thereon 6.2. Specifically, SRMs are petroleum-related Certified Reference Materials and provided by the National Institute of Standards and Technology (NIST) (2). In contrast, Concawe samples are obtained from three separate

refinement processes, and categorized as straight run gas oils (SRGOs), other gas oils (OGOs), and vacuum and hydro-treated gas oils (VHGOs). Polycyclic-aromatic hydrocarbon (PAH), saturated hydrocarbon, and crude oil standards were provided by the Texas A&M Geochemical and Environmental Research Group (GERG) (College Station, TX).

Table 6.1: Standard Reference Materials (SRM) samples from de Carvalho Rocha, Schantz (2).

SRM 2722	Crude Oil	Crude Oil	Crude Oil (Heavy-Sweet)	petro203; petro204; petro205
SRM 2721	Crude Oil	Crude Oil	Crude Oil (Light-Sour)	petro274; petro275; petro276
SRM 2779	Crude Oil	Crude Oil	Gulf of Mexico Crude Oil	petro270; petro271; petro272
SRM 1615	Heavy Refinery Product	Gas Oil	Gas Oil	petro207; petro208; petro209
SRM 1848	Heavy Refinery Product	Motor Oil	Motor Oil Additive	petro218; petro219; petro220
SRM 2770	Heavy Refinery Product	RFO	S in Residual Fuel Oil	petro234; petro235; petro236
SRM 1623c	Heavy Refinery Product	RFO	S in Residual Fuel Oil	petro238; petro239; petro240
SRM 1620c	Heavy Refinery Product	RFO	S in Residual Fuel Oil	petro278; petro279; petro280
SRM 2773	Light Refinery Product	Biodiesel	Biodiesel (Animal-based)	petro230; petro231; petro232
SRM 2772	Light Refinery Product	Biodiesel	Biodiesel (Soy-based)	petro266; petro267; petro268
SRM 2723b	Light Refinery Product	Diesel	Low S Diesel	petro226; petro227; petro228
SRM 1624d	Light Refinery Product	Diesel	Sulfur in Diesel	petro214; petro215; petro216
SRM 2771	Light Refinery Product	Diesel	Zero S Diesel	petro222; petro223; petro224
Gasoline	Light Refinery Product	Gasoline	87 Octane Gasoline	petro258; petro259; petro260
SRM 2299	Light Refinery Product	Gasoline	S in gasoline	petro210; petro211; petro212
JP8	Light Refinery Product	Jet Fuel	Jet Fuel	petro246; petro247; petro248
JP5	Light Refinery Product	Jet Fuel	Jet Fuel	petro250; petro251; petro252
Jet Fuel A	Light Refinery Product	Jet Fuel	Jet Fuel	petro254; petro255; petro256
SRM 1617b	Light Refinery Product	Kerosene	S in Kerosene (High Level)	petro242; petro243; petro244
SRM 1616b	Light Refinery Product	Kerosene	S in Kerosene (Low Level)	petro262; petro263; petro264

Table 6.2: Concawe samples.

Sample ID	Manufacturing Class	CAS RN	CAS Name
CON07	OGO	64742-46-7	Distillates (petroleum), hydrotreated middle
CON09	OGO	64742-80-9	Distillates (petroleum), hydro-desulfurized middle
CON01	SRGO	64741-43-1	Gas oils (petroleum), straight-run
CON05	SRGO	64741-43-1	Gas oils (petroleum), straight-run
CON02	SRGO	68814-87-9	Distillates (petroleum), full-range straight-run middle
CON03	SRGO	68814-87-9	Distillates (petroleum), full-range straight-run middle
CON04	SRGO	68915-96-8	Distillates (petroleum), heavy straight-run
CON12	VHGO	64741-49-7	Condensates (petroleum), vacuum tower
CON13	VHGO	64741-58-8	Gas oils (petroleum), light vacuum
CON14	VHGO	64741-77-1	Distillates (petroleum), light hydrocracked
CON15	VHGO	64742-87-6	Gas oils (petroleum), hydrodesulfurized light vacuum
CON16	VHGO	68334-30-5	Fuels, diesel
CON17	VHGO	68476-30-2	Fuel oil, no. 2
CON18	VHGO	68476-31-3	Fuel oil, no. 4
CON20	VHGO	92045-24-4	Gas oils (petroleum), hydrotreated light vacuum

6.1.2 Chemical Fingerprinting and Experimental Data Processing

The analytical chemistry profile of SRMs is derived via Gas Chromatography-Mass Spectrometry (GC-MS) (2), whereas the chemical fingerprint of Concawe substances is assessed with 3 different analytical chemistry techniques: (i) comprehensive two-dimensional gas chromatography with flame ionization detector (GCxGC-FID), (ii) GC-MS, and (iii) Ion Mobility Mass Spectrometry (IM-MS). The detailed experimental procedure is provided in Ferguson (194).

Table 6.3: List of selected analytes from the GC-MS data of SRM samples for grouping analysis.

Selected Analytes	Quantitation Ion
Decalin	138
C1-decalins	152
C2-decalins	166
C3-decalins	180

Table 6.3 – Continued

Selected Analytes	Quantitation Ion
Naphthalene	128
C1-naphthalenes	142
C2-naphthalenes	156
C3-naphthalenes	170
C4-naphthalenes	184
Benzothiophene	134
C1-benzothiophenes	148
C2-benzothiophenes	162
C3-benzothiophenes	176
Biphenyl	154
Acenaphthylene	152
Acenaphthene	154
Dibenzofuran	168
Fluorene	166
C1-fluorenes	180
C2-fluorenes	194
C3-fluorenes	208
Dibenzothiophene	184
C1-dibenzothiophenes	198
C2-dibenzothiophenes	212
C3-dibenzothiophenes	226
C4-dibenzothiophenes	240
Phenanthrene	178
Anthracene	178
C1-phenanthrene/anthracenes	192
C2-phenanthrene/anthracenes	206
C3-phenanthrene/anthracenes	220
C4-phenanthrene/anthracenes	234

Table 6.3 – Continued

Selected Analytes	Quantitation Ion
Naphthobenzothiophene	234
C1-naphthobenzothiophenes	248
C2-naphthobenzothiophenes	262
C3-naphthobenzothiophenes	276
Fluoranthene	202
Pyrene	202
C1-fluoranthene/pyrenes	216
C2-fluoranthene/pyrenes	230
C3-fluoranthene/pyrenes	244
Benz(a)anthracene	228
Chrysene	228
C1-chrysenes/benzo(a)anthracenes	242
C2-chrysenes/benzo(a)anthracenes	256
C3-chrysenes/benzo(a)anthracenes	270
C4-chrysenes/benzo(a)anthracenes	284
Benzo(b)fluoranthene	252
Benzo(k)fluoranthene	252
Benzo(e)pyrene	252
Benzo(a)pyrene	252
Indeno(1,2,3-cd)pyrene	276
Dibenzo(a,h)anthracene	278
Benzo(g,h,i)perylene	276
Perylene	252

The adopted GC-MS data from de Carvalho Rocha, Schantz (2) is a three-dimensional array, which consists of 23,248 elution times, and the 301 masses in the mass spectra for 60 Standard Reference Materials (triplicate runs of 20 samples). In order to reduce the computational com-

plexity of the grouping analysis and the noise in the GC-MS data, we have selected 55 out of 301 m/z values (i.e. analytes) that correspond to Polycyclic Aromatic Hydrocarbons (PAHs) 6.3 and summed over the entire elution time dimension. This yields a two-dimensional (60 X 55) array, which is then used for grouping analysis.

Raw GCxGC-FID, GC-MS, and IM-MS data files are imported into PetroOrg software (195), where quantitative features are extracted. Analyzed abundance values are normalized relative to the total abundance detected for each sample. Compounds are organized by their molecular class and carbon number range, producing a two dimensional data matrix for each sample (194). Detected compounds vary across instruments due to the diversity of utilized analytical methods and their capabilities of the instrument such as resolution and sensitivity. In particular, GC-MS has analyzed 8 separate molecular classes, that is comprised of n-alkanes and polycyclic aromatic compounds, across carbon number ranges of 4 to 34+, producing a total of 248 compositional data points. GCxGC-FID has analyzed 10 separate molecular classes, producing 310 compositional data points. IM-MS has analyzed 13 molecular classes, producing 403 compositional data points. Notably, IM-MS's ionization method, atmospheric photo ionization (APPI), has expanded aromatic and heteroatom detection capability, but significantly reduced the n-alkane profile compared to the other instruments (146).

6.1.3 Data Analysis and Visualization Framework

We use two analysis workflows for grouping complex substances (Figure 6.2). In the unsupervised analysis, complex substances are grouped based on the similarity between the characteristics (i.e. analytical chemistry profiles) of the samples (complex substances) without prior knowledge of sample labels or categories. To evaluate the outcome of such grouping, we include a quantitative metric into the unsupervised analysis workflow to compare the outcome to a previously reported categorization of the samples (i.e. manufacturing classes). The details of the proposed unsupervised analysis workflow is described below. In the supervised analysis, known categorizations/classes of the samples are used to build classification models, which can then be used to predict the class for an unknown substance. This idea is based on the read-across, where similar complex substances that are grouped together according to their physical/chemical properties may have similar effects (140). Independent of which workflow (unsupervised or supervised analy-

sis), the initial common step is data pre-processing, which is crucial to obtain robust and reliable grouping models.

6.1.3.1 Data Pre-processing

Data pre-processing steps include (i) data formatting, (ii) missing data handling, and (iii) data cleaning and scaling. Following these steps ensure data quality in order to build robust and reliable models. The application of these steps to each specific data sets is provided below.

Unfolding three-dimensional data sets into two-dimensional arrays: This step is required when we have unprocessed, three-dimensional, analytical chemistry data (I x J x K). Unfolding can be performed three-ways by selecting one of the three dimensions as row values and merging the other two dimension as columns (i.e. (I x JK), or (J x IK) or (K x IJ)). The GC-MS data of SRMs, after the experimental data processing step described in Section 2.2, is already two-dimensional, hence unfolding is not performed. Therefore, we only perform this step on the raw GCxGC-FID, GC-MS, and IM-MS data sets from Concawe samples. In particular, row values belong to the complex substances, and columns are the analytical features (measurements), which are the combination of carbon number and molecular class composition. This yields an array size of 15 X 310 for GCxGC-FID, 15 X 248 for GC-MS, and 15 X 403 for IM-MS data sets of the Concawe samples.

Missing data handling: The two-dimensional analytical chemistry data sets are then examined to detect any missing points. The traditional missing data handling methods include replacement of the specific point or complete deletion of the corresponding column/row of the missing data point. Particularly, the missing values can be replaced with (i) zeros, (ii) mean of column or row values, or (iii) median of column or row values. There are also advanced missing data handling methods that impute the missing values using machine learning techniques such as K-nearest neighbor (KNN) (196). Of note, when the data set is small, replacement methods are preferred over deletion methods. In this study, the missing data points within analytical chemistry profiles indicate undetected chemical composition for specific molecular class. Therefore, we have replaced the corresponding missing fields with zeros.

Data cleaning (removing outliers): The data sets are cleaned by removing the columns (carbon number – molecular class compositions) if their standard deviations is 0 prior to each data set column-wise scaling step. The threshold is raised 0.05 due to the smaller sample size of Concawe

data set. This reduces the number of features in GCxGC-FID (from 310 to 192), GC-MS (from 248 to 62), and IM-MS data (from 403 to 68) of Concawe samples. This step does not eliminate any measurements from 60 x 55 array of SRM samples.

Scaling of data sets: Final step prior to data analysis is the scaling of the data sets. The clean two-dimensional arrays are scaled by using row-wise min-max scaling, where each row corresponds to a new sample and each column is a new analytical feature. Each row value is scaled by subtracting the minimum value of that row and then dividing it to the range of the corresponding row. Row-wise scaling is not performed on Concawe data sets, where the data was already pre-processed within PetroOrg software (195) and row-wise scaled. Prior to the classification analysis, we have also performed column-wise min-max scaling on the row-wise scaled data arrays. This additional scaling step is required and crucial in order to ensure that each measurement is equally contributing during classification model training and prevent any implicit bias towards a carbon number-molecular class measurement.

6.1.3.2 Unsupervised Analysis Workflow

Unsupervised analysis examines the patterns of data to draw conclusions for the grouping structure of the samples without the reference categorization information. The two most prevalent unsupervised analysis techniques used in the literature are the clustering analysis (197), and the Self-Organizing Maps (SOM) (198), where the former is utilized in the proposed unsupervised analysis workflow (Figure 6.2 - left). The detailed step-by-step description of this workflow is given below. The R Markdown documentation of this analysis through SRM samples is also provided (199).

Similarity calculation for the clustering analysis: The first step in clustering analysis is to calculate a similarity or dissimilarity matrix. The dimensions of this matrix is pp, where p is the number of samples. In this work, we calculate the Pearson correlation of analytical chemistry data to understand the similarity patterns between the samples of complex substances. Other correlation metrics include Kendall's τ or Spearman's ρ rank correlation coefficients (200), which can also be used in this analysis as an alternative to Pearson correlation.

Dimensionality reduction via Singular Value Decomposition: In this work, we use SVD to perform dimensionality reduction prior to clustering analysis in order to obtain grouping results

from reduced data sets. Singular Value Decomposition (SVD) is a matrix factorization technique that decomposes a matrix X into the product of three matrices, $X = UDV^T$, in such a way that U and V are orthonormal and D is a diagonal matrix with positive real elements (i.e. $D = diag(d_1, \ldots, d_k, 0, \ldots, 0), d_1, \ldots, d_k > 0$, $k = rank(D) = rank(X)$) (201; 202). SVD is typically used as a dimensionality reduction methodology and identifies a lower rank matrix B with rank k that best approximates X, where $k = rank(B) \leq rank(X)$. Hence, SVD allows us to identify the best k-dimensional subspace with respect to the number of points of an $p \times n$ data matrix with p-points and n-dimensions, while sustaining the variance of the original data matrix.

In this step, we use the original data sets of SRM and Concawe samples and perform a different scaling other than the row-wise min-max scaling described in the step 4 of data-preprocessing procedure, which is required for the SVD analysis. First, the not-scaled SRM and Concawe data sets are scaled and centered in a row-wise fashion, using the mean and the standard deviation of each row (i.e. z-score normalization). The scaled and centered data is later decomposed using SVD into its singular values (provided as the diagonal matrix D) and singular vectors (left and right singular vectors are given by U and V, respectively). Here, the goal is to reduce the dimensionality of the original analytical data sets while retaining the observed variance. In this work, we have used a variance threshold of 85% to select the number of singular values and vectors for dimensionality reduction. The reduced data is then used as an input to the similarity calculation as described above. SVD is performed on the original analytical chemistry data sets via R statistical software's svd function of the base package.

Under the unsupervised workflow, we generate two sets of grouping results: one with the original data sets, and one with reduced data sets, where the goal is to understand the effect of dimensionality reduction on the grouping quality. Original results are obtained by following steps 1 and 3 to 5, whereas reduced set of results are generated by following all steps.

Hierarchical clustering: In this work, we adopt hierarchical clustering, where we obtain the final grouping information of the samples, as well as the grouping hierarchy. The grouping hierarchy is then visually represented by trees known as "dendrograms". Specifically, we adopt the average linkage technique during hierarchical clustering, which is the most commonly used linkage technique. Average linkage merges the clusters of samples based on the average distance between two clusters. Other linkage techniques include Ward's method and complete linkage, which can also be

used in this analysis as an alternative to average linkage. This analysis is performed in R statistical software by using the hclust function of the stats package.

Quantification of grouping quality via the Fowlkes-Mallows index: Quantitative comparative assessment of clustering results (i.e. grouping dendrograms) is achieved with the incorporation of the Fowlkes-Mallows index (203). Here, we compare the clustering structures based on the analytical chemistry profiles with the known manufacturing-based categorization of the complex substances. F-M index measures the similarity between two hierarchical clustering structures by cutting both dendrograms into specified number of clusters, and counting the number of matching entries in each cluster. We create two sets of hierarchical clustering dendrograms for both the Concawe and SRM samples. First dendrogram is generated by calculating the Euclidean distance between the indices of a reference categorization. Second is by using the row-wise correlation matrices of the Concawe and SRM samples. Next, the dendrograms are cut into the known number of manufacturing classes (i.e. 3 for Concawe, and 3, 9, or 16 for SRM samples) to assess the number of the common complex substances in the obtained clusters. This number is then used to calculate the F-M index.

F-M index is the geometric mean of precision and recall, two machine learning metrics that are widely used in data-driven modeling (197). It is mathematically expressed in the equation shown below.

$$FM = \sqrt{\frac{TP}{TP+FP} \times \frac{TP}{TP+FN}} \tag{6.1}$$

where TP is True Positive, FP is False Positive, and FN is False Negative. TP indicates the number of complex substances that are grouped under category A in terms of manufacturing category and are also grouped under category A in terms of analytical chemistry profile. In contrast, FP and FN yield the number of complex substances that are grouped differently. F-M index varies between 0 and 1, where 0 indicates the absence of any similarity, and 1 indicates the identicalness between clustering structures (i.e. 100% similarity between reference categorization and clustering results). More details on the F-M index and other metrics for clustering comparison can be found in Wagner & Wagner (204). The F-M index is calculated via the FM_index_R function of the dendextend package in R statistical software.

Null Fowlkes-Mallows index calculation for statistical significance: To test the statistical significance of the grouping results, we also calculate the F-M index under null hypothesis of no relation between two clustering dendrograms. In this work, null F-M index calculation is performed by shuffling the group labels of samples using 1000 permutations. Next, the probability value (p-value) of the original, non-permuted, results with respect to 1000 permutations is calculated to quantify the statistical significance ($p - value \leq 0.05$). The null F-M index calculation with 1000 permutations of the group labels is performed via the Bk_permutations of the dendextend package in R statistical software.

6.1.3.3 Supervised Analysis Workflow

Supervised learning problems are classified under two major categories depending on the nature of the predicted output: (i) regression, where the models are built to predict a continuous output, and (ii) classification problems, where the predicted output is discrete, such as finding the optimal group of an unknown sample. Both type of learning problems are ubiquitously used in various engineering and sciences (36; 161; 160; 192; 193; 205). Here, we are interested in classification type of problems, where predictive models are trained and validated by using a data set with known manufacturing-based categorization. Among the wide range of methodologies used for classification problems, the most prevalent and popular algorithms include logistic regression (206), and Naïve Bayes algorithm for linear classifiers. Decision Trees, Random Forest (191), Support Vector Machines (152; 154), and deep learning algorithms are popular examples of nonlinear classifiers. In this work, we adopt the Random Forest algorithm, which is an advanced ensemble learning technique that can derive nonlinear relationships among data samples to build predictive classifiers for grouping complex substances. The models are evaluated by their classification accuracy, and the results are visualized via ToxPi representation (207) for enhanced interpretation (Figure 6.2 - right). The steps of the proposed supervised analysis workflow are provided below and applied to both Concawe and SRM data sets. The documentation of the analysis through SRM samples is created using R Markdown (199).

Parameter tuning for classification models: Parameter tuning is the initial step in classification analysis to maximize the model performance regardless of the selected algorithm. In Random Forest algorithm, the number of analytical features is tuned via grid search using the trainControl

function of the caret package in R statistical software, where each model training is performed using leave-one-out cross validation with 500 decision trees.

Modeling Random Forest classifier and ranking the analytical features: The final Random Forest classifier is then built on the whole data set with 500 decision trees, where each tree is modeled by using the optimal number of analytical features. In addition, the ranking of the analytical features is obtained by calculating the mean decrease in classification accuracy among the 500 decision trees. This analysis is done via the randomForest function of the randomForest package.

Quantitative and visual communication of the results: In order to evaluate the classification model accuracy, initial step is to extract the confusion matrix of the model. Confusion matrix reports the number of true and false predicted samples for each class, defined as a $c \times c$ matrix, where c is the number of classes. Next, the classification accuracy is calculated, which is the percentage of true predicted number of samples from all classes with respect to the total number of samples. In this work, the classification accuracy serves as the quantitative metric for evaluating the grouping of complex substances.

In addition to the quantification of the classification models, we produce Toxicological Prioritization Index (ToxPi) profiles of complex substances by using the ranked analytical feature list from the classification analysis (207; 208; 209). By integrating multiple data sources into overall, weight-of-evidence Toxicological Priority Index (ToxPi) score, and transforming them into clear visual rankings, ToxPi provides an effective way for visual communication of high-dimensional data sets. Particularly in this work, we integrate top 10 most informative chromatographic features that are extracted during classification modeling step, to obtain the ToxPi visualization of complex substances.

6.2 Results and Discussion

6.2.1 Quantifying the Unsupervised Analysis

A recent study (2) has shown that GC-MS combined with unsupervised chemometric analysis can be used to differentiate among complex substances and mixtures. The authors have concluded that the SOM non-linear method proved to be effective in generating a separation model; however, the model is more difficult to interpret than the linear models such as MPCA and PARAFAC. The unified distance matrix of the SOM analysis of the 20 SRMs 6.1 from de Carvalho Rocha,

Schantz (2) is shown in Figure 6.1. It is clear that the replicates of the same samples are clustering well (15 of 20 samples have all 3 replicates in close proximity to each other) in the SOM analysis (Figure 6.1). However, it is less obvious that the SOM analysis can discriminate among the broader categories of samples (3 classes: crude oils, heavy and light refinery products; 9 classes: crude oils, residual fuel oils, gas oil, motor oil, biodiesels, diesels, gasolines, kerosenes and jet fuels). Only jet fuels and gasoline samples of light refinery products are clustered close to each other (Figure 6.1).

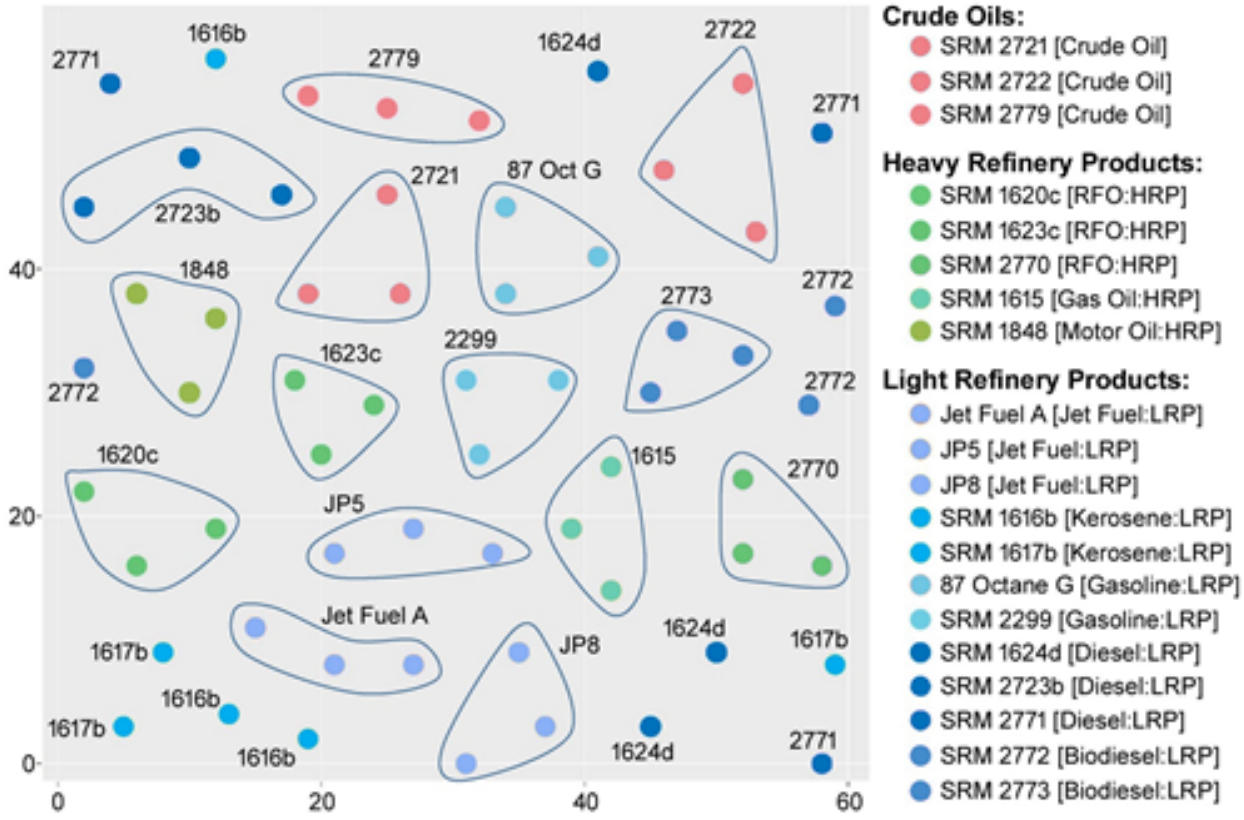


Figure 6.1: SOM recreated from de Carvalho Rocha, Schantz (2).

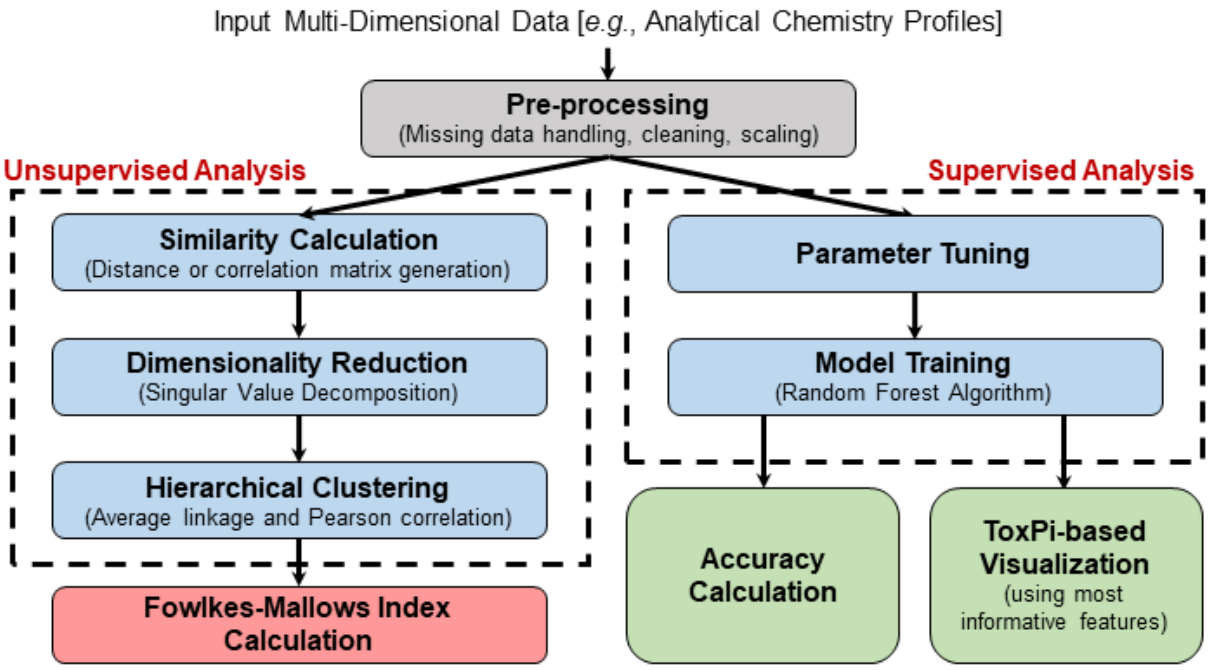


Figure 6.2: Data processing and visualization workflow.

We have used the same data as in de Carvalho Rocha, Schantz (2) to perform unsupervised clustering analysis of the samples (Figure 6.3). The results show that all replicates of 20 substances are clustered tightly, which indicates high reproducibility of the analytical data from GC-MS analysis of these complex substances. However, when 3 or 9 manufacturing classes are considered, the samples are not clustered as tightly as they do in the16 manufacturing classes. For 9 class grouping, replicate samples of gas oils, biodiesels, and motor oils are grouped together in distinct clusters. In 3 class grouping results, only crude oil samples are grouped under one of the three clusters. Although, most of the light refinery products (one gasoline, three diesel, two jet fuels, and two kerosene samples) are clustered together in one of the three groups, one gasoline and two biodiesel samples fall into separate clusters (Figure 6.3).

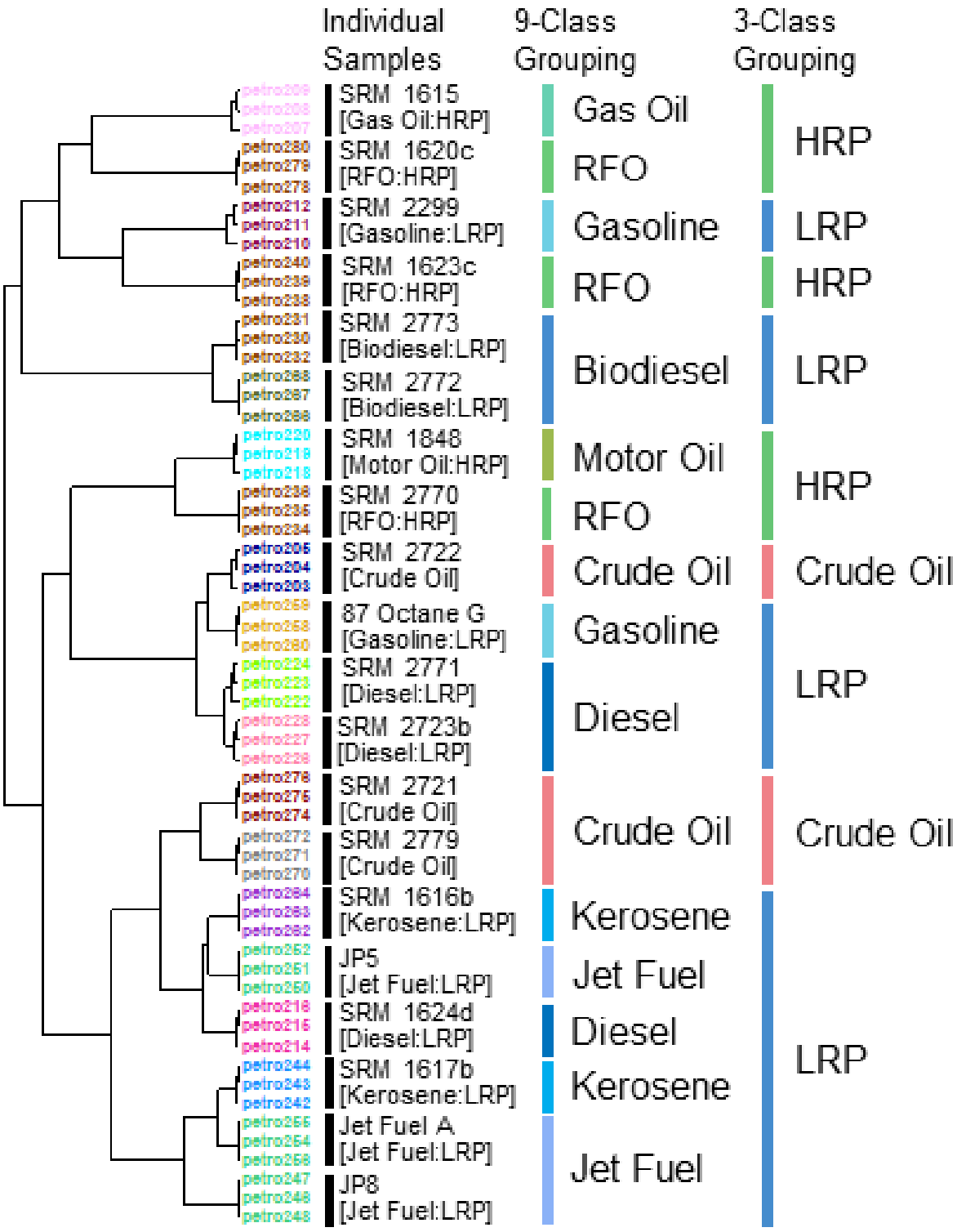
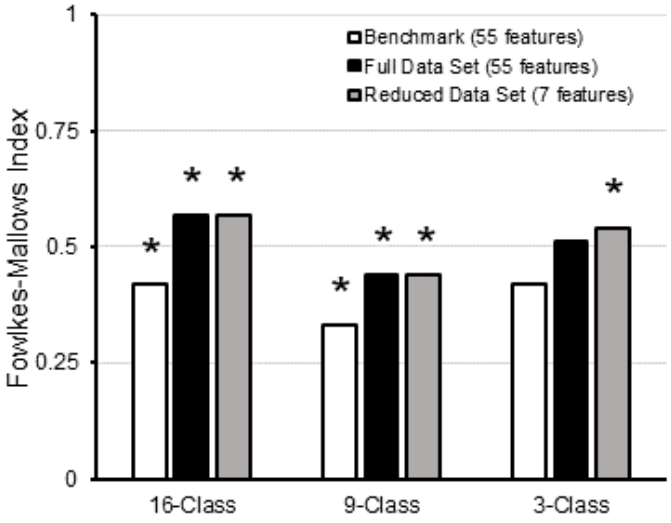


Figure 6.3: Dendrograms for the SRM samples clustering from the reduced data set into 3, 9 and 16 categories.

Next, we aim to quantitatively compare the outcomes of SOM and clustering analyses to the known class identity of these samples. We use the Fowlkes-Mallows index to provide a quantitative metric for such comparisons (203). Although there is no direct method to assess the grouping quality using the SOM analysis, we have extracted the x and y coordinates of each SRM sample on the SOM map (Figure 6.1) and used the Euclidean distance-based similarity matrix to attain a Fowlkes-Mallows index for the SOM-based grouping analysis. The Fowlkes-Mallows index is also used to assess the effect of dimensionality reduction on the outcomes of clustering analyses.



		F-M Index	p-value
16-Class	Benchmark	0.42	0.000
	Full	0.57	0.000
	Reduced	0.57	0.000
9-Class	Benchmark	0.33	0.000
	Full	0.44	0.000
	Reduced	0.44	0.000
3-Class	Benchmark	0.42	0.192
	Full	0.51	0.257
	Reduced	0.54	0.037

Figure 6.4: Fowlkes-Mallows index for the outcomes of clustering of SRM samples.

* indicates that the results are statistically significant.

Figure 6.4 displays the Fowlkes-Mallows indices for SOM-based analysis (called "benchmark" in the figure legend), as well as the full data set of 55 GC-MS features and a reduced set of 7 features after SVD. The p-values of each class are also reported which summarizes the statistical significance of the results (Figure 6.4). The p-values obtained for 3-class grouping are higher than 0.05 for the SOM-based and original data set of SRM samples, which make these results statistically insignificant. The subtle differences that differentiate these materials into 16 categories present themselves as noise when grouped under 3 categories. Hence, the random permutation of these samples lead to higher F-M indices by chance. In contrast, the p-values for 3-class grouping

with the reduced data sets are lower than 0.05. This indicates that dimensionality reduction eliminates the redundant analytical features from the data sets which further reduces the noise, leading to statistically significant results with improved F-M index.

Based on 9 and 16-class groupings of SRMs, one can clearly observe that hierarchical clustering outperforms SOM analysis. The F-M index increases from 0.33 to 0.44, and 0.42 to 0.57 for 9 and 16-class groupings, respectively. Although the dimensionality reduction does not further increase the F-M index for 9 and 16 class groupings, it also does not hinder the grouping quality and provides equally good results with lower number features (7 out of 55).

6.2.2 Importance of Data Sample Size During the Supervised Analysis

Here, we benefit from the read-across hypothesis of "complex substances that group similarly based on manufacturing may impact the environmental health similarly" and move from unsupervised to supervised analysis. To this end, we are building classification models using analytical chemistry profiles of samples. For each of the 20 SRM substances, GC-MS was run three times, which has triplicated the final GC-MS data set size in terms of the number of samples. Thus, an interesting question that we can examine is that how many sample replicates would be adequate to develop data-driven models that can precisely differentiate class patterns.

Figures 6.5 and 6.6 demonstrate the confusion matrices obtained from the trained Random Forest classifiers. These matrices report known ("true") and predicted (through the trained Random Forest classifier) classes for each SRM sample. The results show that we achieve 100% classification accuracy when we use all replicates provided in Table 6.1 and Figure 6.5. The classification accuracy decreases to 65%, 35%, and 15% for 3, 9, and 16-class groupings if we only use 1 out 3 replicates (Table 6.1, Figure 6.6). The main reason for this is that the number of samples per group decreases as the number of classes increases. In particular, 14 out of 16 classes are represented with only a single sample during model training for the 16-class predictions (Figure 6.6C). Similarly, 5 out 9 classes are represented with only a single sample during model training for the 9-class predictions (Figure 6.6B). This decrease in the amount of information per class makes model learning significantly challenging (Table 6.4). Hence, we can conclude that single sample per class does not provide adequate information to capture the individual class characteristics. Moreover, the high-dimensional nature of the GC-MS data with 55 features further hinders the classification

accuracy of SRM materials when using only one sample per category. Yet, the prediction accuracies of the classifiers for each analysis are higher than the random prediction proving that they are statistical significant. This is validated through p-value calculations by using the original and 1000 random permutation grouping results (6.4). The confusion matrices generated from the average of 1000 permutations of SRM samples are provided in Figures 6.7 and 6.8 for 3 and 1 replicates, respectively.

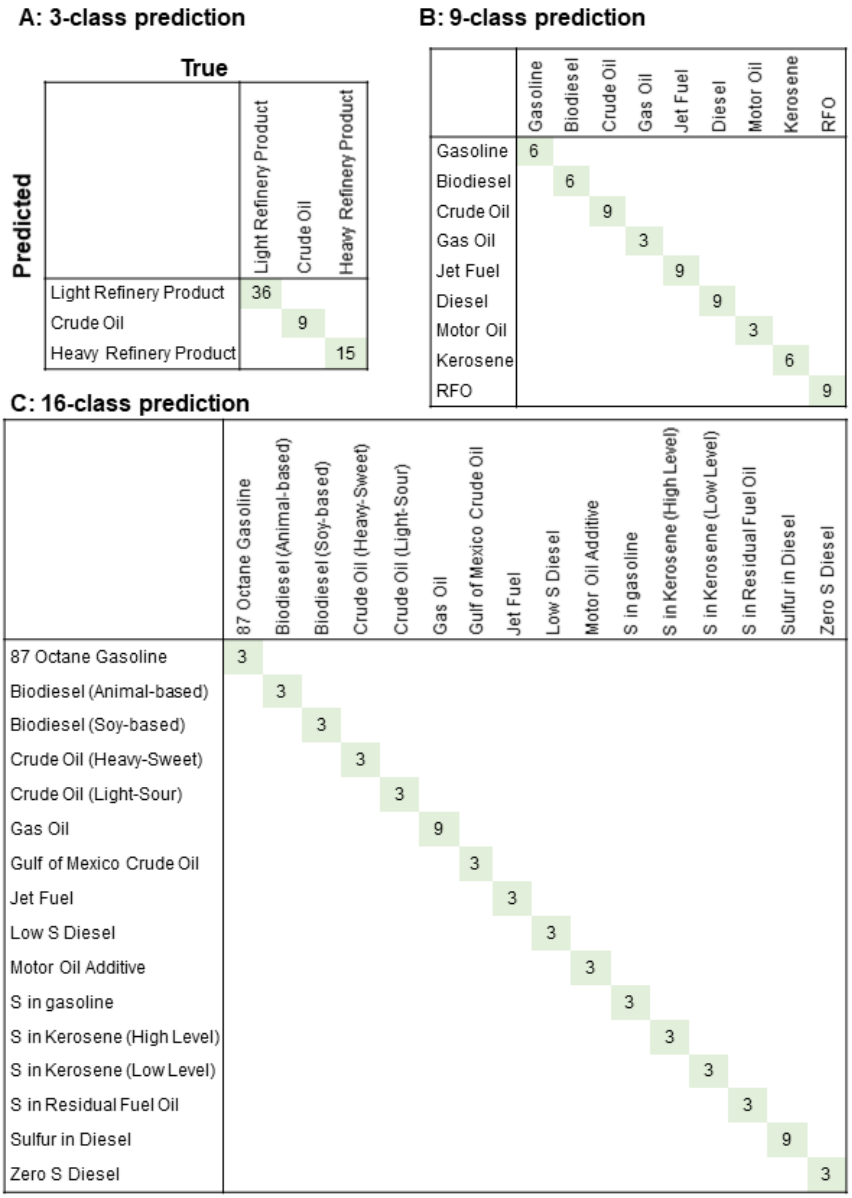


Figure 6.5: Confusion matrices for SRM sample classification with 3 replicates.

(A) 3-class, (B) 9-class, and (C) 16-class grouping.

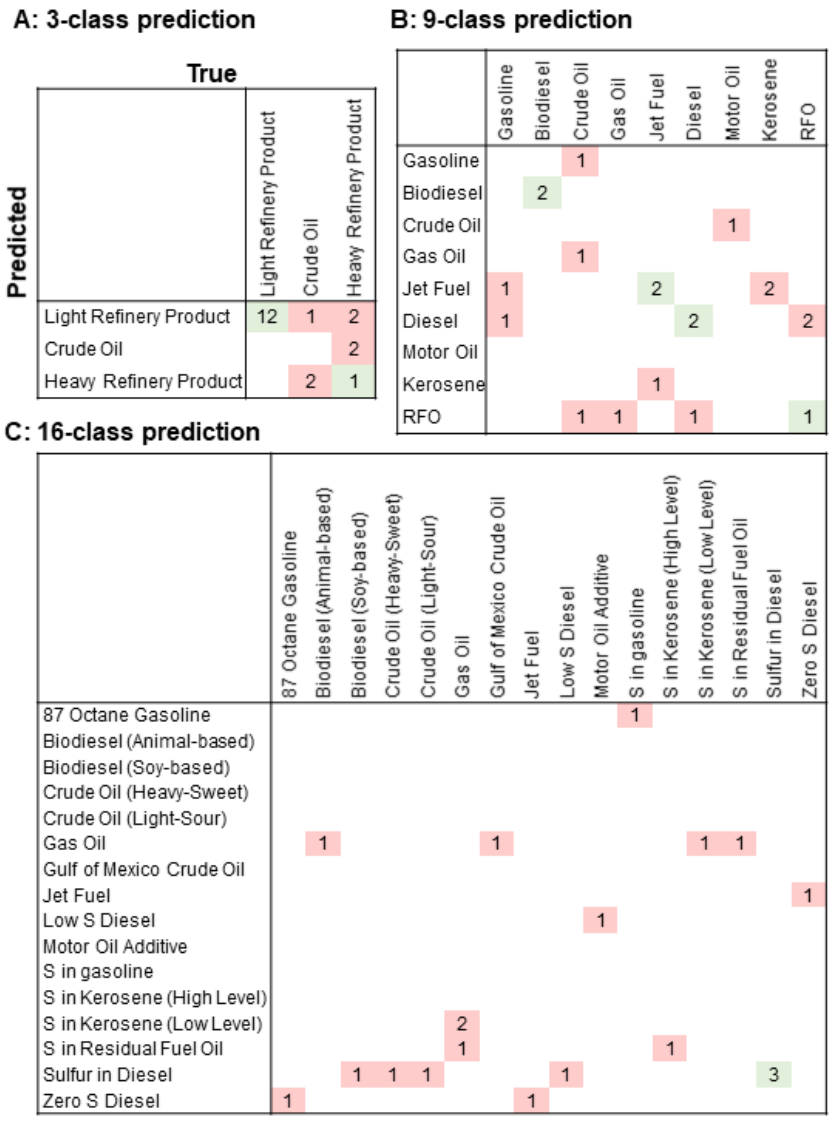


Figure 6.6: Confusion matrices for SRM sample classification with 1 replicate.

(A) 3-class, (B) 9-class, and (C) 16-class grouping.

Similar trend is observed with Concawe samples, where we build classification models using only 1 replicate of each sample (Table 6.5, Figure 6.9). The results demonstrate that classification model accuracies for the Concawe samples are not satisfactory, where the only statistically significant result is obtained from IM-MS data with 60% classification accuracy. Therefore, in order to build an accurate classification model, we need higher number of experimental replicates for each particular complex substance to capture and learn the nonlinear characteristics of their chem-

ical complexity. Although clustering can group the samples accurately independent of the sample size, given that measurements are significantly distinct from each other, data sample size is essential during the classification model building. Nonetheless, each experimental replicate leads to an additional cost and requires extra time, and resources. Thus, minimizing the number of sample replicates while achieving accurate predictive classifiers is of utmost importance. In this work, we have observed that, given high quality analytical chemistry data, 3 replicates suffice to build accurate and robust classifiers. It is important to note that the sample size is critical during the model training phase, where the models benefit from higher number of samples. However, this is not the case for the testing phase where a single experiment is sufficient to predict its class information of an unknown complex substance.

A: 3-class prediction

Predicted \ True	Light Refinery Product	Crude Oil	Heavy Refinery Product
Light Refinery Product	22.39	5.77	9.57
Crude Oil	5.05	1.12	2.08
Heavy Refinery Product	8.56	2.11	3.35

B: 9-class prediction

	Gasoline	Biodiesel	Crude Oil	Gas Oil	Jet Fuel	Diesel	Motor Oil	Kerosene	RFO
Gasoline	0.49	0.59	0.89	0.29	0.86	0.92	0.34	0.63	0.88
Biodiesel	0.57	0.50	0.88	0.29	0.94	0.91	0.32	0.67	0.87
Crude Oil	0.93	0.91	1.23	0.49	1.38	1.35	0.47	0.91	1.40
Gas Oil	0.27	0.31	0.48	0.09	0.46	0.47	0.15	0.28	0.49
Jet Fuel	0.92	0.92	1.41	0.48	1.25	1.39	0.48	0.91	1.42
Diesel	0.95	0.93	1.36	0.43	1.38	1.20	0.42	0.91	1.44
Motor Oil	0.33	0.29	0.45	0.14	0.43	0.40	0.10	0.27	0.43
Kerosene	0.63	0.66	0.88	0.29	0.95	0.95	0.29	0.51	0.90
RFO	0.91	0.89	1.42	0.50	1.36	1.40	0.44	0.90	1.18

C: 16-class prediction

	87 Octane Gasoline	Biodiesel (Animal-based)	Biodiesel (Soy-based)	Crude Oil (Heavy-Sweet)	Crude Oil (Light-Sour)	Gas Oil	Gulf of Mexico Crude Oil	Jet Fuel	Low S Diesel	Motor Oil Additive	S in gasoline	S in Kerosene (High Level)	S in Kerosene (Low Level)	S in Residual Fuel Oil	Sulfur in Diesel	Zero S Diesel
87 Octane Gasoline	0.10	0.15	0.16	0.16	0.14	0.45	0.14	0.14	0.16	0.13	0.15	0.19	0.15	0.16	0.48	0.17
Biodiesel (Animal-based)	0.15	0.08	0.14	0.14	0.17	0.42	0.16	0.13	0.15	0.15	0.15	0.15	0.17	0.14	0.45	0.16
Biodiesel (Soy-based)	0.17	0.15	0.12	0.18	0.16	0.47	0.16	0.16	0.13	0.15	0.15	0.17	0.14	0.15	0.42	0.14
Crude Oil (Heavy-Sweet)	0.15	0.15	0.16	0.09	0.17	0.44	0.16	0.13	0.15	0.16	0.15	0.14	0.15	0.16	0.47	0.15
Crude Oil (Light-Sour)	0.13	0.17	0.15	0.17	0.09	0.45	0.14	0.15	0.15	0.14	0.14	0.17	0.17	0.16	0.44	0.16
Gas Oil	0.45	0.45	0.49	0.45	0.48	1.32	0.47	0.52	0.47	0.46	0.46	0.41	0.47	0.46	1.43	0.45
Gulf of Mexico Crude Oil	0.16	0.16	0.15	0.17	0.15	0.43	0.10	0.15	0.15	0.14	0.16	0.14	0.17	0.15	0.44	0.14
Jet Fuel	0.14	0.13	0.14	0.14	0.16	0.50	0.15	0.10	0.16	0.14	0.14	0.15	0.14	0.13	0.44	0.17
Low S Diesel	0.16	0.15	0.14	0.14	0.15	0.46	0.16	0.16	0.10	0.15	0.16	0.14	0.15	0.16	0.49	0.16
Motor Oil Additive	0.14	0.16	0.16	0.15	0.13	0.44	0.14	0.14	0.15	0.12	0.15	0.14	0.15	0.16	0.45	0.17
S in gasoline	0.16	0.16	0.16	0.15	0.15	0.44	0.15	0.13	0.16	0.15	0.09	0.15	0.13	0.15	0.46	0.15
S in Kerosene (High Level)	0.16	0.14	0.17	0.14	0.15	0.41	0.14	0.15	0.13	0.16	0.15	0.10	0.15	0.15	0.48	0.15
S in Kerosene (Low Level)	0.13	0.18	0.14	0.16	0.17	0.47	0.18	0.14	0.15	0.15	0.15	0.17	0.09	0.16	0.44	0.14
S in Residual Fuel Oil	0.16	0.15	0.14	0.15	0.17	0.46	0.14	0.15	0.15	0.15	0.15	0.16	0.17	0.08	0.46	0.14
Sulfur in Diesel	0.50	0.47	0.44	0.47	0.42	1.40	0.45	0.49	0.47	0.47	0.49	0.47	0.46	0.48	1.22	0.47
Zero S Diesel	0.16	0.17	0.14	0.14	0.15	0.46	0.15	0.18	0.17	0.18	0.16	0.16	0.16	0.14	0.45	0.09

Figure 6.7: Average confusion matrices of 1000 permutations for SRM sample classification. (A) 3-class grouping, (B) 9-class grouping, and (C) 16-class grouping with 3 replicates.

Table 6.4: Classification accuracy of SRM samples.

Prediction class type	Number of sample replicates used	Classification accuracy	Classification accuracy (permuted)	p-value
3-class	3	100%	$44.8 \pm 7.0\%$	0.000
	1	65%	$48.0 \pm 9.2\%$	0.023
9-class	3	100%	$10.9 \pm 5.1\%$	0.000
	1	35%	$6.6 \pm 6.9\%$	0.000
16-class	3	100%	$6.5 \pm 4.1\%$	0.000
	3	100%	$6.5 \pm 4.1\%$	0.000
16-class	1	15%	$4.2 \pm 5.1\%$	0.019

Table 6.5: Classification accuracy of Concawe samples.

Prediction class type	Number of sample replicates used	Classification accuracy	Classification accuracy (permuted)	p-value
3-class (VHGO, SRGO, OGO)	GC-MS	40%	$39.9 \pm 13.5\%$	0.395
	GCxGC-FID	46.7%	$39.4 \pm 13.9\%$	0.222
	IM-MS	60.0%	$41.4 \pm 12.0\%$	0.047

A: 3-class prediction

Predicted \ True	Light Refinery Product	Crude Oil	Heavy Refinery Product
Light Refinery Product	8.98	2.40	4.17
Crude Oil	0.82	0.10	0.31
Heavy Refinery Product	2.20	0.50	0.52

B: 9-class prediction

	Gasoline	Biodiesel	Crude Oil	Gas Oil	Jet Fuel	Diesel	Motor Oil	Kerosene	RFO
Gasoline	0.08	0.19	0.31	0.09	0.28	0.30	0.11	0.19	0.27
Biodiesel	0.19	0.06	0.25	0.07	0.28	0.27	0.08	0.18	0.28
Crude Oil	0.36	0.35	0.30	0.18	0.55	0.56	0.20	0.35	0.58
Gas Oil	0.06	0.07	0.10		0.10	0.11	0.03	0.07	0.09
Jet Fuel	0.37	0.36	0.52	0.17	0.28	0.54	0.15	0.33	0.57
Diesel	0.36	0.36	0.57	0.19	0.58	0.28	0.19	0.35	0.56
Motor Oil	0.08	0.06	0.10	0.04	0.11	0.10		0.08	0.10
Kerosene	0.18	0.18	0.28	0.08	0.27	0.26	0.09	0.07	0.27
RFO	0.33	0.38	0.58	0.19	0.56	0.58	0.16	0.39	0.27

C: 16-class prediction

	87 Octane Gasoline	Biodiesel (Animal-based)	Biodiesel (Soy-based)	Crude Oil (Heavy-Sweet)	Crude Oil (Light-Sour)	Gas Oil	Gulf of Mexico Crude Oil	Jet Fuel	Low S Diesel	Motor Oil Additive	S in gasoline	S in Kerosene (High Level)	S in Kerosene (Low Level)	S in Residual Fuel Oil	Sulfur in Diesel	Zero S Diesel
87 Octane Gasoline		0.04	0.04	0.03	0.03	0.13	0.04	0.04	0.03	0.03	0.04	0.05	0.03	0.04	0.14	0.04
Biodiesel (Animal-based)	0.04		0.04	0.05	0.05	0.15	0.04	0.05	0.03	0.04	0.04	0.03	0.03	0.03	0.12	0.03
Biodiesel (Soy-based)	0.05	0.04		0.04	0.03	0.12	0.05	0.03	0.04	0.03	0.05	0.05	0.05	0.05	0.15	0.04
Crude Oil (Heavy-Sweet)	0.04	0.06	0.04		0.04	0.13	0.03	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.13	0.05
Crude Oil (Light-Sour)	0.04	0.03	0.04	0.04		0.12	0.03	0.04	0.03	0.04	0.04	0.03	0.04	0.04	0.12	0.03
Gas Oil	0.27	0.25	0.24	0.26	0.24	0.42	0.23	0.26	0.25	0.22	0.25	0.23	0.25	0.22	0.81	0.23
Gulf of Mexico Crude Oil	0.03	0.04	0.04	0.03	0.04	0.12		0.04	0.05	0.04	0.03	0.04	0.04	0.04	0.13	0.04
Jet Fuel	0.04	0.03	0.03	0.03	0.05	0.13	0.04		0.05	0.05	0.04	0.04	0.03	0.04	0.13	0.04
Low S Diesel	0.03	0.05	0.04	0.03	0.03	0.14	0.04	0.04		0.04	0.04	0.05	0.03	0.04	0.13	0.04
Motor Oil Additive	0.03	0.04	0.04	0.04	0.03	0.14	0.04	0.05	0.04		0.05	0.05	0.04	0.05	0.12	0.05
S in gasoline	0.04	0.04	0.05	0.04	0.04	0.14	0.03	0.04	0.04	0.05		0.05	0.04	0.04	0.10	0.04
S in Kerosene (High Level)	0.05	0.05	0.05	0.04	0.04	0.14	0.05	0.03	0.05	0.05	0.04		0.04	0.05	0.13	0.04
S in Kerosene (Low Level)	0.03	0.03	0.04	0.04	0.04	0.13	0.04	0.03	0.03	0.04	0.04	0.04		0.04	0.13	0.04
S in Residual Fuel Oil	0.04	0.03	0.03	0.02	0.05	0.12	0.04	0.04	0.04	0.04	0.05	0.04	0.03		0.10	0.04
Sulfur in Diesel	0.25	0.23	0.27	0.25	0.26	0.76	0.26	0.23	0.25	0.24	0.22	0.25	0.26	0.24	0.42	0.25
Zero S Diesel	0.04	0.04	0.02	0.05	0.03	0.12	0.04	0.04	0.04	0.05	0.04	0.04	0.04	0.04	0.14	

Figure 6.8: Average confusion matrices of 1000 permutations for SRM sample classification.

(A) 3-class grouping, (B) 9-class grouping, and (C) 16-class grouping with 1 replicate.

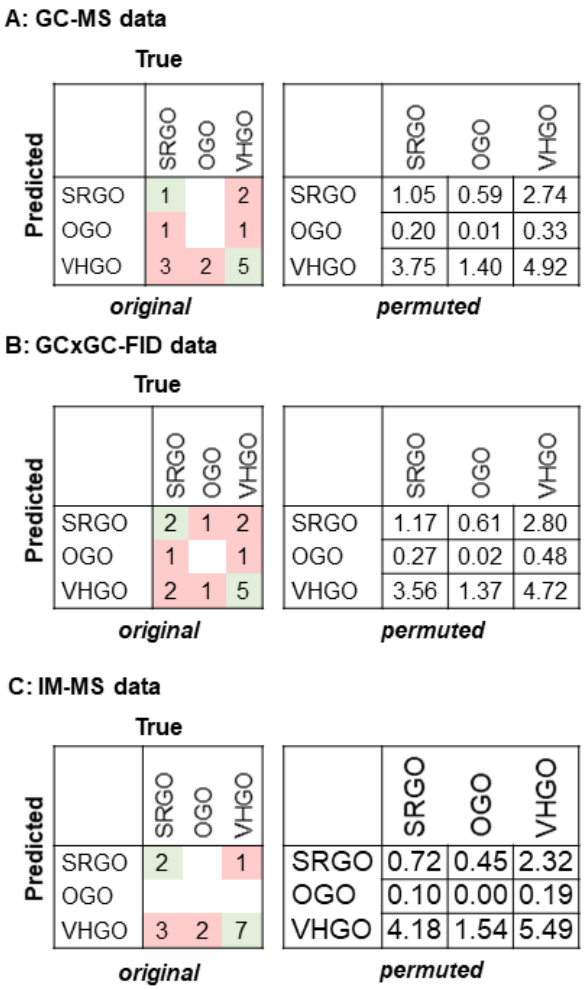


Figure 6.9: Original and average confusion matrices of 1000 permutations for Concawe sample classification.

(A) GC-MS, (B) GCxGC-FID, and (C) IM-MS data.

6.2.3 Facilitation of Data Interpretation via ToxPi Representation

In addition to developing highly accurate classifier models to predict group/class information of an unknown complex substance, we also report the top 10 most informative features that distinctively identify the class patterns of SRM materials (Table 6.6). These informative features help us to facilitate the visual communication of the findings via ToxPi visualization as shown in Figure 6.10.

The ToxPi profiles of SRM samples successfully demonstrate the distinct nature of gas/motor

oils, biodiesels, and crude oils (with the exception of SRM 2722) with respect to the rest of SRMs. Specifically for crude oils, all of the top 10 chromatographic feature help to identify crude oils among SRMs. Whereas for gas/motor oils, the profiles reveal the importance of C2-napthobenzothiophenes for further identification between them. Moreover, the ToxPi profiles show that C3-phenanthrene/anthracenes, C2-naphthobenzothiophenes, and benzothiophene measurements are the characteristics of biodiesel samples and can differentiate them from the rest of the SRMs. We also observe the high similarity among a subgroup of light refinery products that includes jet fuels, kerosenes and gasolines, where the weight of benzothiophene remains to be the unique characteristic among all of them. This proves that the GC-MS data could not provide clear distinction among these substances. Finally, we note the major difference between diesel samples. Unlike the other two diesel samples, SRM 2723b and SRM 2771, most of the top 10 selected analytical features are significant for identifying SRM 1624d. In particular, dibenzothiophene, C1-dibenzothiophenes and C4-naphthalenes are the distinct measurements that differentiate SRM 1624d from the rest of the SRMs the most. Furthermore, the PCA of the extracted ToxPi scores helps us to depict the distinction between the complex substances by using the most informative analytical feature information (Figure 6.11).

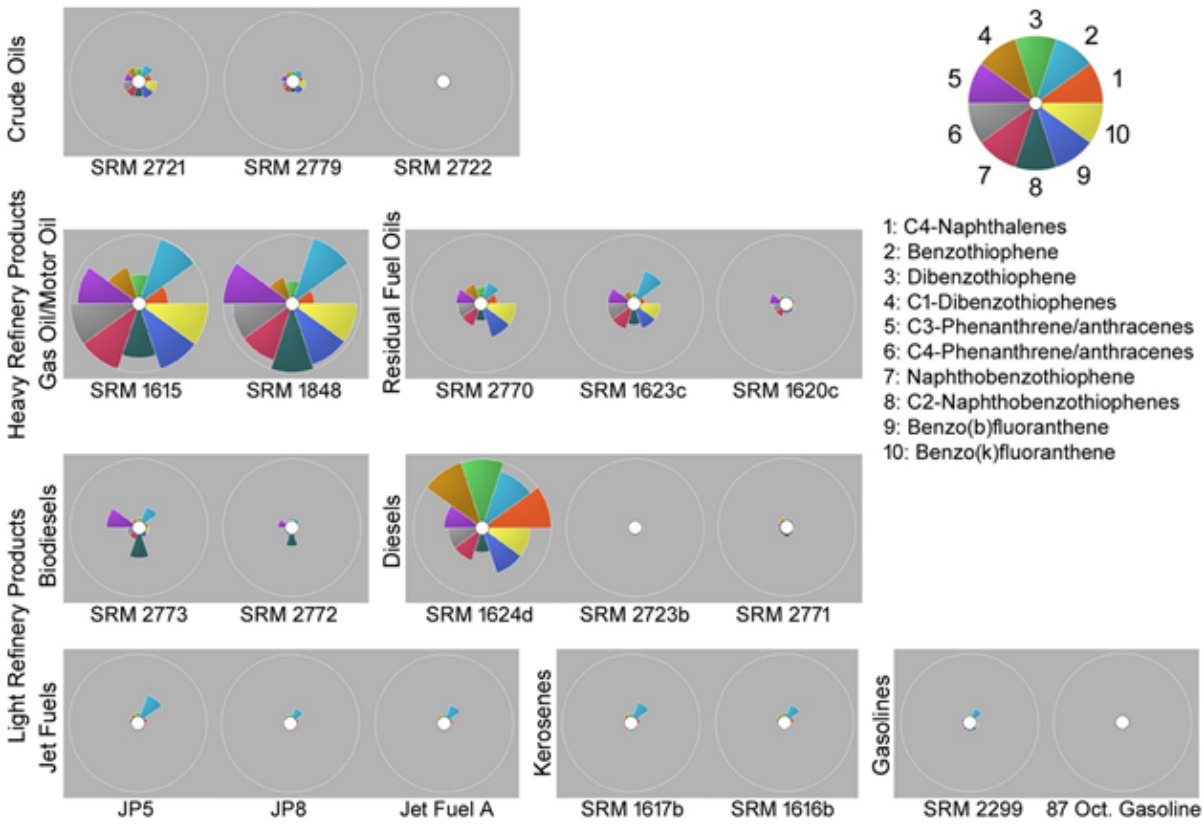


Figure 6.10: ToxPi visualization of SRM samples using top 10 most informative chromatographic features.

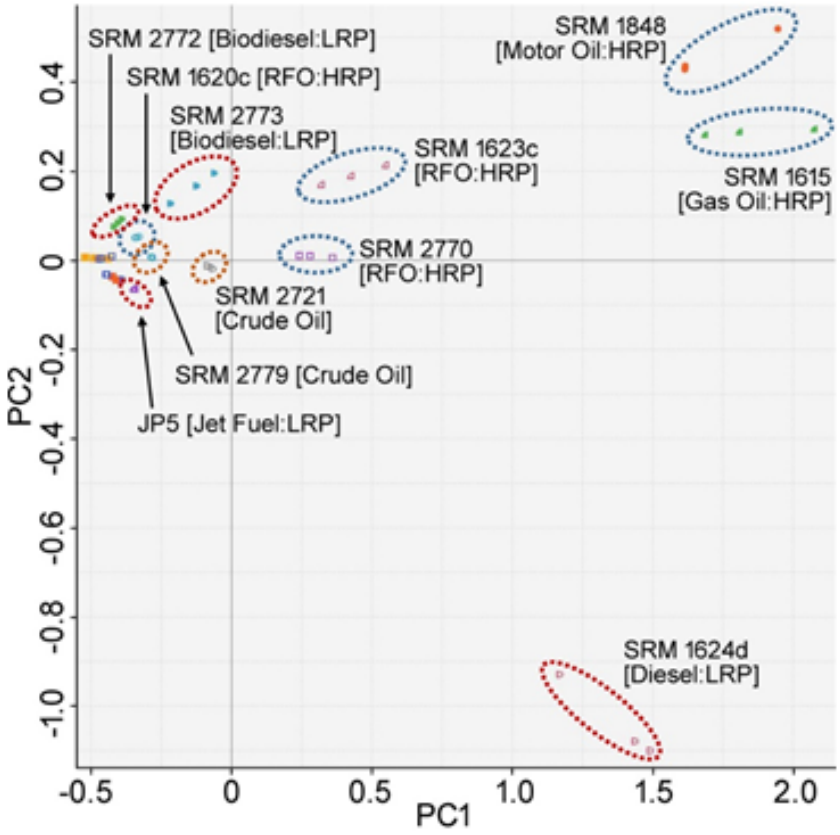


Figure 6.11: PCA of ToxPi scores.

6.2.4 Comparison of GCxGC-FID, GC-MS, and IM-MS Techniques via Fowlkes-Mallows Index

In addition to quantifying the grouping quality and class information, it is imperative to investigate the appropriate analytical chemistry technique that produces the optimal grouping for substances with complex chemistries. The majority of regulatory and standardized chemical compositional analysis protocols utilize GC-MS as the instrument of choice to fingerprint UVCB substances. Generally, a GC-MS instrument employs a capillary column, heated by an oven at a predetermined temperature gradient in order to separate compounds by boiling point and polarity. The eluting compounds are then ionized and analyzed by a detector. Since molecules of specific molecular classes maintain distinct mass ion fracture patterns, a GC-MS is able to differentiate ion signals from multiple compounds. However, the column peak capacity of a GC-MS can become overloaded, causing a baseline hump termed as an unresolved complex mixture (UCM). In such cases, the column no longer has the resolving power to separate all the compounds within the sample, which is typically observed in petroleum substance analysis, since an individual petroleum substance contains more than 10,000 different chemical compounds. This may limit the amount of molecules that can be effectively differentiated by the instrument, and hinder a robust chemical fingerprint production.

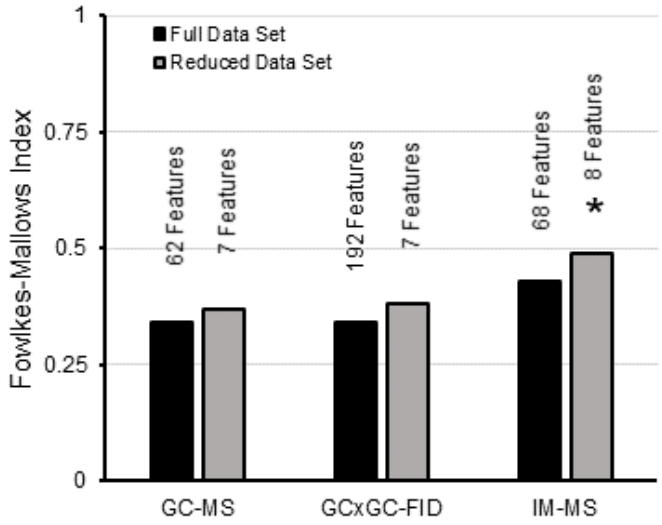
However, in recent years, instrument resolution power and sensitivity has increased, allowing for more detailed characterization of complex substances. The incorporation of two gas chromatography columns with different selectivity (GCxGC-FID) increases the peak capacity of the instrument and allows for improved separation of molecules that form a UCM under GC-MS analysis. Moreover, ion mobility mass spectrometry (IM-MS) incorporates unique ionization methods, electron spray (ESI) or atmospheric photo ionization (APPI), along with separation techniques based on size, shape, and charge of the ionized molecule. This further increases analytical sensitivity and enables improved chemical fingerprinting. Although these two techniques further enhance the ability to characterize complex substances like petroleum products, their application is still novel and not widely studied within the scientific, regulatory, or industrial communities (2; 210). Despite the technological advances that are introduced by GCxGC-FID, and IM-MS techniques over GC-MS, there is no evidence examining any potential improvements on complex substance

Table 6.6: Top 10 most informative GC-MS chromatographic features with respect to the classification accuracy of the petroleum substances.

GC-MS Chromatographic Feature	3-class predictions rank	3-class predictions Mean Decrease in Accuracy	9-class predictions rank	9-class predictions Mean Decrease in Accuracy	16-class predictions rank	16-class predictions Mean Decrease in Accuracy
C4-Naphthalenes	6	6.82	13	8.68	1	10.38
Naphthobenzothiophene	10	6.54	14	8.56	7	9.44
C2-Naphthobenzothiophenes	24	6.12	2	9.09	10	9.4
Benzothiophene	21	6.25	3	9.08	14	9.25
C3-Phenanthrene/anthracenes	5	6.9	6	8.86	33	8.55
C4-Phenanthrene/anthracenes	8	6.58	35	7.76	3	9.57
Benzo(b)fluoranthene	19	6.28	20	8.42	8	9.42
Dibenzothiophene	41	5.76	4	9.02	4	9.56
C1-Dibenzothiophenes	2	7.1	22	8.35	26	8.69
Benzo(k)fluoranthene	27	6.06	16	8.53	11	9.34

grouping. Therefore, we utilize the Fowlkes-Mallows index to provide comparative assessment between these three analytical chemistry techniques using the Concawe samples.

Figure 6.12 demonstrates that GCxGC-FID and GC-MS yield statistically insignificant F-M indices due to the limited number of sample size. IM-MS is the only one yielding statistically significant results, only after dimensionality reduction, which provides the most useful information to reveal the class differences among the Concawe samples. Their corresponding clustering dendrograms are provided in Figure 6.13. Specifically, the F-M index of the grouping of Concawe samples with 8 features generated via the IM-MS technique is 0.49. Although we cannot draw specific conclusions among GCxGC-FID and GC-MS, we can report that IM-MS performs superior than the other two techniques in terms of capturing the chemical characteristics of complex substances.



		F-M Index	p-value
GC-MS	Full	0.34	0.723
	Reduced	0.37	0.176
GCxGC-FID	Full	0.34	0.723
	Reduced	0.38	0.169
IM-MS	Full	0.43	0.166
	Reduced	0.49	0.035

Figure 6.12: F-M index for the outcomes of clustering of Concawe samples analyzed using 3 different techniques.

* indicates that the results are statistically significant.

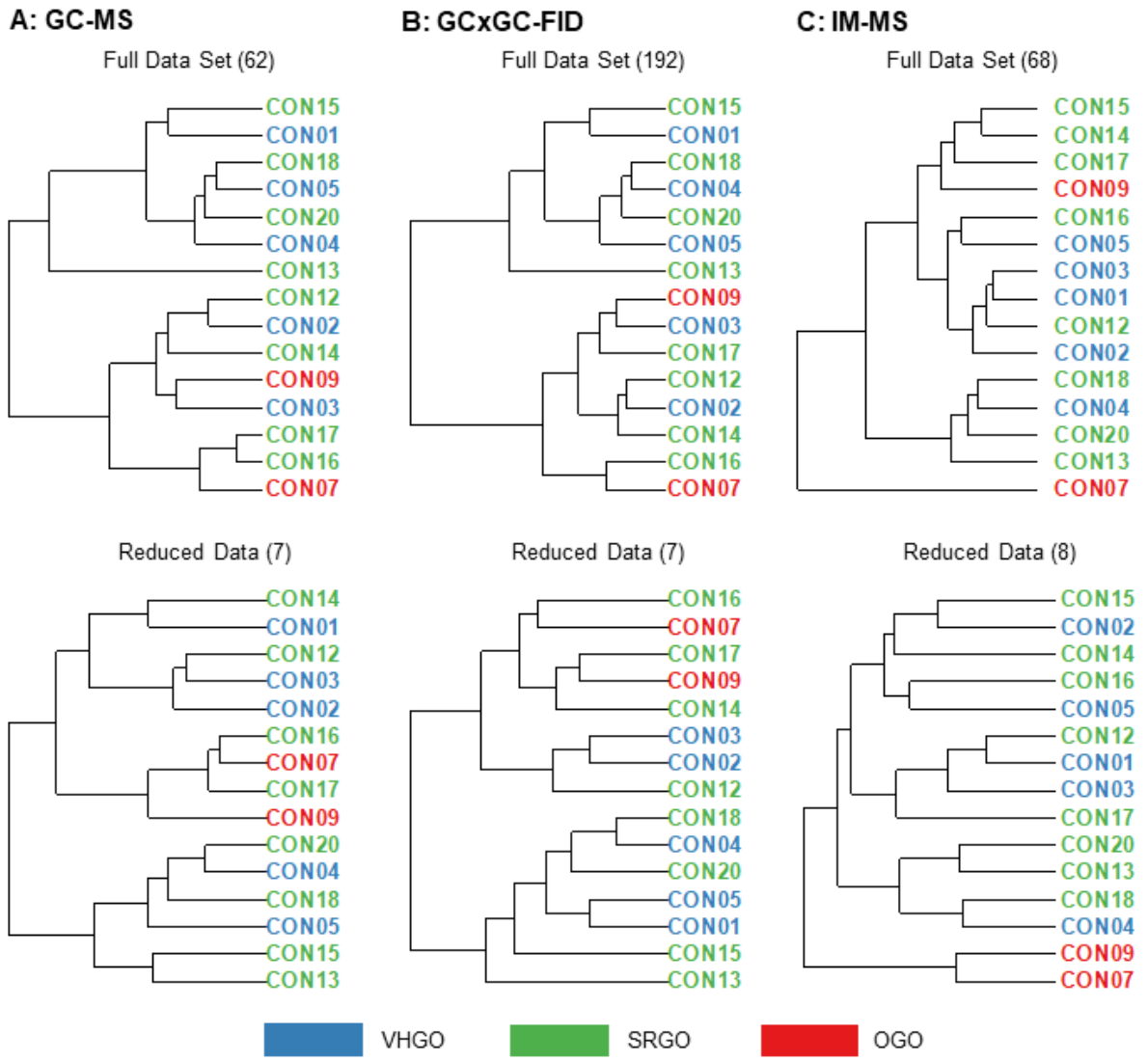


Figure 6.13: Dendrograms for Concawe samples clustering from the reduced data set analyzed using 3 different techniques.

6.3 Conclusions

In this study, we establish a data-driven framework for optimal grouping complex chemical substances based on their chemical characteristics, and providing quantitative and visual evaluation to facilitate the interpretation of the complex chemical nature of substances/mixtures. The designed framework consists of two analysis workflows with two different perspectives. In unsupervised analysis workflow, we examine the grouping of the complex substances by using their

chemical fingerprints derived from various analytical techniques, and quantitatively compare the grouping hierarchy to a reference categorization through F-M index. Whereas, in supervised analysis workflow, we benefit from the "read-across" hypothesis, that is similar complex substances that are grouped together based on their chemical structure (i.e. manufacturing category) are prone to behave similarly in terms of environmental health risk assessment. Hence, we can train highly accurate classification models by using the available information on categorization of known complex substances. The generated models can then be used to predict the environmental health impact of future unknown complex substances. The common highlight of both workflows is on the quantitative metrics, which immensely facilitates the comparative assessment of different parameters, such as distinct analytical techniques, data set sizes, or different number of categorization of samples to elucidate the optimal grouping of complex substances. Additionally, we incorporate the ToxPi representation of complex substances with the most informative analytical features to further deliver insights from the developed data-driven classification models.

Our results have shown that in order to assess the statistical significance of grouping results, it is highly important to permute category labels of complex substances and to calculate p-value for the obtained results regardless of the selected workflow. In addition, the dimensionality reduction plays a key role in reducing the noise in the extracted high-dimensional analytical chemistry profiles. Dimensionality reduction allows similar or higher grouping quality with significantly reduced number of measurements. The selection of the most informative features further improves data interpretation significantly through advanced data visualization techniques, such as the ToxPi representation. This further facilitates communication of the complex substance characteristics with regulatory decision-makers.

7. CONCLUSION AND FUTURE WORK

7.1 Conclusion

In this dissertation, we have developed novel data-driven frameworks by advancing big data analytics to solve challenging problems in process systems engineering. We have established the s-FDD framework for simultaneous fault detection and diagnosis and extended it to enable simultaneous fault identification and diagnosis. The framework is based on developing nonlinear SVM models while performing model-informed feature selection. The iterative model building enables to find the optimal model that yields the highest testing accuracy with the smallest set of informative features. As the dimensionality of the process data grows with the expedited data collection technologies, the importance of dimensionality reduction techniques during model building becomes more evident. Furthermore, the algorithm that the s-FDD framework follows enables simultaneous modeling and model-informed feature selection, where we are able to provide the explicit process variable measurements for fault diagnosis. While the most of the process monitoring methodologies use feature extraction techniques for dimensionality reduction. These analysis provides integrated features which obfuscate the interpretation of analysis (fault diagnosis in process monitoring). When the top n informative features are derived from the feature extraction based techniques, we actually need to acquire/collect information from $> n$ features. This is due to the fact that the selected n features are described as the linear combination of $> n$ features. The s-FDD framework and its extension to fault identification is presented in Chapter 2. By implementing hierarchical clustering analysis prior to the s-FDD framework, we have proposed a hierarchy-based classification scheme for fault identification. This is particularly important for industrial operations where one does not need to check/monitor the process with multiple models simultaneously, instead uses a hierarchy of models for identifying the problem. Chapter 3 and Chapter 4 demonstrate the application of the s-FDD framework on continuous and batch processes, respectively. Specifically, we have adopted the popular Tennessee Eastman process, which is a benchmark process used in numerous process monitoring applications in the literature, for continuous operation monitoring. The comparison between the s-FDD framework and the other available process monitoring algorithms/frameworks in the literature clearly proves the major advantage of using the s-FDD

framework, where one achieves highly accurate two-class *C*-SVM classifiers for fault detection. Moreover, we have utilized another benchmark process for batch process monitoring application, which is penicillin production via fed-batch process. Here, we have investigated three different time horizon approaches during model building: (i) one-step rolling time horizon basis analysis, (ii) two-step rolling time horizon basis analysis, and (iii) evolving time horizon analysis, in which we alter the amount of historical data usage during model training in the offline phase. Although the optimal time horizon approach is observed to be fault-specific, the two-step rolling time horizon based analysis is preferred, where we are able to include sufficient historical process data during the data-driven modeling without introducing noise.

The Industry 4.0 and Smart Manufacturing evolutions have raised the automation level in industry which subsequently reduced the number of personnel in manufacturing operations and increased the capital employed in production equipment and civil structures. Hence, adopting maintenance strategies during industrial operations have become prevalent and crucial for a reliable operation. Due to this prevalent use of maintenance strategies, we have examined an operation wherein maintenance strategies are adopted and the process is monitored by the s-FDD framework. Chapter 5 presents a proof-of-concept analysis, where the integrated use of corrective (i.e. "run-to-failure" reactive) and preventive maintenance strategies with the s-FDD framework can lower the process downtime significantly, and maximize the profit. Furthermore, a novel corrective maintenance strategy is formulated by integrating multi-parametric model predictive control and the s-FDD framework and presented the "parametric fault-tolerant control" concept in Chapter 5. Specifically, we have designed a fault-tolerant multi-parametric model predictive controller, and built accurate and robust fault detection and reconstruction mechanism for passing precise information of the process conditions to the controller during online monitoring. Similarly, we have benefited from the s-FDD framework to build the optimal data-driven models with high accuracy and low number of features. The parametric fault-tolerant control concept is tested on the fed-batch operation of penicillin production. The developed time-specific models have provided accurate process condition information (i.e. fault existence, fault type, magnitude and direction) to the mp-MPC, which accordingly enables the rapid changes in the control actions if necessary during the operation. The presented work in Section 5.3 in Chapter 5 is an active fault-tolerant control strategy that can provide an adaptive approach for smart operation in process systems engineering.

Finally, the advances in data-driven modeling analysis presented in previous chapters are utilized to tackle challenges in environmental health field under the Texas A&M Superfund Research Program. Chapter 6 introduces a framework to optimally group unknown complex substances, that are contaminated during environmental emergencies (i.e. hurricane, flooding etc.), with the groups of substances with known environmental health impact. This grouping technique with a known class of substances is known as read-across, where the chemical/biological effects of the unknown chemical mixtures/complex substances are examined through similarity analysis. This work uses the analytical chemistry data of complex substances, and provides an extensive comparative analysis between different data-driven techniques for optimal group identification during the environmental emergency-related contamination events.

Overall, the presented work in this dissertation provides accurate decision support tools in various areas of process systems engineering. As the data collection is significantly expedited in real time with the technological advancements, accurate data-driven decision making is becoming a major need in both in academy and industry. With the presented novel data-driven frameworks/tools in this dissertation, we aim to meet this need, where we enable accurate data-driven decision making in the areas of process monitoring, and environmental health analytics.

7.2 Key Contributions

The key contributions of the dissertation are summarized below.

1. A novel data-driven process monitoring framework (s-FDD framework), which achieves highly accurate fault detection/identification models with optimal process descriptors, has been developed and validated for both continuous and batch chemical operations. The results show that s-FDD framework outsmarts the existing statistical process monitoring techniques in the literature in terms of accurate fault detection rate. The selected optimal feature set guides the fault diagnosis, thus enables simultaneous fault detection and diagnosis. This further minimizes the process runtime spent under faulty condition, and ensures rapid process recovery for sustaining process safety and profitability (Chapter 2, Chapter 3, and Chapter 4).

2. A novel corrective maintenance optimization strategy is introduced by establishing parametric fault-tolerant control systems design framework. This is a novel active fault-tolerant

strategy that can eliminate the process downtime by enabling rapid switches between a priori mapped control actions. By merging multi-parametric model predictive control design and the s-FDD framework capabilities, the presented strategy proposes an adaptive approach for smart operation (Chapter 5).

3. A data-driven framework is established for grouping of unknown complex substances with the known chemicals during environmental emergency-related emergencies. The communication of the grouping results is significantly facilitated via quantitative metrics and visualization techniques. The presented work bridges the gap between optimal grouping of complex substances and the quantitative evaluation and communication of the grouping outcomes, which is essential for regulatory decision-making in the environmental health area (Chapter 6).

7.3 Future Work

By using the presented modeling and model-informed feature selection algorithm based on nonlinear SVMs, we can tackle numerous challenging problems in process systems engineering where high-dimensionality of the data set hinders accurate data-driven model generation. The two challenging and interesting problems in process monitoring, which can significantly benefit from the presented algorithm with advanced regression analysis, are online product quality monitoring and prediction of end-product quality. Under smart manufacturing initiatives, product quality monitoring has become a popular research area. In order to save operation cost and time, building an accurate and reliable tool for product quality prediction is of utmost importance. Moreover, we can adopt one-class SVM algorithms to build fault detection and diagnosis models. In this case, only normal operation data, which is ubiquitous in historical databases of industries rather than faulty operation data, is required to train the data-driven models. Finally, the proposed active fault tolerant control strategy based on multi-parametric programming and the s-FDD framework, which uses the nonlinear SVM based modeling and model-informed feature selection algorithm, can be further improved by incorporating fault identification and diagnosis model building. In this work, we have built fault-specific classifiers for fault detection within the presented parametric FTC strategy. This requires to monitor the ongoing chemical process for faults with distinct models. We have also presented the fault identification scheme, which performs hierarchy-based fault

detection via the s-FDD framework. Therefore, integration of the presented fault identification scheme within the presented parametric FTC design framework is proposed in order to increase the process efficiency by lowering the number of checks, thus improving the efficiency. This also improves the interpretation of the decision-support tool and renders it more user-friendly. Below, we provide brief details on how to perform the proposed tasks with the algorithms used within this work.

7.3.1 Improving the s-FDD Framework via Integrating One-class Classification Techniques

Often time, the industry has large amounts of historical process data, yet lacks the faulty operation data or the size of the faulty operation data is significantly smaller with respect to the normal operation data. Although, this problem can be achieved by simulating faulty operation data and use both of them to build two-class fault detection and diagnosis models via the s-FDD framework, performing additional simulations for faulty process data generation can be expensive in some industrial processes. In order to address this issue, one can adopt one-class classification techniques via SVM algorithms and perform simultaneous modeling and model-informed feature selection as described in Section 2.2 in Chapter 2 to build fault detection and diagnosis models. Specifically, one can train one-class SVM models to separate normal operation from any other faulty operation data. The model learns the range of the normal operation data and treats the any other process data as outliers (i.e. faulty data). The models generate binary answers, where $+1$ indicates the operation is normal, meaning that the online analyzed process data falls within the range of the previously learned normal operation data set. On the other hand, the model generates -1 to signify the operator that there is an ongoing abnormality. When an abnormality is detected, the optimal feature subset of the trained one-class SVM model is able to produce the diagnosis, so that operators can further focus on the selected process variables and act to recover the process. One-class based analysis techniques can be incorporated within the s-FDD framework to further improve its decision support where faulty operation data is not readily available and expensive or impossible to simulate.

7.3.2 Online Monitoring of the Product Quality and Predicting the Batch End-product Quality

Meeting the demand of high product quality is becoming challenging with increasing process structure complexity, which increases the number of controlled variables. Hence, development of data-driven tools for predicting online product and end-product quality is essential and necessary. Here, the idea is to quantify the quality of the product by analyzing the ongoing process data. Monitoring product quality with data-driven models requires the use of advanced regression analysis. However, building regression models without selecting an informative feature subset yields inaccurate results. As the size of the process data grows, building regression models with the most informative process descriptors has become crucial. Therefore, as a future work, we propose to build C or ν-parameterized Support Vector Regression (C- or ν-SVR) models with optimal feature subset by using the simultaneous modeling and feature selection algorithm based on nonlinear SVMs (Section 2.2 in Chapter 2). The procedure is similar to the fault-specific classifier modeling for fault detection and diagnosis via the s-FDD framework. The built regressors can then be used for monitoring the product quality online and predicting the end-product quality. Furthermore, the extracted features during simultaneous C- or ν-SVR modeling and model-informed feature selection would reveal the key informative process descriptors for tracking the product quality. Thus, if the monitored product quality approximates or falls below the pre-determined quality threshold, the selected optimal feature subset of the regressor would significantly facilitate the interpretation of the problem source.

7.3.3 Improving the Multi-parametric Fault-Tolerant Control Strategy with Fault Identification

In this dissertation, we have presented the application of the nonlinear SVM based simultaneous modeling and feature selection algorithm for building fault detection/identification and diagnosis models. We have established the s-FDD framework, where we build two-class fault-specific C-SVM classifiers for fault detection and diagnosis. Moreover, by formulating a hierarchy-based fault detection and diagnosis scheme, we extend the use of the s-FDD framework for fault identification. Additionally, we have developed the "parametric fault-tolerant control" strategies by integrating the multi-parametric model predictive controller design with the s-FDD framework.

In particular, we have used fault-specific classifiers for fault detection within the fault detection and reconstruction mechanism. Here, we propose to adopt fault identification scheme via s-FDD framework in order to improve the efficiency of fault detection among various faults. This requires the development of fault detection classifiers at each level of the separation hierarchy as described in Section 2.3.2 of Chapter 2. However, note that, this does not change the regressor development scheme, where we still need to train fault-specific regressors to estimate the fault magnitude. Furthermore, each additional fault needs to be considered as an additional parameter within the mp-MPC design. The advantage and premise of incorporating fault identification within the parametric FTC framework is rapid identification of the fault in a cascaded manner, where we do not check for each fault with separate models. This is especially important when the number of faults, considered during the parametric fault-tolerant system development, increase.

REFERENCES

[1] P. R. Lyman and C. Georgakis, "Plant-wide control of the tennessee eastman problem," *Computers & Chemical Engineering*, vol. 19, no. 3, pp. 321–331, 1995.

[2] W. F. de Carvalho Rocha, M. M. Schantz, D. A. Sheen, P. M. Chu, and K. A. Lippa, "Unsupervised classification of petroleum certified reference materials and other fuels by chemometric analysis of gas chromatography-mass spectrometry data," *Fuel*, vol. 197, pp. 248–258, 2017.

[3] S. Mahadevan and S. L. Shah, "Fault detection and diagnosis in process data using one-class support vector machines," *Journal of Process Control*, vol. 19, no. 10, pp. 1627–1639, 2009.

[4] Y. Xiao, H. Wang, W. Xu, and J. Zhou, "Robust one-class svm for fault detection," *Chemometrics and Intelligent Laboratory Systems*, vol. 151, pp. 15–25, 2016.

[5] S. Yin, S. X. Ding, X. Xie, and H. Luo, "A review on basic data-driven approaches for industrial process monitoring," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 11, pp. 6414–6428, 2014.

[6] M. S. Peters, K. D. Timmerhaus, R. E. West, K. Timmerhaus, and R. West, *Plant design and economics for chemical engineers*, vol. 4. McGraw-Hill New York, 1968.

[7] C. A. Floudas, A. M. Niziolek, O. Onel, and L. R. Matthews, "Multi-Scale Systems Engineering for Energy and the Environment: Challenges and Opportunities," *AIChE Journal*, vol. 62, no. 3, pp. 602–623, 2016.

[8] Q. P. He and J. Wang, "Statistical process monitoring as a big data analytics tool for smart manufacturing," *Journal of Process Control*, 2017.

[9] M. Helu, D. Libes, J. Lubell, K. Lyons, and K. Morris, "Enabling smart manufacturing technologies for decision-making support," *Proc ASME Des Eng Tech Conf*, vol. 1B, 2016.

[10] G.-H. Kim, S. Trimi, and J.-H. Chung, "Big data applications in the government sector," *Communications of the ACM*, vol. 57, no. 3, pp. 78–85, 2014.

[11] L. Chiang, B. Lu, and I. Castillo, "Big Data Analytics in Chemical Engineering," *Annual Review of Chemical and Biomolecular Engineering*, vol. 8, no. 1, pp. 63–85, 2017.

[12] M. Reis and G. Gins, "Industrial Process Monitoring in the Big Data/Industry 4.0 Era: from

Detection, to Diagnosis, to Prognosis," *Processes*, vol. 5, no. 3, p. 35, 2017.

[13] D. A. C. Beck, J. M. Carothers, V. R. Subramanian, and J. Pfaendtner, "Data Science: Accelerating Innovation and Discovery in Chemical Engineering," *AIChE Journal*, vol. 62, no. 5, pp. 1402–1416, 2016.

[14] F. Tao, Q. Qi, A. Liu, and A. Kusiak, "Data-driven smart manufacturing," *Journal of Manufacturing Systems*, vol. 48, pp. 157–169, 2018.

[15] S. Munirathinam and B. Ramadoss, "Big data predictive analtyics for proactive semiconductor equipment maintenance," in *2014 IEEE International Conference on Big Data (Big Data)*, pp. 893–902, Oct 2014.

[16] J. Krumeich, D. Werth, P. Loos, J. Schimmelpfennig, and S. Jacobi, "Advanced planning and control of manufacturing processes in steel industry through big data analytics: Case study and architecture proposal," in *2014 IEEE International Conference on Big Data (Big Data)*, pp. 16–24, IEEE, 2014.

[17] M. M. Papathanasiou, M. Onel, I. Nascu, and E. N. Pistikopoulos, "Computational tools in the assistance of personalized healthcare," in *Computer Aided Chemical Engineering*, vol. 42, pp. 139–206, Elsevier, 2018.

[18] K.-D. Thoben, S. Wiesner, and T. Wuest, "Industrie 4.0" and smart manufacturing–a review of research issues and application examples," *Int. J. Autom. Technol*, vol. 11, no. 1, 2017.

[19] T. F. Edgar and E. N. Pistikopoulos, "Smart manufacturing and energy systems," *Computers & Chemical Engineering*, 2017.

[20] L. H. Chiang, R. E. L., and B. R. D., *Fault Detection and Diagnosis in Industrial Systems*. Advanced Textbooks in Control and Signal Processing, Springer, 2001.

[21] L. H. Chiang, E. L. Russell, and R. D. Braatz, "Fault diagnosis in chemical processes using fisher discriminant analysis, discriminant partial least squares, and principal component analysis," *Chemometrics and Intelligent Laboratory Systems*, vol. 50, no. 2, pp. 243–252, 2000.

[22] V. Venkatasubramanian, R. Rengaswamy, K. Yin, and S. N. Kavuri, "A review of process fault detection and diagnosis," *Computers & Chemical Engineering*, vol. 27, no. 3, pp. 293–311, 2003.

[23] R. Isermann, "Model-based fault-detection and diagnosis – status and applications," *Annual*

Reviews in Control, vol. 29, no. 1, pp. 71 – 85, 2005.

[24] K. Takeda, B. Shibata, Y. Tsuge, and H. Matsuyama, "The improvement of fault diagnosis algorithm using signed directed graph," *IFAC Proceedings Volumes*, vol. 27, no. 5, pp. 351 – 356, 1994. IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes (SAFEPROCESS'94), Espoo, Finland, 13-16 June.

[25] Y.-K. Liu, G.-H. Wu, C.-L. Xie, Z.-Y. Duan, M.-J. Peng, and M.-K. Li, "A fault diagnosis method based on signed directed graph and matrix for nuclear power plants," *Nuclear Engineering and Design*, vol. 297, no. Supplement C, pp. 166 – 174, 2016.

[26] M. Chen, A. X. Zheng, J. Lloyd, M. I. Jordan, and E. Brewer, "Failure diagnosis using decision trees," in *International Conference on Autonomic Computing, 2004. Proceedings.*, pp. 36–43, May 2004.

[27] V. Venkatasubramanian, R. Vaidyanathan, and Y. Yamamoto, "Process fault detection and diagnosis using neural networks—i. steady-state processes," *Computers & Chemical Engineering*, vol. 14, no. 7, pp. 699 – 712, 1990.

[28] G. Silvestri, F. B. Verona, M. Innocenti, and M. Napolitano, "Fault detection using neural networks," in *Neural Networks, 1994. IEEE World Congress on Computational Intelligence., 1994 IEEE International Conference on*, vol. 6, pp. 3796–3799 vol.6, Jun 1994.

[29] Q. Zhao and Z. Xu, "Design of a novel knowledge-based fault detection and isolation scheme," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 34, pp. 1089–1095, April 2004.

[30] Z. Ge, Z. Song, and F. Gao, "Review of Recent Research on Data-Based Process Monitoring," *Industrial & Engineering Chemistry Research*, vol. 52, no. 10, pp. 3543–3562, 2013.

[31] V. Venkatasubramanian, R. Rengaswamy, and S. N. Kavuri, "A review of process fault detection and diagnosis part II: Qualitative models and search strategies," *Computers & Chemical Engineering*, vol. 27, no. 3, pp. 313–326, 2003.

[32] A. Narasingam and J. S.-I. Kwon, "Data-driven identification of interpretable reduced-order models using sparse regression," *Computers & Chemical Engineering*, vol. 119, pp. 101–111, 2018.

[33] K. Zhou, C. Fu, and S. Yang, "Big data driven smart energy management: From big data to big insights," *Renewable and Sustainable Energy Reviews*, vol. 56, pp. 215 – 225, 2016.

[34] M. M. Papathanasiou, M. Onel, I. Nascu, and E. N. Pistikopoulos, "Chapter 6 - computational tools in the assistance of personalized healthcare," in *Quantitative Systems Pharmacology* (D. Manca, ed.), vol. 42 of *Computer Aided Chemical Engineering*, pp. 139 – 206, Elsevier, 2018.

[35] M. Onel, B. Beykal, M. Wang, F. A. Grimm, L. Zhou, F. A. Wright, T. D. Phillips, I. Rusyn, and E. N. Pistikopoulos, "Optimal chemical grouping and sorbent material design by data analysis, modeling and dimensionality reduction techniques," in *28th European Symposium on Computer Aided Process Engineering* (A. Friedl, J. J. Klemeš, S. Radl, P. S. Varbanov, and T. Wallek, eds.), vol. 43 of *Computer Aided Chemical Engineering*, pp. 421 – 426, Elsevier, 2018.

[36] B. Beykal, F. Boukouvala, C. A. Floudas, N. Sorek, H. Zalavadia, and E. Gildin, "Global optimization of grey-box computational systems using surrogate functions and application to highly constrained oil-field operations," *Computers & Chemical Engineering*, vol. 114, pp. 99 – 110, 2018. FOCAPO/CPC 2017.

[37] D. Assouline, N. Mohajeri, and J.-L. Scartezzini, "Quantifying rooftop photovoltaic solar energy potential: A machine learning approach," *Solar Energy*, vol. 141, pp. 278 – 296, 2017.

[38] Y. Liu, Q. Liu, W. Wang, J. Zhao, and H. Leung, "Data-driven based model for flow prediction of steam system in steel industry," *Information Sciences*, vol. 193, pp. 104 – 114, 2012.

[39] M. Kano and Y. Nakagawa, "Data-based process monitoring, process control, and quality improvement: Recent developments and applications in steel industry," *Computers & Chemical Engineering*, vol. 32, no. 1, pp. 12 – 24, 2008. Process Systems Engineering: Contributions on the State-of-the-Art.

[40] S. J. Qin, "Survey on data-driven industrial process monitoring and diagnosis," *Annual Reviews in Control*, vol. 36, no. 2, pp. 220–234, 2012.

[41] B. Paya, I. Esat, and M. Badi, "Artificial neural network based fault diagnostics of rotating machinery using wavelet transforms as a preprocessor," *Mechanical Systems and Signal Processing*, vol. 11, no. 5, pp. 751–765, 1997.

[42] M. Bach-Andersen, B. Rømer-Odgaard, and O. Winther, "Deep learning for automated driv-

etrain fault detection," *Wind Energy*, vol. 21, no. 1, pp. 29–41, 2018.

[43] L. Breiman, *Classification and regression trees*. Routledge, 2017.

[44] Y. Zhao, L. Yang, B. Lehman, J.-F. de Palma, J. Mosesian, and R. Lyons, "Decision tree-based fault detection and classification in solar photovoltaic arrays," in *Applied power electronics conference and exposition (APEC), 2012 twenty-seventh annual IEEE*, pp. 93–99, IEEE, 2012.

[45] S. Yin, X. Gao, H. R. Karimi, and X. Zhu, "Study on support vector machine-based fault detection in tennessee eastman process," in *Abstract and Applied Analysis*, vol. 2014, Hindawi, 2014.

[46] Y. Xiao, H. Wang, W. Xu, and J. Zhou, "Robust one-class svm for fault detection," *Chemometrics and Intelligent Laboratory Systems*, vol. 151, pp. 15–25, 2016.

[47] Z. Yin and J. Hou, "Recent advances on svm based fault diagnosis and process monitoring in complicated industrial processes," *Neurocomputing*, vol. 174, pp. 643–650, 2016.

[48] D. L. de Souza, M. H. Granzotto, G. M. de Almeida, and L. C. Oliveira-Lopes, "Fault detection and diagnosis using support vector machines-a svc and svr comparison," *Journal of Safety Engineering*, vol. 3, no. 1, pp. 18–29, 2014.

[49] Y. Zhao, S. Wang, and F. Xiao, "Pattern recognition-based chillers fault detection method using support vector data description (svdd)," *Applied Energy*, vol. 112, pp. 1041–1048, 2013.

[50] F. A. P. Peres and F. S. Fogliatto, "Variable selection methods in multivariate statistical process control: A systematic literature review," *Computers & Industrial Engineering*, vol. 115, pp. 603 – 619, 2018.

[51] I. Jaffel, O. Taouali, M. F. Harkat, and H. Messaoud, "Kernel principal component analysis with reduced complexity for nonlinear dynamic process monitoring," *The International Journal of Advanced Manufacturing Technology*, vol. 88, pp. 3265–3279, Feb 2017.

[52] I. González and I. Sánchez, "Variable selection for multivariate statistical process control," *Journal of Quality Technology*, vol. 42, no. 3, pp. 242–259, 2010.

[53] C.-C. Hsu, M.-C. Chen, and L.-S. Chen, "Integrating independent component analysis and support vector machine for multivariate process monitoring," *Computers & Industrial Engineering*, vol. 59, no. 1, pp. 145 – 156, 2010.

[54] W. Jiang, K. Wang, and F. Tsung, "A variable-selection-based multivariate ewma chart for process monitoring and diagnosis," *Journal of Quality Technology*, vol. 44, no. 3, pp. 209–230, 2012.

[55] P. Nomikos and J. F. MacGregor, "Monitoring batch processes using multiway principal component analysis," *AIChE Journal*, vol. 40, no. 8, pp. 1361–1375, 1994.

[56] Y. Dong and S. J. Qin, "A novel dynamic pca algorithm for dynamic data modeling and process monitoring," *Journal of Process Control*, 2017.

[57] "A Systematic Methodology for Comparing Batch Process Monitoring Methods: Part I-Assessing Detection Strength," *Industrial and Engineering Chemistry Research*, vol. 55, no. 18, pp. 5342–5358, 2016.

[58] B. R. Bakshi, "Multiscale pca with application to multivariate statistical process monitoring," *AIChE Journal*, vol. 44, no. 7, pp. 1596–1610, 1998.

[59] F. Li, G. Church, M. Janakiram, H. Gholston, and G. Runger, "Fault detection for batch monitoring and discrete wavelet transforms," *Quality and Reliability Engineering International*, vol. 27, no. 8, pp. 999–1008, 2011.

[60] H. Yu, F. Khan, and V. Garaniya, "A sparse pca for nonlinear fault diagnosis and robust feature discovery of industrial processes," *AIChE Journal*, 2016.

[61] M. J. PlOVOSO and K. A. KOSANOVICH, "Applications of multivariate statistical methods to process monitoring and controller design," *International Journal of Control*, vol. 59, no. 3, pp. 743–765, 1994.

[62] P. Nomikos and J. F. MacGregor, "Multi-way partial least squares in monitoring batch processes," *Chemometrics and Intelligent Laboratory Systems*, vol. 30, no. 1, pp. 97 – 108, 1995. InCINC '94 Selected papers from the First International Chemometrics Internet Conference.

[63] U. Kruger and G. Dimitriadis, "Diagnosis of process faults in chemical systems using a local partial least squares approach," *AIChE Journal*, vol. 54, no. 10, pp. 2581–2596, 2008.

[64] G. Li, S. J. Qin, and D. Zhou, "Geometric properties of partial least squares for process monitoring," *Automatica*, vol. 46, no. 1, pp. 204 – 210, 2010.

[65] K. P. Detroja, R. D. Gudi, S. C. Patwardhan, and K. Roy, "Fault detection and isolation using correspondence analysis," *Industrial & Engineering Chemistry Research*, vol. 45, no. 1,

pp. 223–235, 2006.

[66] K. P. Detroja, R. D. Gudi, and S. C. Patwardhan, "Data reduction and fault diagnosis using principle of distributional equivalence," in *2011 International Symposium on Advanced Control of Industrial Processes (ADCONIP)*, pp. 30–35, May 2011.

[67] J.-M. Lee, C. Yoo, S. W. Choi, P. A. Vanrolleghem, and I.-B. Lee, "Nonlinear process monitoring using kernel principal component analysis," *Chemical Engineering Science*, vol. 59, no. 1, pp. 223–234, 2004.

[68] Y. Zhang, H. Zhou, S. J. Qin, and T. Chai, "Decentralized fault diagnosis of large-scale processes using multiblock kernel partial least squares," *IEEE Transactions on Industrial Informatics*, vol. 6, no. 1, pp. 3–10, 2010.

[69] E. L. Russell, L. H. Chiang, and R. D. Braatz, "Fault detection in industrial processes using canonical variate analysis and dynamic principal component analysis," *Chemometrics and Intelligent Laboratory Systems*, vol. 51, no. 1, pp. 81–93, 2000.

[70] G. Lee, C. Han, and E. S. Yoon, "Multiple-fault diagnosis of the tennessee eastman process based on system decomposition and dynamic pls," *Industrial & Engineering Chemistry Research*, vol. 43, no. 25, pp. 8037–8048, 2004.

[71] M. Kano, S. Tanaka, S. Hasebe, I. Hashimoto, and H. Ohno, "Monitoring independent components for fault detection," *AIChE Journal*, vol. 49, no. 4, pp. 969–976, 2003.

[72] D. Li, Y. Zhou, G. Hu, and C. J. Spanos, "Optimal sensor configuration and feature selection for ahu fault detection and diagnosis," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 3, pp. 1369–1380, 2017.

[73] Y. Li, Y. Yang, G. Li, M. Xu, and W. Huang, "A fault diagnosis scheme for planetary gearboxes using modified multi-scale symbolic dynamic entropy and mrmr feature selection," *Mechanical Systems and Signal Processing*, vol. 91, pp. 295–312, 2017.

[74] B. Chebel-Morello, S. Malinowski, and H. Senoussi, "Feature selection for fault detection systems: application to the tennessee eastman process," *Applied Intelligence*, vol. 44, no. 1, pp. 111–122, 2016.

[75] K. Yan, L. Ma, Y. Dai, W. Shen, Z. Ji, and D. Xie, "Cost-sensitive and sequential feature selection for chiller fault detection and diagnosis," *International Journal of Refrigeration*, vol. 86, pp. 401–409, 2018.

[76] Z. Wei, Y. Wang, S. He, and J. Bao, "A novel intelligent method for bearing fault diagnosis based on affinity propagation clustering and adaptive feature selection," *Knowledge-Based Systems*, vol. 116, pp. 1–12, 2017.

[77] C. Xia and J. Howell, "Isolating multiple sources of plant-wide oscillations via independent component analysis," *Control Engineering Practice*, vol. 13, no. 8, pp. 1027–1035, 2005.

[78] F. Yang and D. Xiao, "Progress in root cause and fault propagation analysis of large-scale industrial processes," *Journal of Control Science and Engineering*, vol. 2012, 2012.

[79] M. Bauer, J. W. Cox, M. H. Caveness, J. J. Downs, and N. F. Thornhill, "Finding the direction of disturbance propagation in a chemical process using transfer entropy," *IEEE Transactions on Control Systems Technology*, vol. 15, no. 1, pp. 12–21, 2007.

[80] C. Xia, J. Howell, and N. F. Thornhill, "Detecting and isolating multiple plant-wide oscillations via spectral independent component analysis," *Automatica*, vol. 41, no. 12, pp. 2067–2075, 2005.

[81] X. Zang and J. Howell, "Isolating the root cause of propagated oscillations in process plants," *International Journal of Adaptive Control and Signal Processing*, vol. 19, no. 4, pp. 247–265, 2005.

[82] J. Liu and D.-S. Chen, "Fault isolation using modified contribution plots," *Computers & Chemical Engineering*, vol. 61, pp. 9–19, 2014.

[83] J. Yu and M. M. Rashid, "A novel dynamic bayesian network-based networked process monitoring approach for fault detection, propagation identification, and root cause diagnosis," *AIChE Journal*, vol. 59, no. 7, pp. 2348–2365, 2013.

[84] T. Yuan and S. J. Qin, "Root cause diagnosis of plant-wide oscillations using granger causality," *Journal of Process Control*, vol. 24, no. 2, pp. 450–459, 2014.

[85] A. H. Tsang, A. K. Jardine, and H. Kolodny, "Measuring maintenance performance: a holistic approach," *International Journal of Operations & Production Management*, vol. 19, no. 7, pp. 691–715, 1999.

[86] A. Kelly, *Maintenance strategy*. Elsevier, 1997.

[87] C. Vassiliadis and E. Pistikopoulos, "Maintenance scheduling and process optimization under uncertainty," *Computers & Chemical Engineering*, vol. 25, no. 2-3, pp. 217–236, 2001.

[88] C. Vassiliadis and E. Pistikopoulos, "Maintenance-based strategies for environmental risk

minimization in the process industries," *Journal of Hazardous Materials*, vol. 71, no. 1-3, pp. 481–501, 2000.

[89] J. Davis, T. Edgar, J. Porter, J. Bernaden, and M. Sarli, "Smart manufacturing, manufacturing intelligence and demand-dynamic performance," *Computers & Chemical Engineering*, vol. 47, pp. 145–156, 2012.

[90] R. Dekker, "Applications of maintenance optimization models: a review and analysis," *Reliability Engineering & System Safety*, vol. 51, no. 3, pp. 229–240, 1996.

[91] P. O'Donovan, K. Leahy, K. Bruton, and D. T. O'Sullivan, "An industrial big data pipeline for data-driven analytics maintenance applications in large-scale smart manufacturing facilities," *Journal of Big Data*, vol. 2, no. 1, p. 25, 2015.

[92] Z. Tian and H. Liao, "Condition based maintenance optimization for multi-component systems using proportional hazards model," *Reliability Engineering & System Safety*, vol. 96, no. 5, pp. 581–589, 2011.

[93] F. I. Khan and M. M. Haddara, "Risk-based maintenance (rbm): a quantitative approach for maintenance/inspection scheduling and planning," *Journal of Loss Prevention in the Process Industries*, vol. 16, no. 6, pp. 561–573, 2003.

[94] G. A. Susto, A. Schirru, S. Pampuri, S. McLoone, and A. Beghi, "Machine learning for predictive maintenance: A multiple classifier approach," *IEEE Transactions on Industrial Informatics*, vol. 11, no. 3, pp. 812–820, 2015.

[95] T. Nakagawa, *Maintenance theory of reliability*. Springer Science & Business Media, 2006.

[96] K. Arif-Uz-Zaman, M. E. Cholette, L. Ma, and A. Karim, "Extracting failure time data from industrial maintenance records using text mining," *Advanced Engineering Informatics*, vol. 33, pp. 388–396, 2017.

[97] R. Karim, J. Westerberg, D. Galar, and U. Kumar, "Maintenance analytics–the new know in maintenance," *IFAC-PapersOnLine*, vol. 49, no. 28, pp. 214–219, 2016.

[98] J. Gardner, D. Koutra, J. Mroueh, V. Pang, A. Farahi, S. Krassenstein, and J. Webb, "Driving with data: Modeling and forecasting vehicle fleet maintenance in detroit," *arXiv preprint arXiv:1710.06839*, 2017.

[99] M. Fernandes, A. Canito, V. Bolón-Canedo, L. Conceição, I. Praça, and G. Marreiros, "Data analysis and feature selection for predictive maintenance: A case-study in the metallurgic

industry," *International Journal of Information Management*, vol. 46, pp. 252–262, 2019.

[100] V. Ravi and R. Patil, "Unsupervised time series data analysis for error pattern extraction for predictive maintenance," in *International Conference on Advances in Computing and Data Sciences*, pp. 1–10, Springer, 2018.

[101] A. Kusiak, "Fundamentals of smart manufacturing: A multi-thread perspective," *Annual Reviews in Control*, 2019.

[102] H. Alwi, C. Edwards, and C. P. Tan, *Fault detection and fault-tolerant control using sliding modes*. Springer Science & Business Media, 2011.

[103] M. Blanke, M. Staroswiecki, and N. E. Wu, "Concepts and methods in fault-tolerant control," in *Proceedings of the 2001 American Control Conference.(Cat. No. 01CH37148)*, vol. 4, pp. 2606–2620, IEEE, 2001.

[104] R. J. Patton, "Fault-tolerant control: the 1997 situation," *IFAC Proceedings Volumes*, vol. 30, no. 18, pp. 1029–1051, 1997.

[105] J. Jiang and X. Yu, "Fault-tolerant control systems: A comparative study between active and passive approaches," *Annual Reviews in Control*, vol. 36, no. 1, pp. 60–72, 2012.

[106] J. McMahan, "Flight 1080," *Air Line Pilot*, vol. 47, no. 7, pp. 6–11, 1978.

[107] C. Edwards, T. Lombaerts, H. Smaili, *et al.*, "Fault tolerant flight control," *Lecture Notes in Control and Information Sciences*, vol. 399, pp. 1–560, 2010.

[108] D. M. Himmelblau, *Fault detection and diagnosis in chemical and petrochemical processes*, vol. 8. Elsevier Science Ltd, 1978.

[109] Q. Zhao and J. Jiang, "Reliable state feedback control system design against actuator failures," *Automatica*, vol. 34, no. 10, pp. 1267–1272, 1998.

[110] J. Jiang, "Fault-tolerant control systems—an introductory overview," *Automatica SINCA*, vol. 21, no. 1, pp. 161–174, 2005.

[111] Y. Zhang and J. Jiang, "Bibliographical review on reconfigurable fault-tolerant control systems," *Annual Reviews in Control*, vol. 32, no. 2, pp. 229–252, 2008.

[112] W. Wong and L. W. Lee, "Fault-resilient automobile control system," Sept. 28 1999. US Patent 5,957,985.

[113] Y.-s. Jeong, S.-K. Sul, S. E. Schulz, and N. R. Patel, "Fault detection and fault-tolerant control of interior permanent-magnet motor drive system for electric vehicle," *IEEE Trans-*

actions on Industry Applications, vol. 41, no. 1, pp. 46–51, 2005.

[114] H. Noura, D. Theilliol, J.-C. Ponsart, and A. Chamseddine, *Fault-tolerant control systems: Design and practical applications*. Springer Science & Business Media, 2009.

[115] P. D. Christofides, J. F. Davis, N. H. El-Farra, D. Clark, K. R. Harris, and J. N. Gipson, "Smart plant operations: Vision, progress and challenges," *AIChE Journal*, vol. 53, no. 11, pp. 2734–2741, 2007.

[116] J. Davis, T. Edgar, R. Graybill, P. Korambath, B. Schott, D. Swink, J. Wang, and J. Wetzel, "Smart manufacturing," *Annual Review of Chemical and Biomolecular Engineering*, vol. 6, pp. 141–160, 2015.

[117] B. Yu and Y. Zhang, "Fault-tolerant control of a boeing 747-100/200 based on a laguerre function-based mpc scheme," *IFAC-PapersOnLine*, vol. 49, no. 17, pp. 58 – 63, 2016. 20th IFAC Symposium on Automatic Control in AerospaceACA 2016.

[118] R. Hallouzi and M. Verhaegen, "Reconfigurable fault tolerant control of a boeing 747 using subspace predictive control," in *AIAA Guidance, Navigation and Control Conference and Exhibit*, p. 6665, 2007.

[119] F. Caccavale and L. Villani, *Fault diagnosis and fault tolerance for mechatronic systems: Recent advances*, vol. 1. Springer Science & Business Media, 2002.

[120] L. Ye, S. Wang, F. Bing, O. Malik, and Y. Zeng, "Control/maintenance strategy fault tolerant mode and reliability analysis for hydro power stations," *IEEE Transactions on Power Systems*, vol. 16, no. 3, pp. 340–345, 2001.

[121] H. Noura, D. Sauter, F. Hamelin, and D. Theilliol, "Fault-tolerant control in dynamic systems: Application to a winding machine," *IEEE Control Systems Magazine*, vol. 20, no. 1, pp. 33–49, 2000.

[122] F. Pirmoradi, F. Sassani, and C. De Silva, "Fault detection and diagnosis in a spacecraft attitude determination system," *Acta Astronautica*, vol. 65, no. 5-6, pp. 710–729, 2009.

[123] B. Xiao, Q. Hu, and Y. Zhang, "Adaptive sliding mode fault tolerant attitude tracking control for flexible spacecraft under actuator saturation," *IEEE Transactions on Control Systems Technology*, vol. 20, no. 6, pp. 1605–1612, 2012.

[124] Y. Jiang, Q. Hu, and G. Ma, "Adaptive backstepping fault-tolerant control for flexible spacecraft with unknown bounded disturbances and actuator failures," *ISA Transactions*, vol. 49,

no. 1, pp. 57 – 69, 2010.

[125] S. Yin, B. Xiao, S. X. Ding, and D. Zhou, "A review on recent development of spacecraft attitude fault tolerant control system," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 5, pp. 3311–3320, 2016.

[126] E. Eryurek and B. R. Upadhyaya, "Fault-tolerant control and diagnostics for large-scale systems," *IEEE Control Systems Magazine*, vol. 15, no. 5, pp. 34–42, 1995.

[127] Z. Deng, X. Shi, G. Xia, and M. Fu, "Fault tolerant control for steam generators in nuclear power plant," *Nuclear Power Engineering*, vol. 31, no. 1, pp. 107–111, 2010.

[128] Y. Guzman, *Theoretical Advances in Robust Optimization, Feature Selection, and Biomarker Discovery*. PhD thesis, Princeton University, Princeton, NJ, 2016.

[129] J. A. Horney, G. A. Casillas, E. Baker, K. W. Stone, K. R. Kirsch, K. Camargo, T. L. Wade, and T. J. McDonald, "Comparing residential contamination in a houston environmental justice neighborhood before and after hurricane harvey," *PloS One*, vol. 13, no. 2, p. e0192660, 2018.

[130] D. D. Reible, C. N. Haas, J. H. Pardue, and W. J. Walsh, "Toxic and contaminant concerns generated by hurricane katrina," vol. 132, no. 6, 2006.

[131] H. W. Mielke, C. R. Gonzales, E. T. Powell, and P. W. Mielke, "Environmental and health disparities in residential communities of new orleans: The need for soil lead intervention to advance primary prevention," *Environment International*, vol. 51, pp. 73–81, 2013.

[132] H. W. Mielke, "Environmental lead after hurricane katrina," *Environmental Health Perspectives*, vol. 120, no. 5, p. a188, 2012.

[133] J. Chou, D. Elbers, G. Clement, B. Bursavich, T. Tian, W. Zhang, and K. Yang, "In situ monitoring (field screening) and assessment of lead and arsenic contaminants in the greater new orleans area using a portable x-ray fluorescence analyser," *Journal of Environmental Monitoring*, vol. 12, no. 9, pp. 1722–1729, 2010.

[134] A. H. Knap and I. Rusyn, "Environmental exposures due to natural disasters," *Reviews on Environmental Health*, vol. 31, no. 1, pp. 89–92, 2016.

[135] S. Vardoulakis, C. Dimitroulopoulou, J. Thornes, K.-M. Lai, J. Taylor, I. Myers, C. Heaviside, A. Mavrogianni, C. Shrubsole, Z. Chalabi, *et al.*, "Impact of climate change on the domestic indoor environment and associated health risks in the uk," *Environment Interna-*

tional, vol. 85, pp. 299–313, 2015.

[136] P. W. Elbers, A. Girbes, M. L. Malbrain, and R. Bosman, "Right dose, right now: using big data to optimize antibiotic dosing in the critically ill," *Anaesthesiology Intensive Therapy*, vol. 47, no. 5, pp. 457–463, 2015.

[137] M. H. Berlian, T. E. R. Sahputra, B. J. W. Ardi, L. W. Dzatmika, A. R. A. Besari, R. W. Sudibyo, and S. Sukaridhoto, "Design and implementation of smart environment monitoring and analytics in real-time system framework based on internet of underwater things and big data," in *Electronics Symposium (IES), 2016 International*, pp. 403–408, IEEE, 2016.

[138] I. V. Tetko, O. Engkvist, U. Koch, J.-L. Reymond, and H. Chen, "Bigchem: Challenges and opportunities for big data analysis in chemistry," *Molecular Informatics*, vol. 35, no. 11-12, pp. 615–621, 2016.

[139] S. D. Dimitrov, D. G. Georgieva, T. S. Pavlov, Y. H. Karakolev, P. G. Karamertzanis, M. Rasenberg, and O. G. Mekenyan, "Uvcb substances: methodology for structural description and application to fate and hazard assessment," *Environmental Toxicology and Chemistry*, vol. 34, no. 11, pp. 2450–2462, 2015.

[140] E. C. Agency, "Read-across assessment framework (raaf) - considerations on multi-constituent substances and uvcbs," 2017.

[141] C. R. Clark, R. H. McKee, J. J. Freeman, D. Swick, S. Mahagaokar, G. Pigram, L. G. Roberts, C. J. Smulders, and P. W. Beatty, "A ghs-consistent approach to health hazard classification of petroleum substances, a class of uvcb substances," *Regulatory Toxicology and Pharmacology*, vol. 67, no. 3, pp. 409–420, 2013.

[142] M. Bell and J. M. Blais, ""-omics" workflow for paleolimnological and geological archives: A review," *Science of The Total Environment*, 2019.

[143] Y. Cho, A. Ahmed, A. Islam, and S. Kim, "Developments in ft-icr ms instrumentation, ionization techniques, and data interpretation methods for petroleomics," *Mass Spectrometry Reviews*, vol. 34, no. 2, pp. 248–263, 2015.

[144] C. REACH, "Analytical characterisation of petroleum uvcb substances," *Brussels, Belgium*, 2012.

[145] N. R. Catlin, B. J. Collins, S. S. Auerbach, S. S. Ferguson, J. M. Harnly, C. Gennings, S. Waidyanatha, G. E. Rice, S. L. Smith-Roe, K. L. Witt, *et al.*, "How similar is similar

enough? a sufficient similarity case study with ginkgo biloba extract.," *Food and Chemical Toxicology: an international journal published for the British Industrial Biological Research Association*, vol. 118, p. 328, 2018.

[146] F. A. Grimm, W. K. Russell, Y.-S. Luo, Y. Iwata, W. A. Chiu, T. Roy, P. J. Boogaard, H. B. Ketelslegers, and I. Rusyn, "Grouping of petroleum substances as example uvcbs by ion mobility-mass spectrometry to enable chemical composition-based read-across," *Environmental Science & Technology*, vol. 51, no. 12, pp. 7197–7207, 2017.

[147] A. G. Marshall and R. P. Rodgers, "Petroleomics: Chemistry of the underworld," *Proceedings of the National Academy of Sciences*, vol. 105, no. 47, pp. 18090–18095, 2008.

[148] R. W. Rozett and E. M. Petersen, "Methods of factor analysis of mass spectra," *Analytical Chemistry*, vol. 47, no. 8, pp. 1301–1308, 1975.

[149] A. Flexer, "Limitations of self-organizing maps for vector quantization and multidimensional scaling," in *Advances in neural information processing systems*, pp. 445–451, 1997.

[150] H. Yin, "Connection between self-organizing maps and metric multidimensional scaling," in *2007 International Joint Conference on Neural Networks*, pp. 1025–1030, IEEE, 2007.

[151] J. Rank, "Classification and risk assessment of chemicals: the case of dehp in the light of reach," *The Journal of Transdisciplinary Environmental Studies*, vol. 4, no. 3, pp. 1–15, 2005.

[152] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[153] V. N. Vapnik, *The Nature of Statistical Learning Theory*. New York, NY, USA: Springer-Verlag New York, 1995.

[154] B. Scholkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, MA, USA: MIT Press, 2001.

[155] J. Sobieszczanski-Sobieski, J.-F. Barthelemyy, and K. M. Riley, "Sensitivity of optimum solutions of problem parameters," *AIAA Journal*, vol. 20, no. 9, pp. 1291–1299, 1982.

[156] J.-F. Barthelemy and J. Sobieszczanski-Sobieski, "Optimum sensitivity derivatives of objective functions in nonlinear programming," *AIAA Journal*, vol. 21, no. 6, pp. 913–915, 1983.

[157] E. Castillo, R. Mínguez, and C. Castillo, "Sensitivity analysis in optimization and reliability

problems," *Reliability Engineering & System Safety*, vol. 93, no. 12, pp. 1788–1800, 2008.

[158] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene selection for cancer classification using support vector machines," *Machine Learning*, vol. 46, no. 1, pp. 389–422, 2002.

[159] C.-C. Chang and C.-J. Lin, "Libsvm: a library for support vector machines," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 3, p. 27, 2011.

[160] C. A. Kieslich, P. Tamamis, Y. A. Guzman, M. Onel, and C. A. Floudas, "Highly accurate structure-based prediction of hiv-1 coreceptor usage suggests intermolecular interactions driving tropism," *PloS One*, vol. 11, no. 2, p. e0148974, 2016.

[161] C. Keasar, L. J. McGuffin, B. Wallner, G. Chopra, B. Adhikari, D. Bhattacharya, L. Blake, L. O. Bortot, R. Cao, B. Dhanasekaran, *et al.*, "An analysis and evaluation of the wefold collaborative for protein structure prediction and its pipelines in casp11 and casp12," *Scientific Reports*, vol. 8, no. 1, p. 9939, 2018.

[162] P. Nomikos and J. F. MacGregor, "Monitoring batch processes using multiway principal component analysis," *AIChE Journal*, vol. 40, no. 8, pp. 1361–1375, 1994.

[163] A. N. Baraldi and C. K. Enders, "An introduction to modern missing data analyses," *Journal of School Psychology*, vol. 48, no. 1, pp. 5–37, 2010.

[164] P. H. Rezvan, K. J. Lee, and J. A. Simpson, "The rise of multiple imputation: a review of the reporting and implementation of the method in medical research," *BMC Medical Research Methodology*, vol. 15, no. 1, p. 30, 2015.

[165] T. D. Pigott, "A review of methods for missing data," *Educational Research and Evaluation*, vol. 7, no. 4, pp. 353–383, 2001.

[166] J. Y. Lee and M. P. Styczynski, "Ns-knn: a modified k-nearest neighbors approach for imputing metabolomics data," *Metabolomics*, vol. 14, no. 12, p. 153, 2018.

[167] J. Han, J. Pei, and M. Kamber, *Data mining: concepts and techniques*. Elsevier, 2011.

[168] B. Lu, S. Xu, J. Stuber, and T. F. Edgar, "Constrained selective dynamic time warping of trajectories in three dimensional batch data," *Chemometrics and Intelligent Laboratory Systems*, vol. 159, pp. 138–150, 2016.

[169] Y. Zhang, B. Lu, and T. F. Edgar, "Batch trajectory synchronization with robust derivative dynamic time warping," *Industrial & Engineering Chemistry Research*, vol. 52, no. 35, pp. 12319–12328, 2013.

[170] J. M. González-Martínez, A. Ferrer, and J. A. Westerhuis, "Real-time synchronization of batch trajectories for on-line multivariate statistical process control using dynamic time warping," *Chemometrics and Intelligent Laboratory Systems*, vol. 105, no. 2, pp. 195–206, 2011.

[171] C. Zhao, S. Mo, F. Gao, N. Lu, and Y. Yao, "Statistical analysis and online monitoring for handling multiphase batch processes with varying durations," *Journal of Process Control*, vol. 21, no. 6, pp. 817–829, 2011.

[172] M. Fransson and S. Folestad, "Real-time alignment of batch process data using cow for on-line process monitoring," *Chemometrics and Intelligent Laboratory Systems*, vol. 84, no. 1-2, pp. 56–61, 2006.

[173] C. Ündey, S. Ertunç, and A. Çınar, "Online batch/fed-batch process performance monitoring, quality prediction, and variable-contribution analysis for diagnosis," *Industrial & Engineering Chemistry Research*, vol. 42, no. 20, pp. 4645–4658, 2003.

[174] C. Ündey, B. A. Williams, and A. Cınar, "Monitoring of batch pharmaceutical fermentations: Data synchronization, landmark alignment, and real-time monitoring," *IFAC Proceedings Volumes*, vol. 35, no. 1, pp. 271–276, 2002.

[175] J. H. Ward Jr, "Hierarchical grouping to optimize an objective function," *Journal of the American Statistical Association*, vol. 58, no. 301, pp. 236–244, 1963.

[176] J. J. Downs and E. F. Vogel, "A plant-wide industrial process control problem," *Computers & Chemical Engineering*, vol. 17, no. 3, pp. 245–255, 1993.

[177] C. A. Rieth, B. D. Amsel, R. Tran, and M. B. Cook, "Additional tennessee eastman process simulation data for anomaly detection evaluation," 2017.

[178] P. Domingos, "The role of occam's razor in knowledge discovery," *Data Mining and Knowledge Discovery*, vol. 3, no. 4, pp. 409–425, 1999.

[179] G. Birol, C. Ündey, and A. Çinar, "A modular simulation package for fed-batch fermentation: Penicillin production," *Computers & Chemical Engineering*, vol. 26, no. 11, pp. 1553–1565, 2002.

[180] J. Van Impe and G. Gins, "An extensive reference dataset for fault detection and identification in batch processes," *Chemometrics and Intelligent Laboratory Systems*, vol. 148, pp. 20–31, 2015.

[181] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos, "The explicit linear quadratic regulator for constrained systems," *Automatica*, vol. 38, no. 1, pp. 3–20, 2002.

[182] E. N. Pistikopoulos, N. A. Diangelakis, R. Oberdieck, M. M. Papathanasiou, I. Nascu, and M. Sun, "Paroc—an integrated framework and software platform for the optimisation and advanced model-based control of process systems," *Chemical Engineering Science*, vol. 136, pp. 115 – 138, 2015. Control and Optimization of Smart Plant Operations.

[183] G. Gins, J. Vanlaer, P. Van den Kerkhof, and J. F. Van Impe, "The raymond simulation package—generating raypresentative monitoring data to design advanced process monitoring and control algorithms," *Computers & Chemical Engineering*, vol. 69, pp. 108–118, 2014.

[184] N. A. Diangelakis, B. Burnak, J. Katz, and E. N. Pistikopoulos, "Process design and control optimization: A simultaneous approach by multi-parametric programming," *AIChE Journal*, vol. 63, no. 11, pp. 4827–4846, 2017.

[185] G. S. Ogumerem and E. N. Pistikopoulos, "Parametric optimization and control toward the design of a smart metal hydride refueling system," *AIChE Journal*, vol. 0, no. 0, p. e16680.

[186] M. M. Papathanasiou, B. Burnak, J. Katz, N. Shah, and E. N. Pistikopoulos, "Assisting continuous biomanufacturing through advanced control in downstream purification," *Computers & Chemical Engineering*, vol. 125, pp. 232 – 248, 2019.

[187] B. Burnak, J. Katz, N. A. Diangelakis, and E. N. Pistikopoulos, "Simultaneous process scheduling and control: A multiparametric programming-based approach," *Industrial & Engineering Chemistry Research*, vol. 57, no. 11, pp. 3963–3976, 2018.

[188] B. Burnak, N. A. Diangelakis, J. Katz, and E. N. Pistikopoulos, "Integrated process design, scheduling, and control using multiparametric programming," *Computers & Chemical Engineering*, vol. 125, pp. 164 – 184, 2019.

[189] J. Katz, B. Burnak, and E. N. Pistikopoulos, "The impact of model approximation in multiparametric model predictive control," *Chemical Engineering Research and Design*, vol. 139, pp. 211 – 223, 2018.

[190] R. Oberdieck, N. A. Diangelakis, M. M. Papathanasiou, I. Nascu, and E. N. Pistikopoulos, "Pop – parametric optimization toolbox," *Industrial & Engineering Chemistry Research*, vol. 55, no. 33, pp. 8979–8991, 2016.

[191] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[192] M. Onel, C. A. Kieslich, Y. A. Guzman, C. A. Floudas, and E. N. Pistikopoulos, "Big data approach to batch process monitoring:simultaneous fault detection and diagnosis using nonlinear support vector machine-based feature selection," *Computers & Chemical Engineering*, 2018.

[193] M. Onel, C. A. Kieslich, and E. N. Pistikopoulos, "A nonlinear support vector machine-based feature selection approach for fault detection and diagnosis: Application to the tennessee eastman process," *AIChE Journal*, vol. 65, no. 3, pp. 992–1005, 2019.

[194] K. Ferguson, "Characterization of Complex Substances Used in Biological Profiling Through Determination of the Free Concentration Within In Vitro Assays," Master's thesis, Texas A&M University, College Station, TX, 2018.

[195] "PetroOrg Software." `http://software.Petroorg.com`, 2014.

[196] K. T. Do, S. Wahl, J. Raffler, S. Molnos, M. Laimighofer, J. Adamski, K. Suhre, K. Strauch, A. Peters, C. Gieger, *et al.*, "Characterization of missing values in untargeted ms-based metabolomics data and evaluation of missing data handling strategies," *Metabolomics*, vol. 14, no. 10, p. 128, 2018.

[197] E. Alpaydin, *Introduction to machine learning*. MIT press, 2014.

[198] T. Kohonen, "The self-organizing map," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464–1480, 1990.

[199] M. Onel, B. Beykal, K. Ferguson, W. A. Chiu, T. J. McDonald, L. Zhou, J. S. House, F. A. Wright, D. A. Sheen, I. Rusyn, and E. N. Pistikopoulos, "Grouping of Complex Substances Using Analytical Chemistry Data: A Framework for Quantitative Evaluation and Visualization." `http://parametric.tamu.edu/research/Onel_etAl_2019_Rmarkdown.html`, 2019. [Online; accessed 26-May-2019].

[200] M. Mukaka, "Statistics corner: A guide to appropriate use of correlation coefficient in medical research," *Malawi Medical Journal*, vol. 24, no. 3, pp. 69–71, 2012.

[201] A. Blum, J. Hopcroft, and R. Kannan, "Foundations of data science," *Vorabversion Eines Lehrbuchs*, 2016.

[202] B. R. Donald, *Algorithms in structural molecular biology*. MIT Press, 2011.

[203] E. B. Fowlkes and C. L. Mallows, "A method for comparing two hierarchical clusterings,"

Journal of the American Statistical Association, vol. 78, no. 383, pp. 553–569, 1983.

[204] S. Wagner and D. Wagner, *Comparing clusterings: an overview*. Universität Karlsruhe, Fakultät für Informatik Karlsruhe, 2007.

[205] B. Beykal, F. Boukouvala, C. A. Floudas, and E. N. Pistikopoulos, "Optimal design of energy systems using constrained grey-box multi-objective optimization," *Computers & Chemical Engineering*, vol. 116, pp. 488–502, 2018.

[206] D. W. Hosmer Jr, S. Lemeshow, and R. X. Sturdivant, *Applied logistic regression*, vol. 398. John Wiley & Sons, 2013.

[207] S. W. Marvel, K. To, F. A. Grimm, F. A. Wright, I. Rusyn, and D. M. Reif, "Toxpi graphical user interface 2.0: Dynamic exploration, visualization, and sharing of integrated data models," *BMC Bioinformatics*, vol. 19, no. 1, p. 80, 2018.

[208] D. M. Reif, M. T. Martin, S. W. Tan, K. A. Houck, R. S. Judson, A. M. Richard, T. B. Knudsen, D. J. Dix, and R. J. Kavlock, "Endocrine profiling and prioritization of environmental chemicals using toxcast data," *Environmental Health Perspectives*, vol. 118, no. 12, pp. 1714–1720, 2010.

[209] D. M. Reif, M. Sypa, E. F. Lock, F. A. Wright, A. Wilson, T. Cathey, R. R. Judson, and I. Rusyn, "Toxpi gui: an interactive visualization tool for transparent integration of data from diverse sources of evidence," *Bioinformatics*, vol. 29, no. 3, pp. 402–403, 2012.

[210] S. A. Stout and Z. Wang, "Chemical fingerprinting methods and factors affecting petroleum fingerprints in the environment," in *Standard Handbook Oil Spill Environmental Forensics*, pp. 61–129, Elsevier, 2016.

APPENDIX A

APPLICATION OF THE S-FDD FRAMEWORK FOR CONTINUOUS PROCESS MONITORING

A.1 Tennessee Eastman Process Reactions and Process Variables

The reactions occurring in the reactor are as follows:

$$\begin{aligned} A_{(g)} + C_{(g)} + D_{(g)} &\rightarrow G_{(l)} \\ A_{(g)} + C_{(g)} + E_{(g)} &\rightarrow H_{(l)} \\ A_{(g)} + E_{(g)} &\rightarrow F_{(l)} \\ 3D_{(g)} &\rightarrow 2F_{(l)} \end{aligned} \tag{A.1}$$

The process contains 11 manipulated (Table A2) and 41 measured (Table A1) variables.

Table A1: Measured variables in the Tennessee Eastman process.

Variable No	Description	Measurement Type
1	Feed A (Stream 1)	Process
2	Feed D (Stream 2)	Process
3	Feed E (Stream 3)	Process
4	Total Feed (Stream 4)	Process
5	Recycle Flow (Stream 8)	Process
6	Reactor Feed Rate (Stream 6)	Process
7	Reactor Pressure	Process
8	Reactor Level	Process
9	Reactor Temperature	Process
10	Purge Rate (Stream 9)	Process
11	Product Separator Temperature	Process
12	Product Separator Level	Process
13	Product Separator Pressure	Process
14	Product Separator Underflow	Process
15	Stripper Level	Process
16	Stripper Pressure	Process
17	Stripper Underflow (Stream 11)	Process
18	Stripper Temperature	Process
19	Stripper Steam Flow	Process
20	Compressor Work	Process
21	Reactor Cooling Water Outlet Temperature	Process
22	Separator Cooling Water Outlet Temperature	Process
23	Component A (Stream 6)	Composition
24	Component B (Stream 6)	Composition
25	Component C (Stream 6)	Composition
26	Component D (Stream 6)	Composition
27	Component E (Stream 6)	Composition
28	Component F (Stream 6)	Composition
29	Component A (Stream 9)	Composition
30	Component B (Stream 9)	Composition
31	Component C (Stream 9)	Composition
32	Component D (Stream 9)	Composition
33	Component E (Stream 9)	Composition
34	Component F (Stream 9)	Composition
35	Component G (Stream 9)	Composition
36	Component H (Stream 9)	Composition
37	Component D (Stream 11)	Composition
38	Component E (Stream 11)	Composition
39	Component F (Stream 11)	Composition
40	Component G (Stream 11)	Composition
41	Component H (Stream 11)	Composition

Table A2: Manipulated variables in the Tennessee Eastman process.

Variable No	Description
42	D Feed Flow (Stream 2)
43	E Feed Flow (Stream 3)
44	A Feed Flow (Stream 1)
45	Total Feed Flow (Stream 4)
46	Compressor Recycle Valve
47	Purge Valve (Stream 9)
48	Separator Pot Liquid Flow (Stream 10)
49	Stripper Liquid Product Flow
50	Stripper Steam Valve
51	Reactor Cooling Water Flow
52	Condenser Cooling Water Flow

A.2 Performance Metric Terminology and Formulations

The model performances are assessed with 5 metrics: (i) area under the curve (AUC), (ii) fault detection rate, or recall, (iii) accuracy, (iv) false alarm rate, and (v) false negative rate. These metrics are derivations achieved from the confusion (a.k.a error) matrix, which is a two-dimensional contingency table used to evaluate performance of a classifier model in statistics and machine learning. Below, we provide terminology and formulation of the 5 metrics adopted in this dissertation.

Confusion matrix is a two-dimensional matrix containing the number of correct and false classification of instances for a binary classification problem. The elements of the matrix are True Positives (TP), False Positives (FP), True Negatives (TN), and False Negatives (FN). In terms of process monitoring, these numbers indicate:

True Positives (TP): Predicting faulty operation as faulty.

True Negatives (TN): Predicting fault-free (normal) operation as normal.

False Positives (FP): Predicting normal operation as faulty.

False Negatives (FN): Predicting faulty operation as normal.

Accordingly,

Positive (P): TP+FN

Negative (N): TN+FP

Accuracy is the percentage of correctly classified instances among all instances, which is calculated as follows:

$$Accuracy = \frac{TP+TN}{P+N}.$$

Accuracy is the most effective metric in the cases where class distribution is somewhat balanced. Recall (also referred as detection rate) is a measure of completeness.

$$Recall = \frac{TP}{TP+FN}.$$

In this work, misclassification of the instances are assessed via false alarm rate (FAR) and false negative rate (FNR):

$$FAR = \frac{FP}{FP+TN}.$$

$$FNR = \frac{FN}{FN+TP}.$$

In the case of imbalanced data sets, collective evaluation of two metrics, namely recall and specificity, gains importance where

$$Specificity = \frac{TN}{TN+FP}.$$

Receiver Operating Characteristics (ROC) curve is the plot illustrating the classifier performance based on recall and specificity at varying classification thresholds. Particularly, the curve demonstrates true positive rate (recall) versus false positive rate (1-$Specificity$) for all possible classification thresholds and area under this curve (a.k.a Area Under the Curve (AUC)) is a single metric derived from this advanced performance assessment.

A.3 *C*-SVM Model Parameters

Table A3: SVM hyperparameters for the models reported in Table 3.2.

Fault	**Optimal Feature Subset Size**	$\hat{C}$	$\hat{\gamma}$
1	2	6	0
2	5	2	-4
3	13	0	3
4	1	-8	-1
5	4	10	-6
6	2	-9	0
7	3	-8	0
8	7	3	4
9	17	2	2
10	14	6	4
11	29	5	3
12	9	10	7
13	7	10	3
14	3	-8	3
15	27	1	2
16	5	4	2
17	32	10	-5
18	34	10	-4
19	2	9	10
20	14	8	3
21	1	0	9

Table A4: SVM hyperparameters for the models reported in Table 3.4.

Fault	**Optimal Feature Subset Size**	$\hat{C}$	$\hat{\gamma}$
1	18	3	0
2	10	9	-3
3	10	9	7
4	1	-8	1
5	14	0	-1
6	2	-10	0
7	4	-1	5
8	12	4	1
9	2	1	4
10	15	10	-2
11	2	9	-2
12	5	4	4
13	8	5	1
14	2	-5	8
15	16	0	5
16	2	2	3
17	28	10	-5
18	5	9	0
19	10	10	-3
20	14	10	-3

Table A5: SVM hyperparameters for the models reported in Table 3.3.

Fault	**Optimal Feature Subset Size**	$\hat{C}$	$\hat{\gamma}$
8	4	5	4
17	27	10	-5

Table A6: C-SVM hyperparameters for the models reported in Table 3.5.

Fault	**Optimal Feature Subset Size**	$\hat{C}$	$\hat{\gamma}$
5	3	9	-7
18	2	9	1
19	3	10	-2
20	13	10	-3

A.4 Fault Diagnosis

The diagnosis of the entire faults of the *Chaing et. al* and *Rieth et. al* data sets are provided below. Note that Faults 3, 9, and 15 are excluded due to poor model performance. Asterisks imply the existence of alternative models.

Table A7: Diagnosis from the Table 3.2 end-models developed with *Chiang et. al* data set.

Fault	Optimal Feature Subset Size	Selected Process Variables
1	2	16, 44
2	5	7, 16, 10, 47, 13
3	13	24, 28, 29, 26, 33, 31, 50, 25, 30, 27, 35, 18, 1
4	1	51
5	4	4, 11, 52, 17
6	2	44, 1
7	3	45, 7, 13
8	7	39, 44, 16, 20, 7, 23, 46
9	17	39, 41, 38, 40, 37, 50, 18, 19, 7, 13, 16, 20, 31, 33, 29, 35, 28
10	14	41, 39, 38, 37, 40, 50, 19, 18, 20, 7, 13, 16, 31, 29
11	29	24, 29, 26, 30, 31, 32, 35, 50, 25, 33, 34, 23, 27, 18, 7, 16, 20, 38, 10, 19, 13, 37, 39, 44, 1, 41, 51, 47, 9
12	9	7, 16, 50, 18, 13, 19, 20, 38, 33
13	7	39, 40, 18, 7, 38, 23, 3
14	3	9, 51, 21
15	27	39, 41, 37, 40, 38, 50, 18, 19, 20, 7, 13, 16, 1, 44, 25, 31, 23, 33, 29, 36, 35, 34, 24, 30, 27, 47, 10
16	5	50, 19, 18, 20, 13
17	32	38, 39, 40, 41, 21, 37, 19, 20, 33, 27, 34, 30, 1, 11, 25, 28, 24, 23, 35, 36, 26, 10, 3, 2, 22, 14, 48, 47, 32, 42, 8, 49
18	34	39, 40, 37, 41, 14, 49, 17, 48, 5, 52, 15, 12, 3, 9, 2, 32, 10, 26, 28, 24, 6, 11, 36, 20, 50, 22, 44, 13, 34, 1, 7, 8, 25, 18
19	2	32, 31
20	14	41, 39, 40, 37, 50, 18, 46, 13, 19, 7, 16, 11, 33, 27
21	1	45

Table A8: Diagnosis from the Table 3.4 end-models developed with *Rieth et. al* data set.

Fault	Optimal Feature Subset Size	Selected Process Variables
1	18	44, 16, 41, 4, 1, 11, 18, 21, 22, 20, 7, 51, 46, 38, 33, 13, 23, 24
2	10	38, 41, 10, 25, 16, 21, 7, 20, 47, 30
3	10	47, 51, 38, 16, 50, 18, 19, 21, 20, 13
4	1	51
5	14	52, 11, 17, 4, 18, 19, 46, 50, 20, 16, 44, 38, 29, 22
6	2	44, 1
7	4	19, 18, 50, 45
8	12	39, 41, 37, 16, 20, 44, 7, 46, 1, 27, 29, 40
9	2	51, 13
10	15	41, 38, 39, 37, 18, 19, 40, 25, 31, 29, 26, 23, 1, 50, 16
11	2	9, 51
12	5	16, 38, 35, 25, 11
13	8	41, 39, 7, 37, 40, 16, 32, 21
14	2	51, 9
15	16	22, 7, 13, 18, 50, 19, 11, 16, 38, 35, 20, 9, 21, 46, 4, 29
16	2	19, 50
17	28	35, 24, 38, 28, 18, 20, 19, 21, 46, 26, 36, 42, 37, 25, 29, 30, 39, 41, 40, 44, 32, 34, 22, 8, 10, 27, 31, 23
18	5	22, 8, 20, 11, 31
19	10	13, 16, 46, 50, 19, 5, 7, 20, 38, 6
20	14	38, 39, 41, 16, 52, 17, 18, 30, 35, 29, 40, 13, 7, 47

APPENDIX B

PARAMETRIC FAULT-TOLERANT CONTROL SYSTEMS DESIGN

B.1 Actuator Fault: Bias in Water Flow Rate

This section includes the complete set of reactor temperature and water flow rate profiles with $\pm$ 0.5 K threshold on the set point for each time-specific models for the actuator fault case.

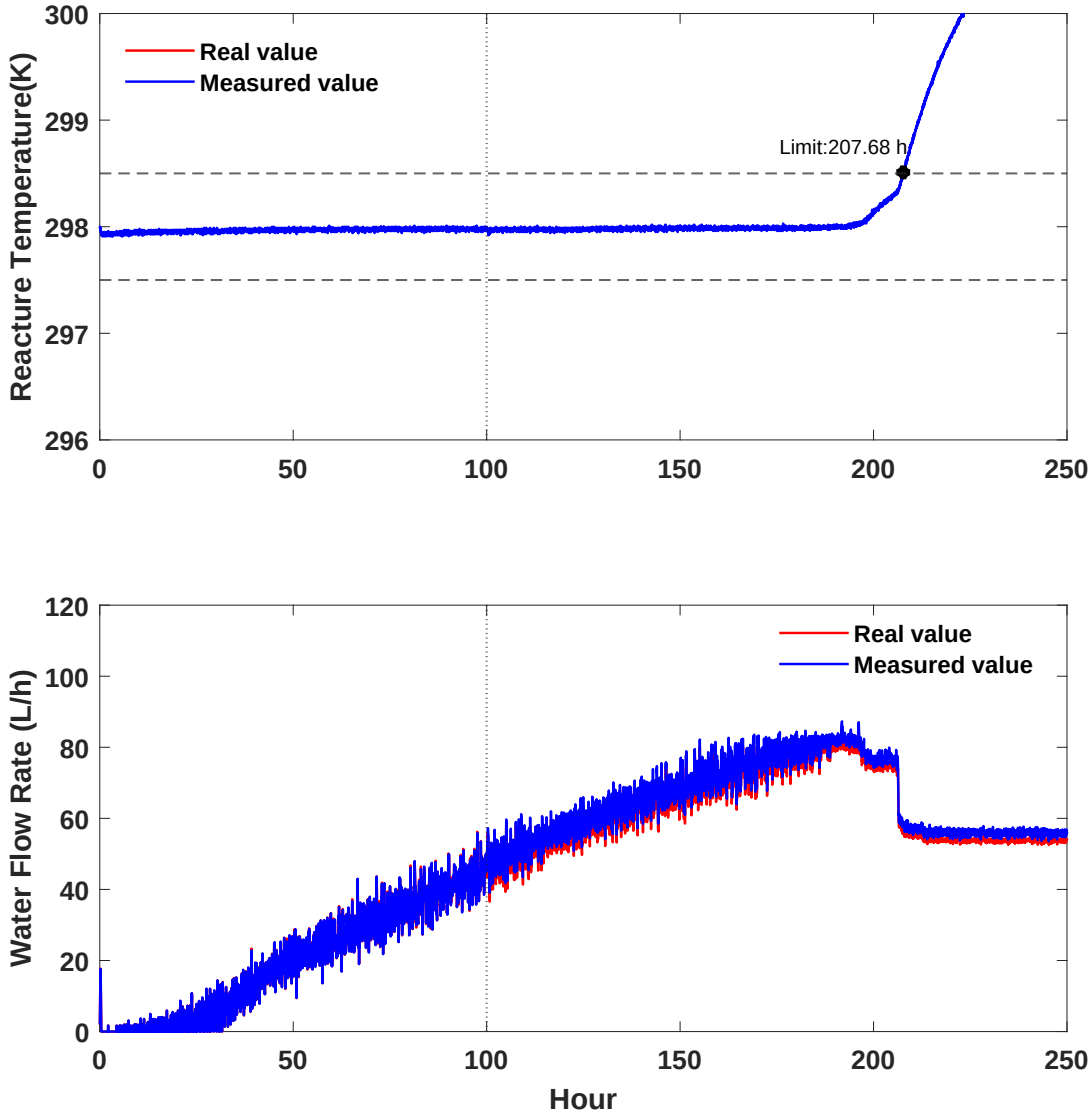


Figure A1: Reactor temperature and water flow rate profiles for process with actuator fault. Fault Onset: 100 h, Fault Magnitude: +2.0.

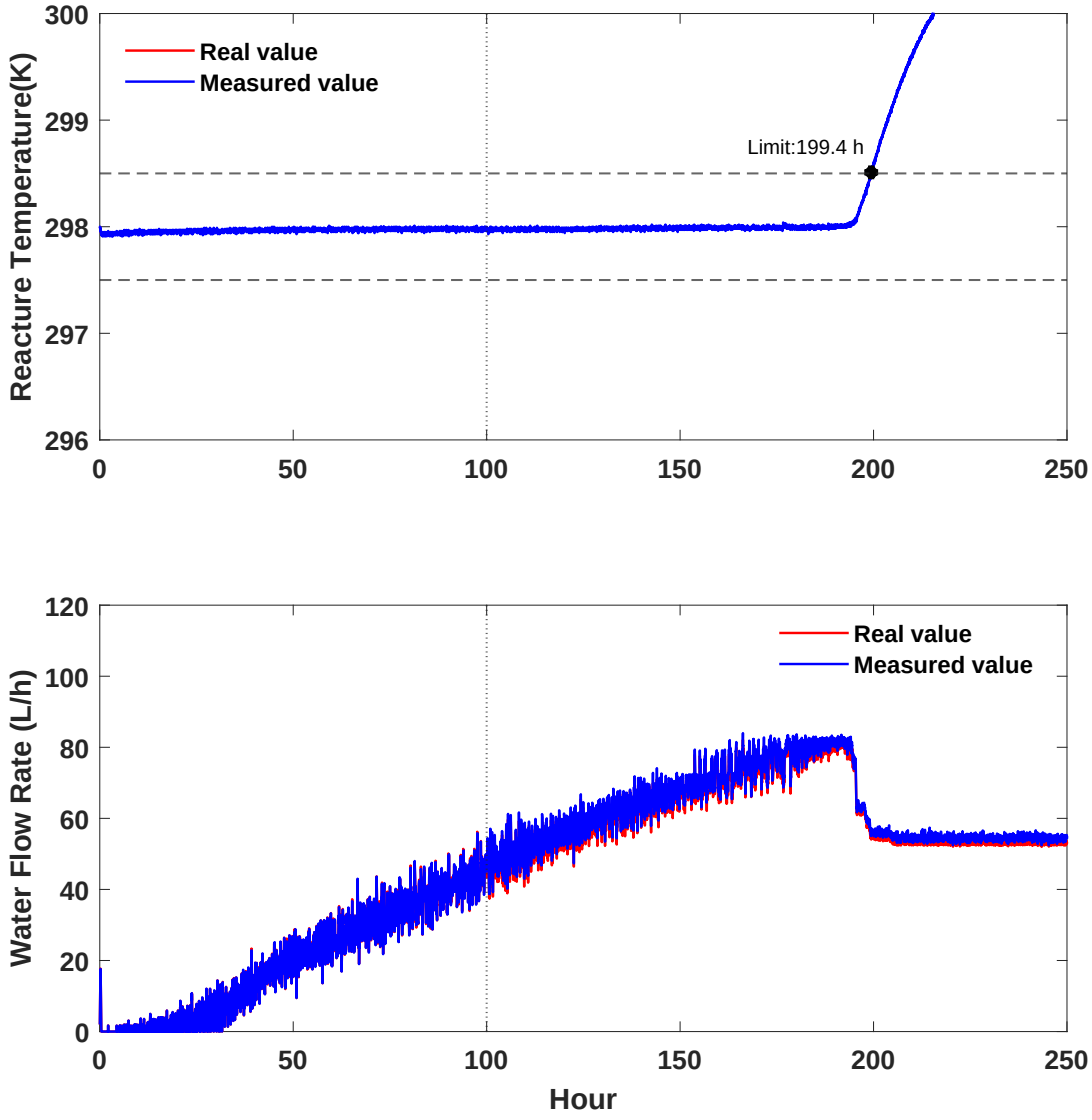


Figure A2: Reactor temperature and water flow rate profiles for process with actuator fault. Fault Onset: 100 h, Fault Magnitude: +1.5.

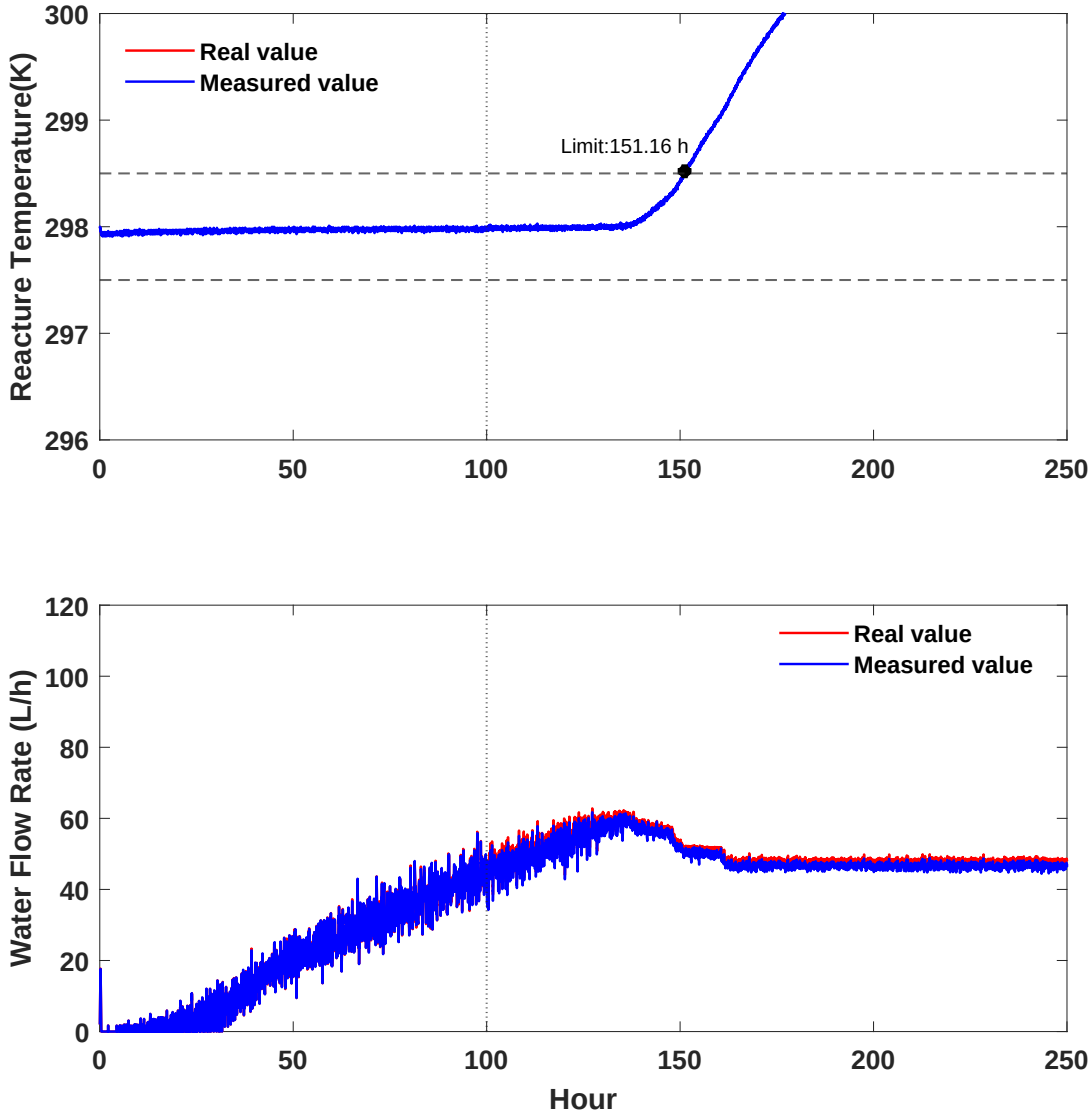


Figure A3: Reactor temperature and water flow rate profiles for process with actuator fault. Fault Onset: 100 h, Fault Magnitude: -1.5.

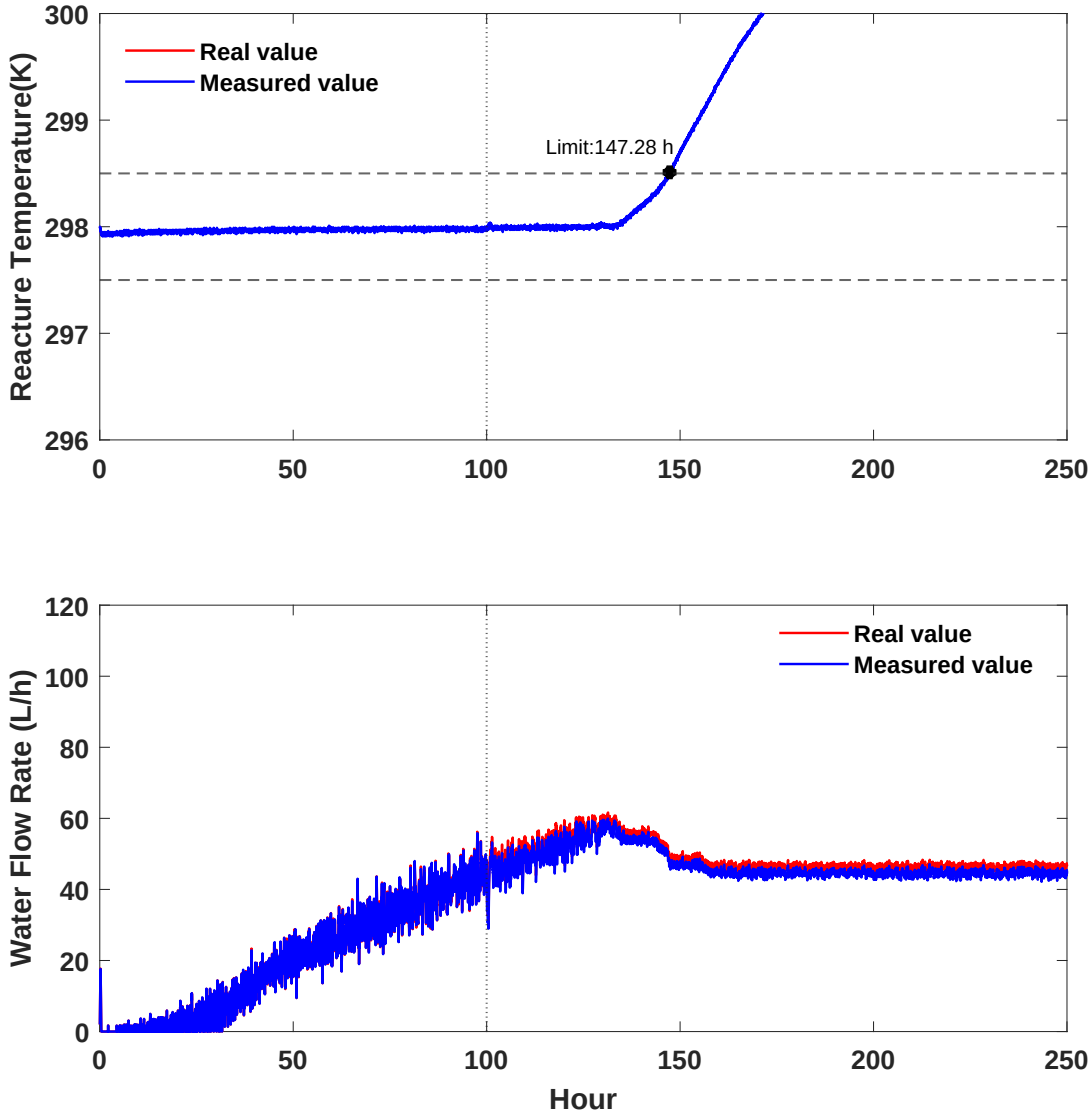


Figure A4: Reactor temperature and water flow rate profiles for process with actuator fault. Fault Onset: 100 h, Fault Magnitude: -2.0.

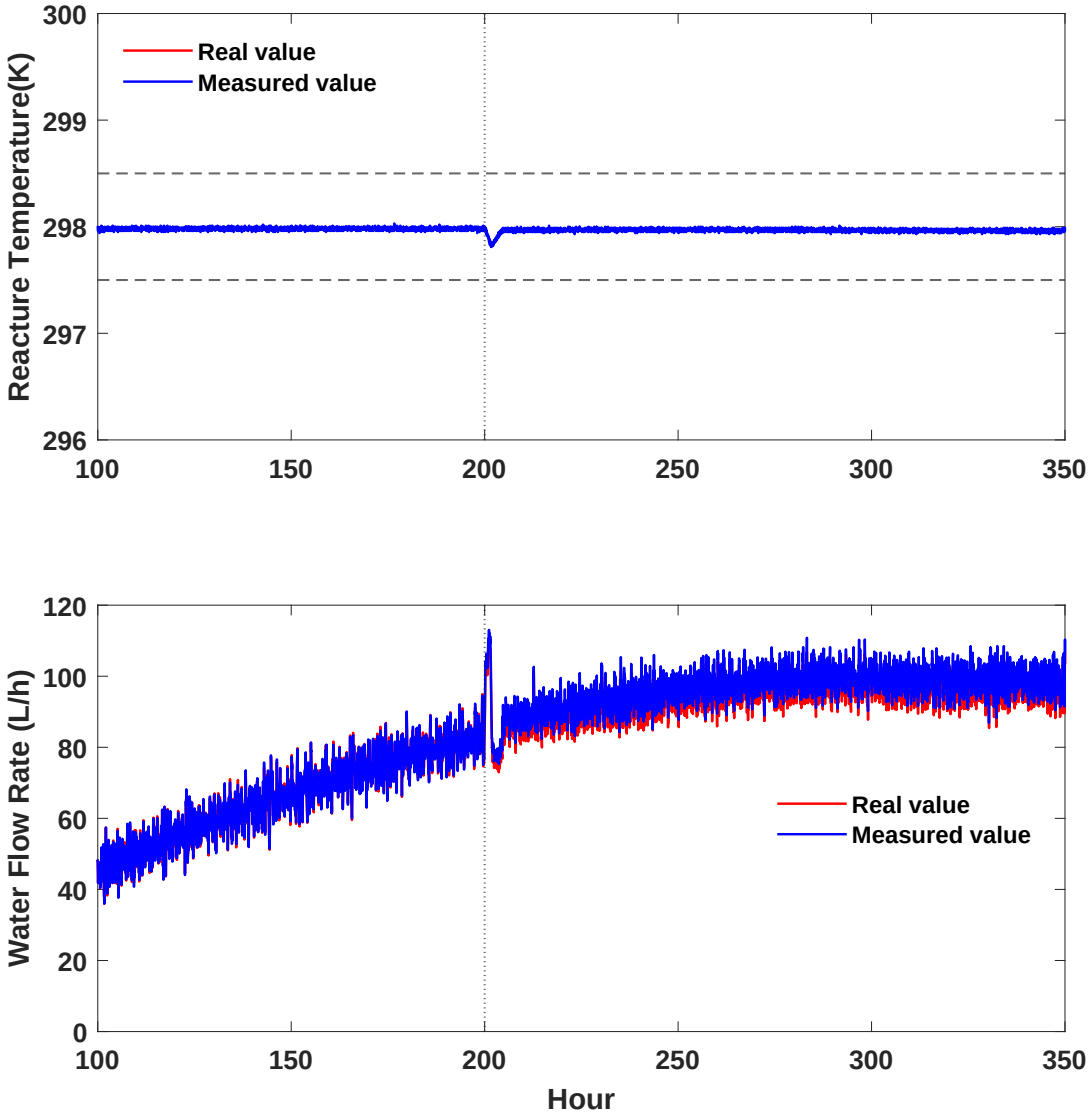


Figure A5: Reactor temperature and water flow rate profiles for process with actuator fault. Fault Onset: 200 h, Fault Magnitude: +2.5.

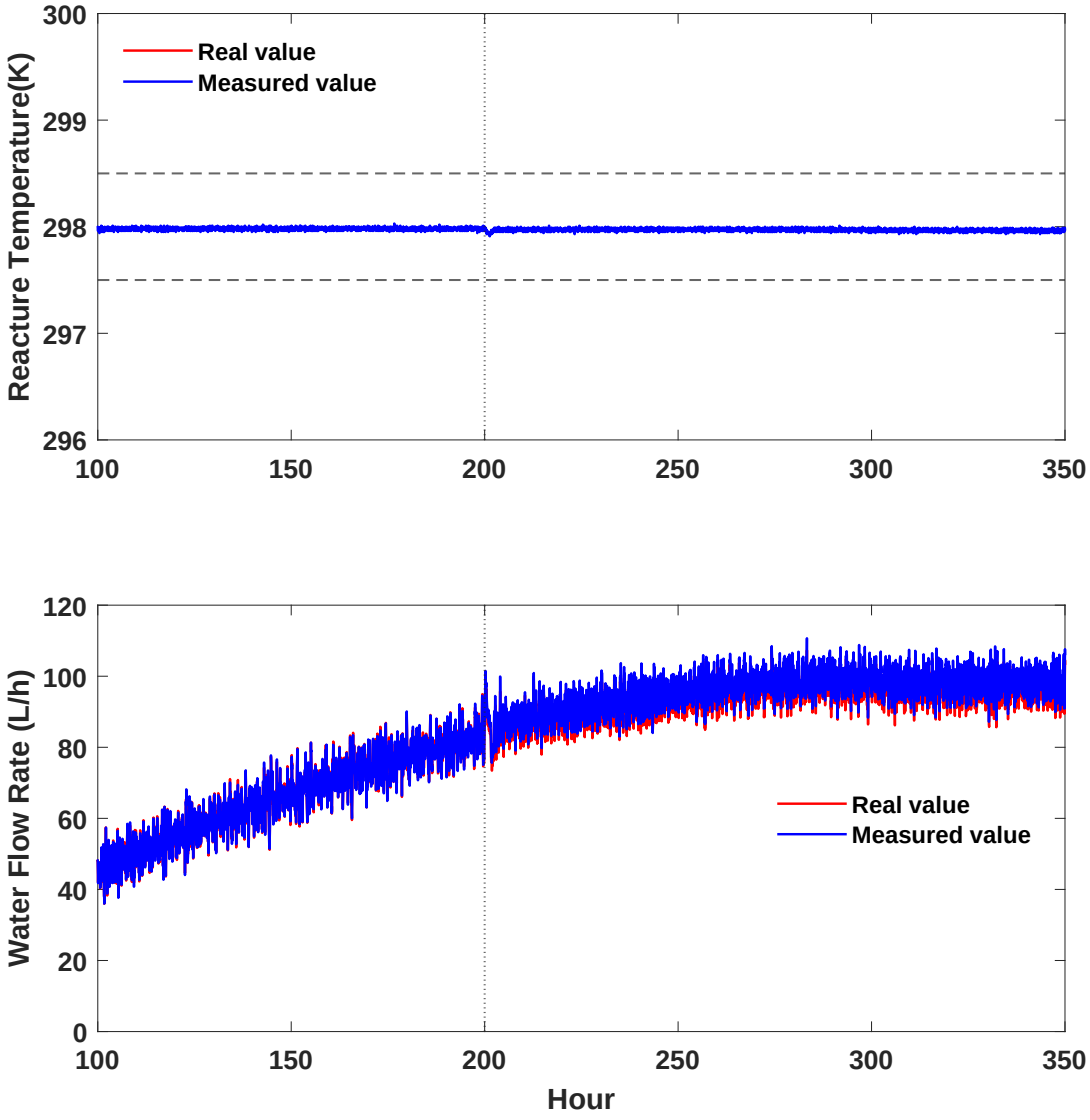


Figure A6: Reactor temperature and water flow rate profiles for process with actuator fault. Fault Onset: 200 h, Fault Magnitude: +2.0.

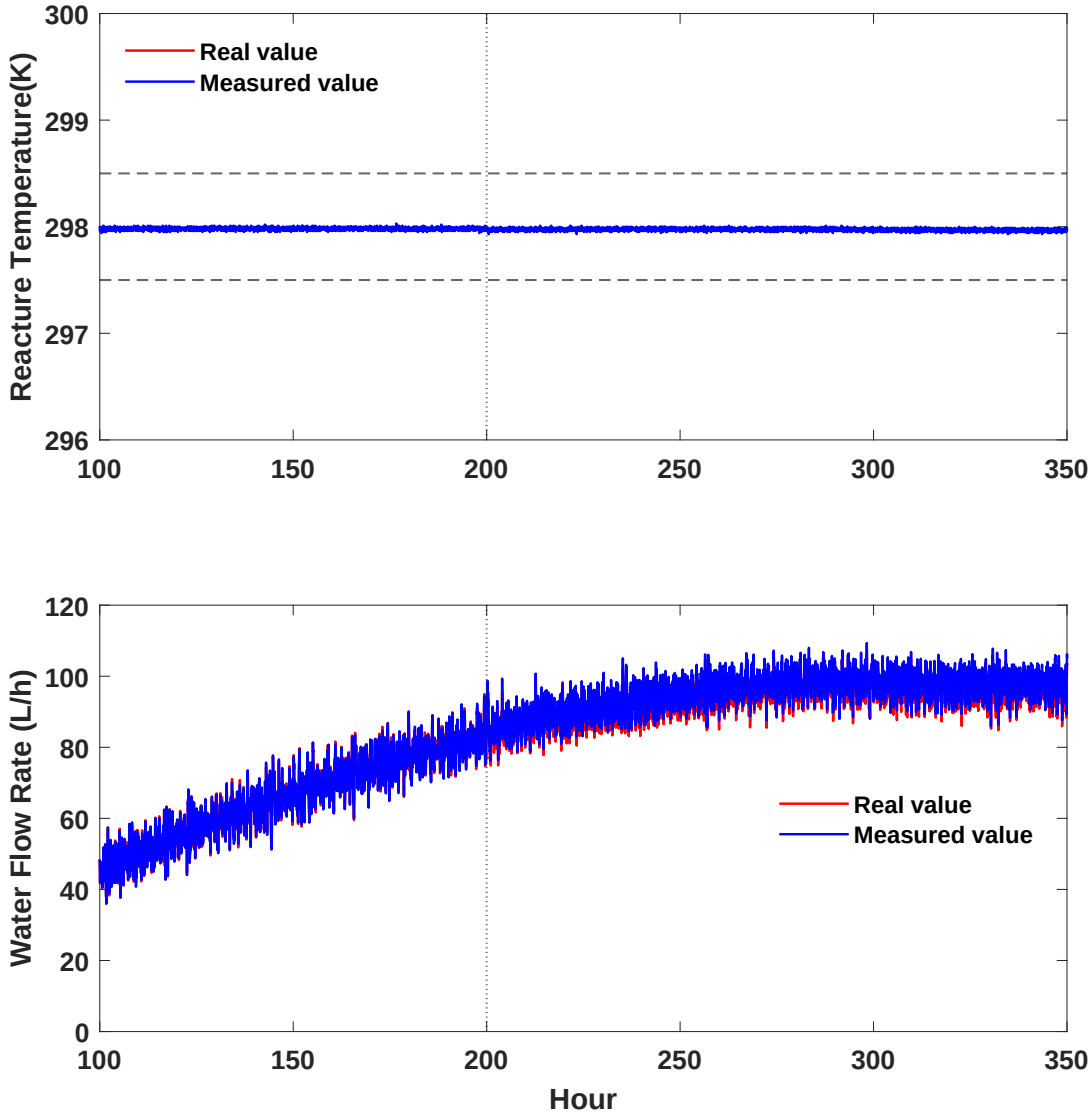


Figure A7: Reactor temperature and water flow rate profiles for process with actuator fault. Fault Onset: 200 h, Fault Magnitude: +1.5.

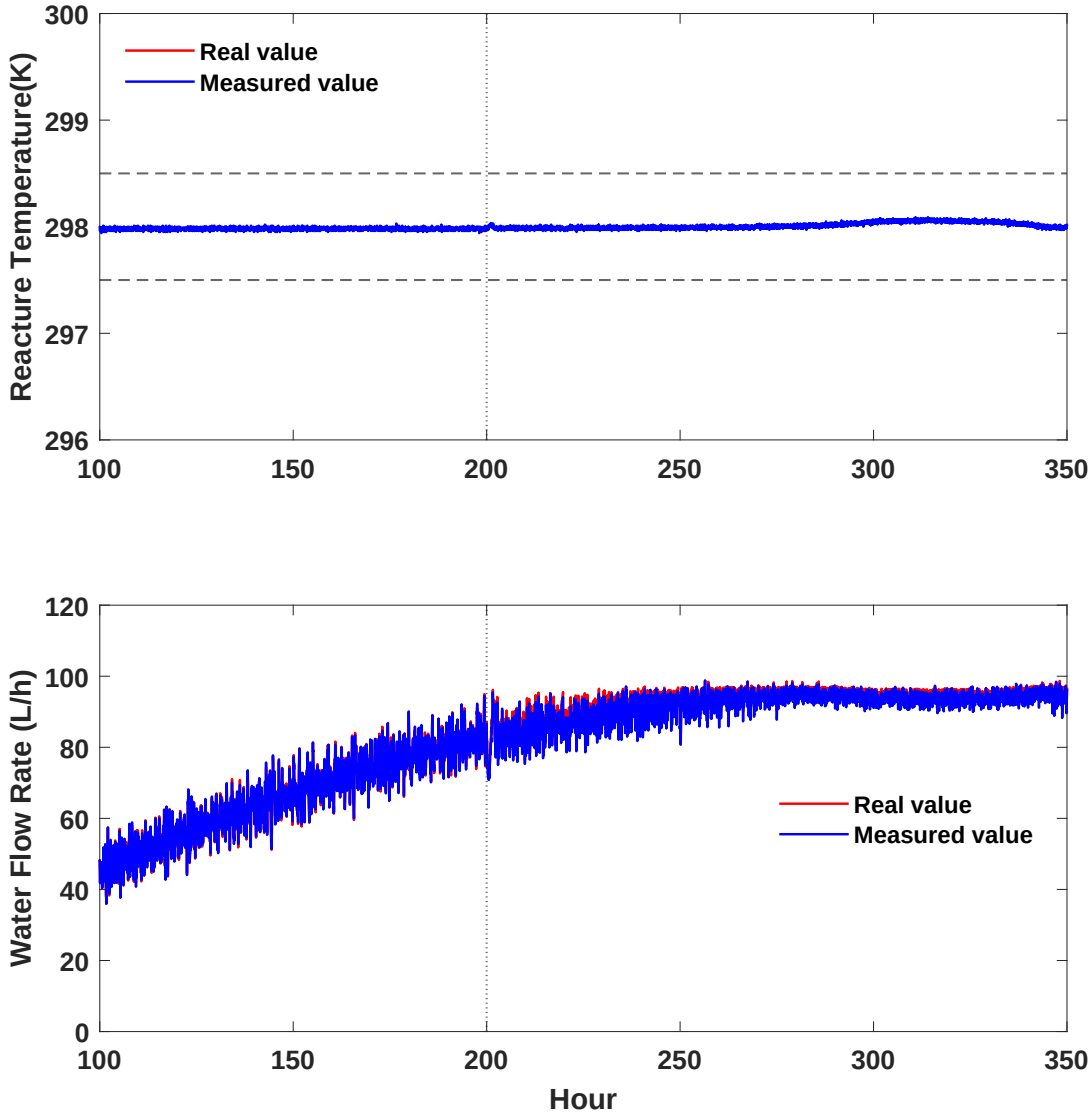


Figure A8: Reactor temperature and water flow rate profiles for process with actuator fault. Fault Onset: 200 h, Fault Magnitude: -1.5.

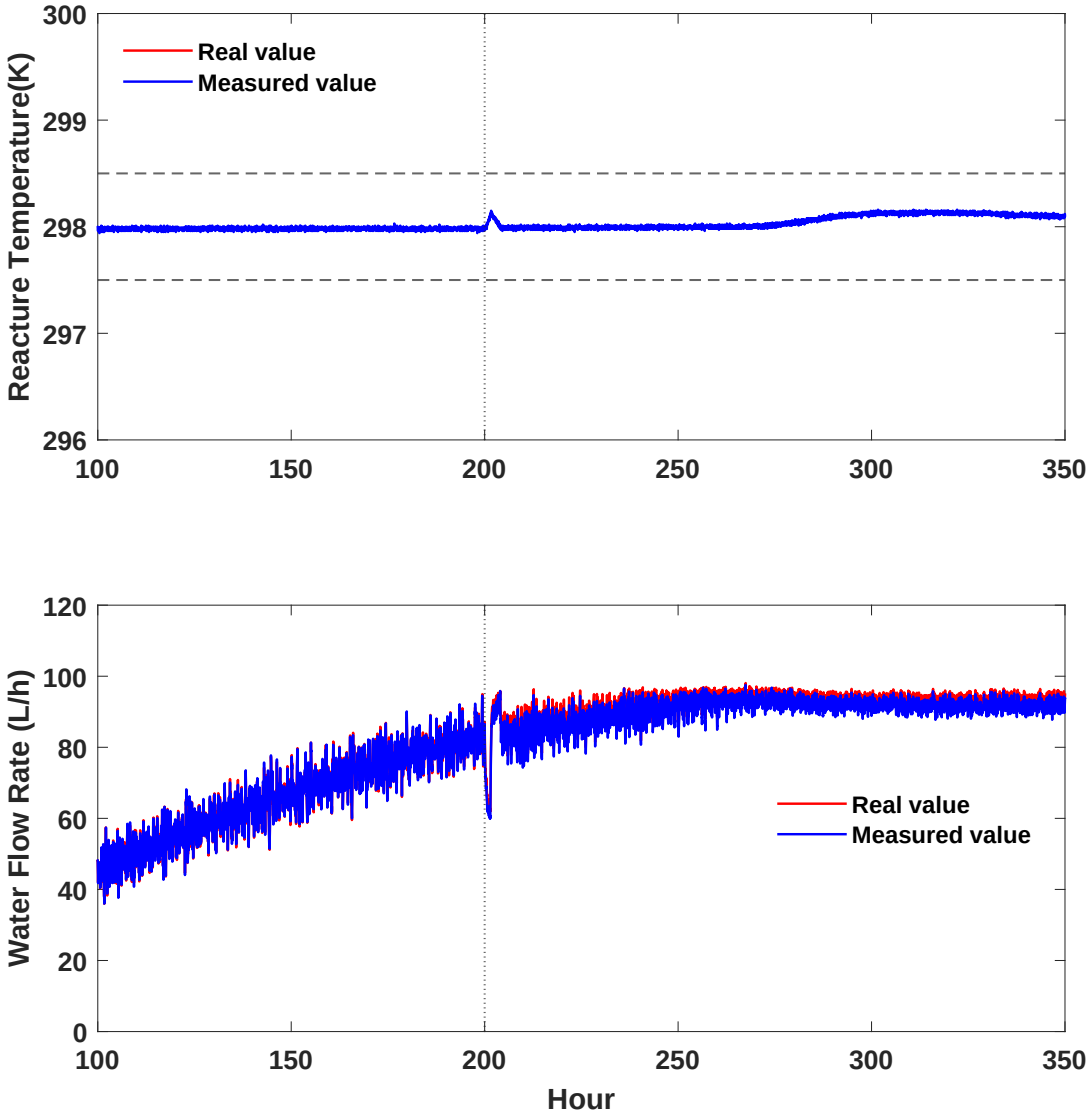


Figure A9: Reactor temperature and water flow rate profiles for process with actuator fault. Fault Onset: 200 h, Fault Magnitude: -2.0.

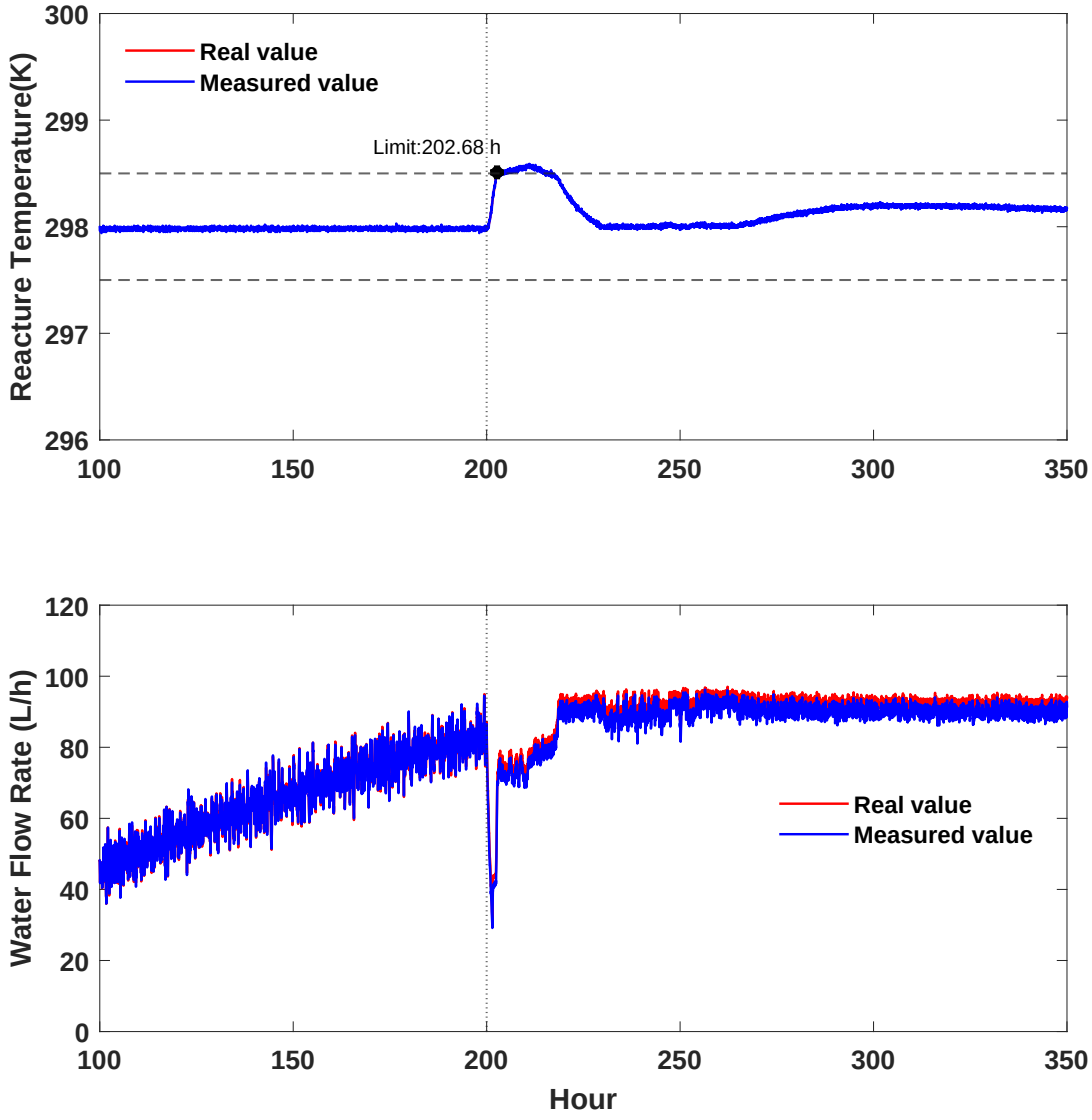


Figure A10: Reactor temperature and water flow rate profiles for process with actuator fault. Fault Onset: 200 h, Fault Magnitude: -2.5.

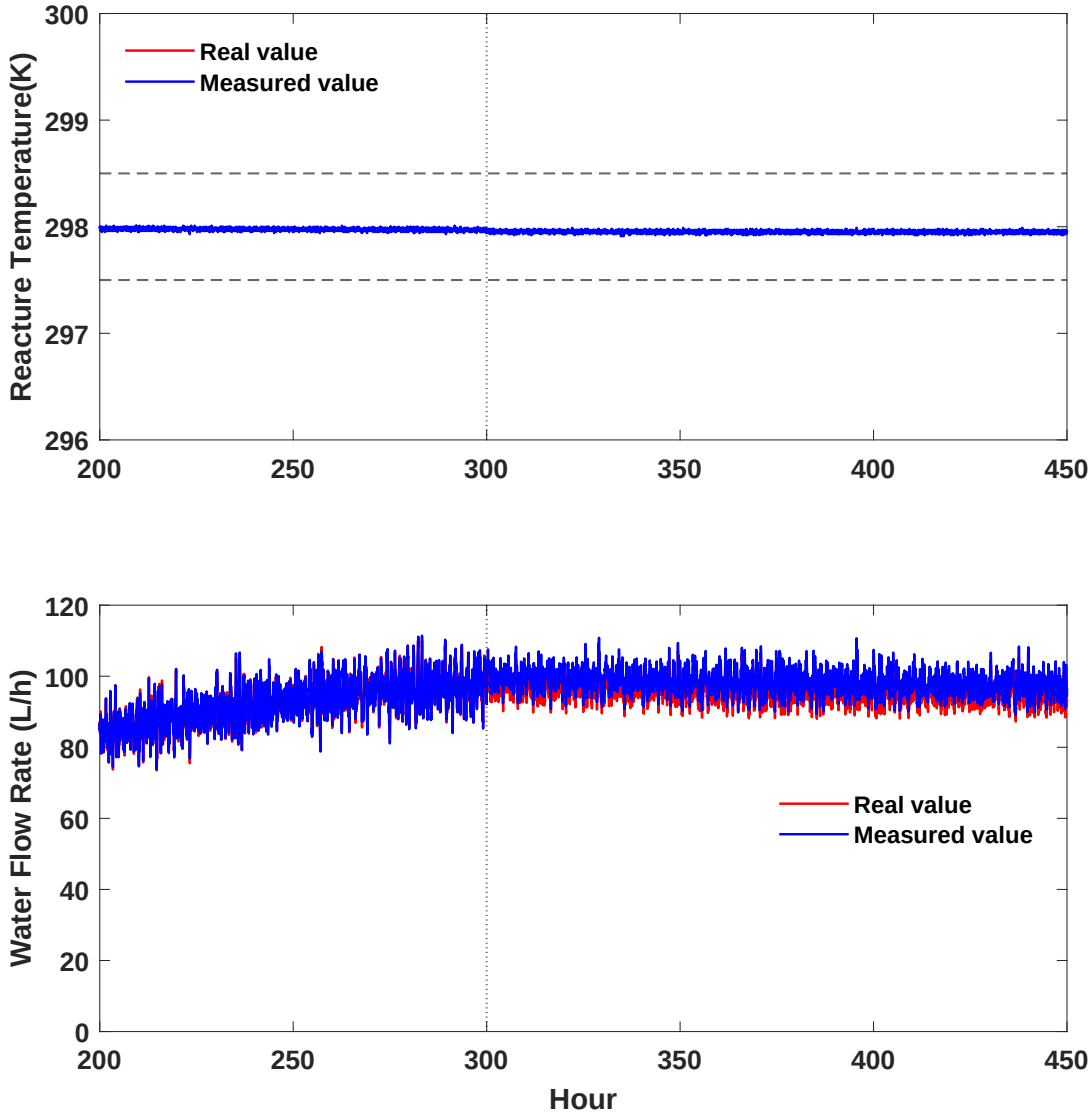


Figure A11: Reactor temperature and water flow rate profiles for process with actuator fault. Fault Onset: 300 h, Fault Magnitude: +2.5.

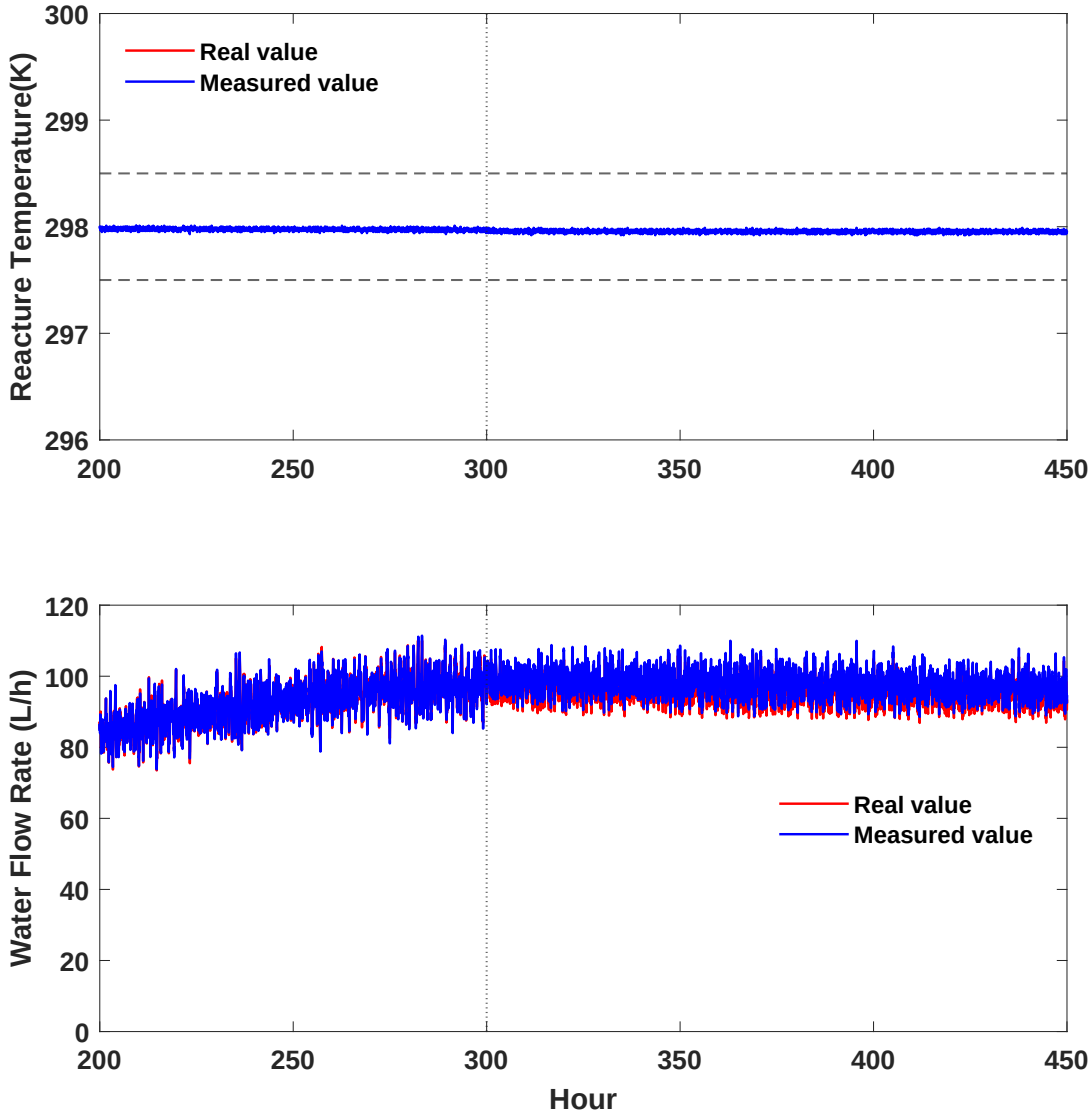


Figure A12: Reactor temperature and water flow rate profiles for process with actuator fault. Fault Onset: 300 h, Fault Magnitude: +2.0.

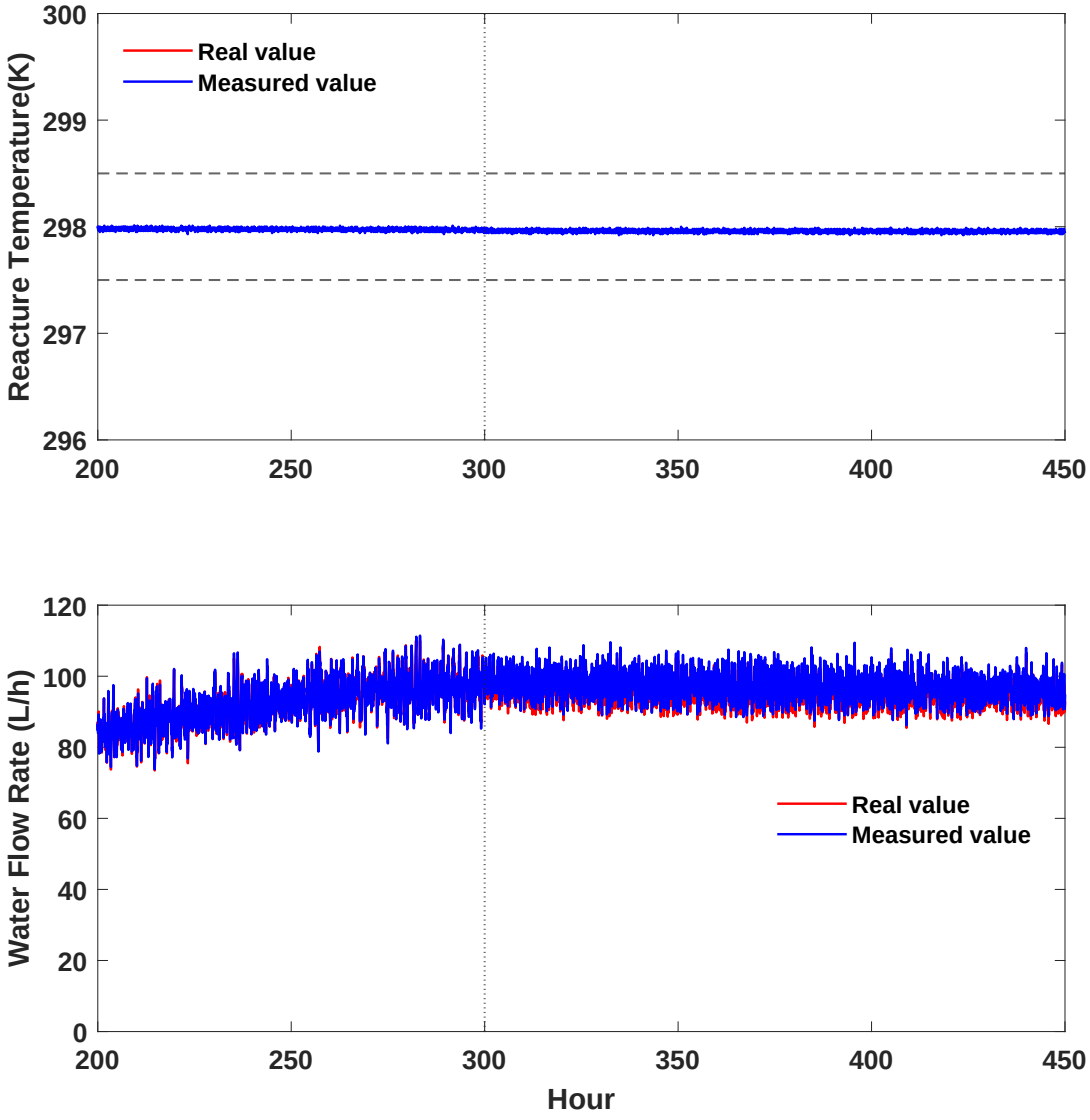


Figure A13: Reactor temperature and water flow rate profiles for process with actuator fault. Fault Onset: 300 h, Fault Magnitude: +1.5.

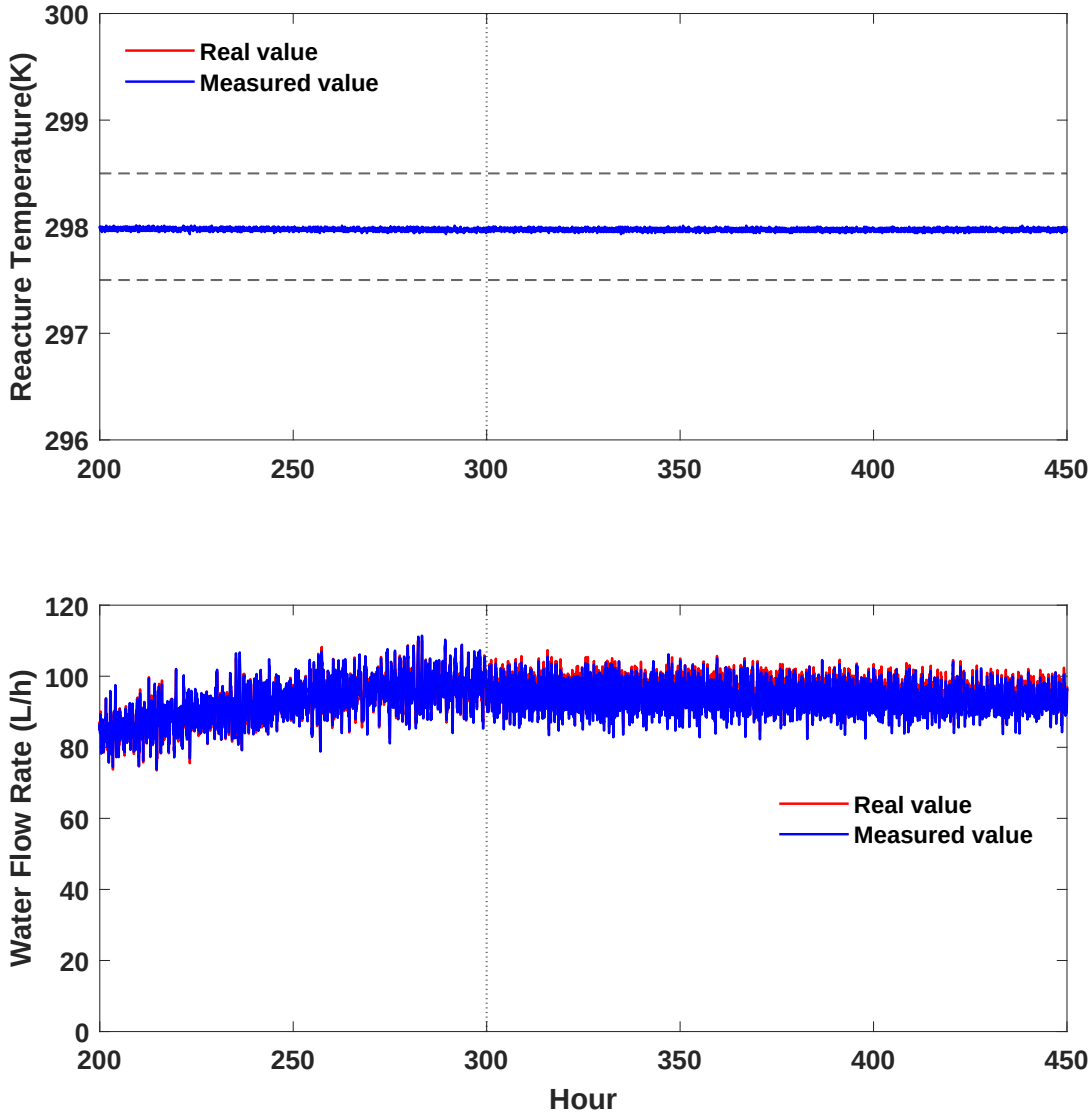


Figure A14: Reactor temperature and water flow rate profiles for process with actuator fault. Fault Onset: 300 h, Fault Magnitude: -1.5.

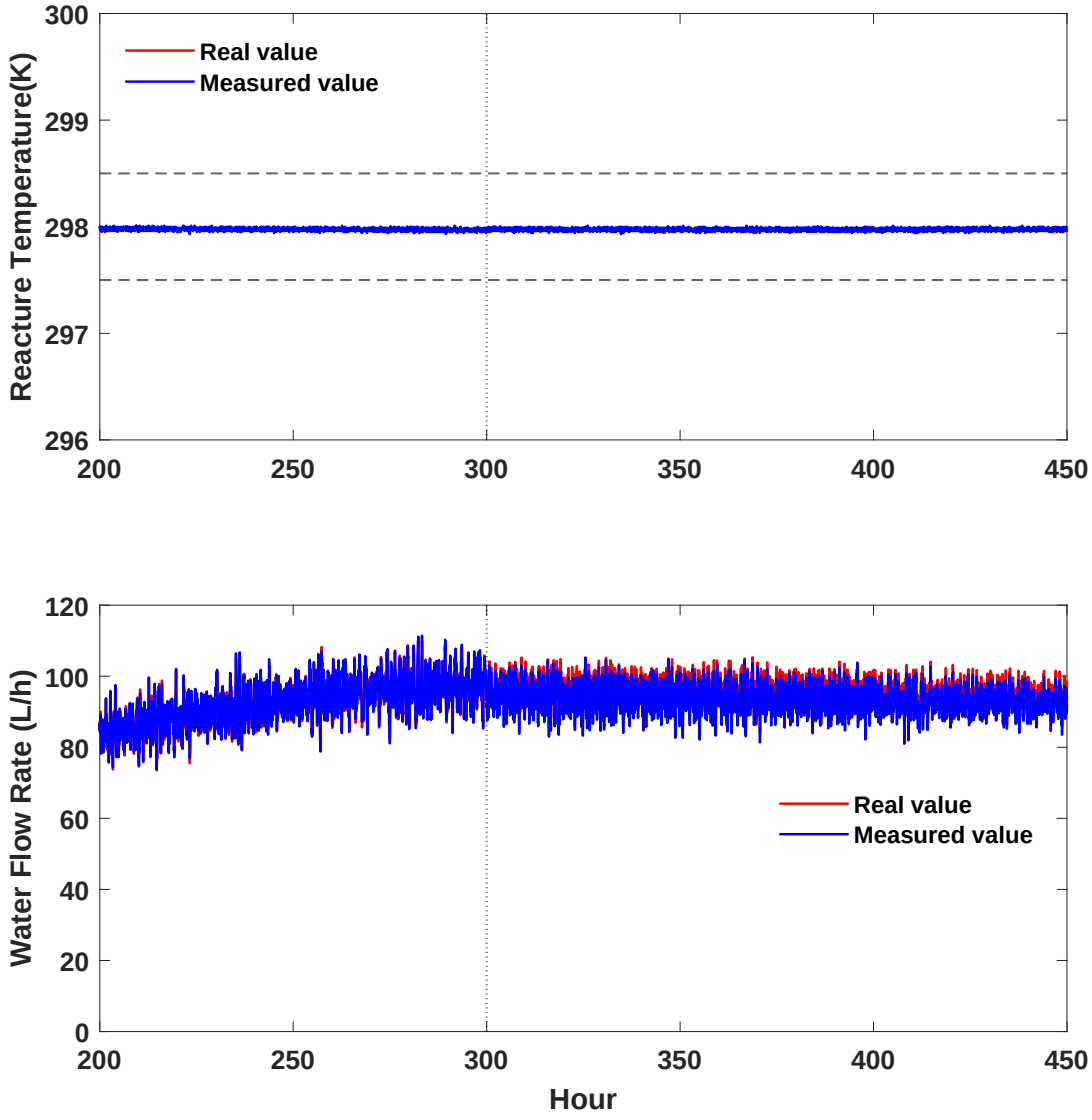


Figure A15: Reactor temperature and water flow rate profiles for process with actuator fault. Fault Onset: 300 h, Fault Magnitude: -2.0.

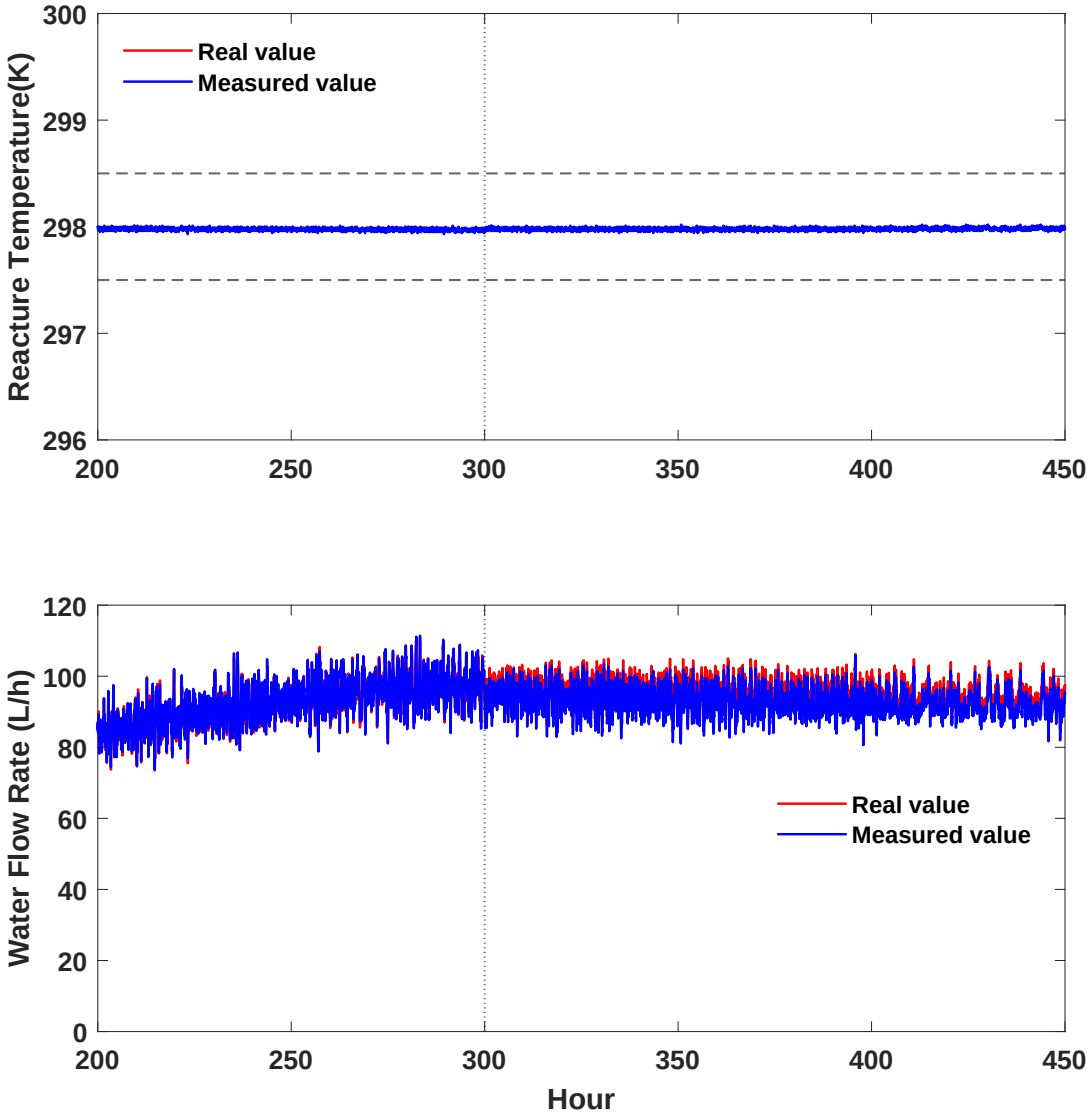


Figure A16: Reactor temperature and water flow rate profiles for process with actuator fault. Fault Onset: 300 h, Fault Magnitude: -2.5.

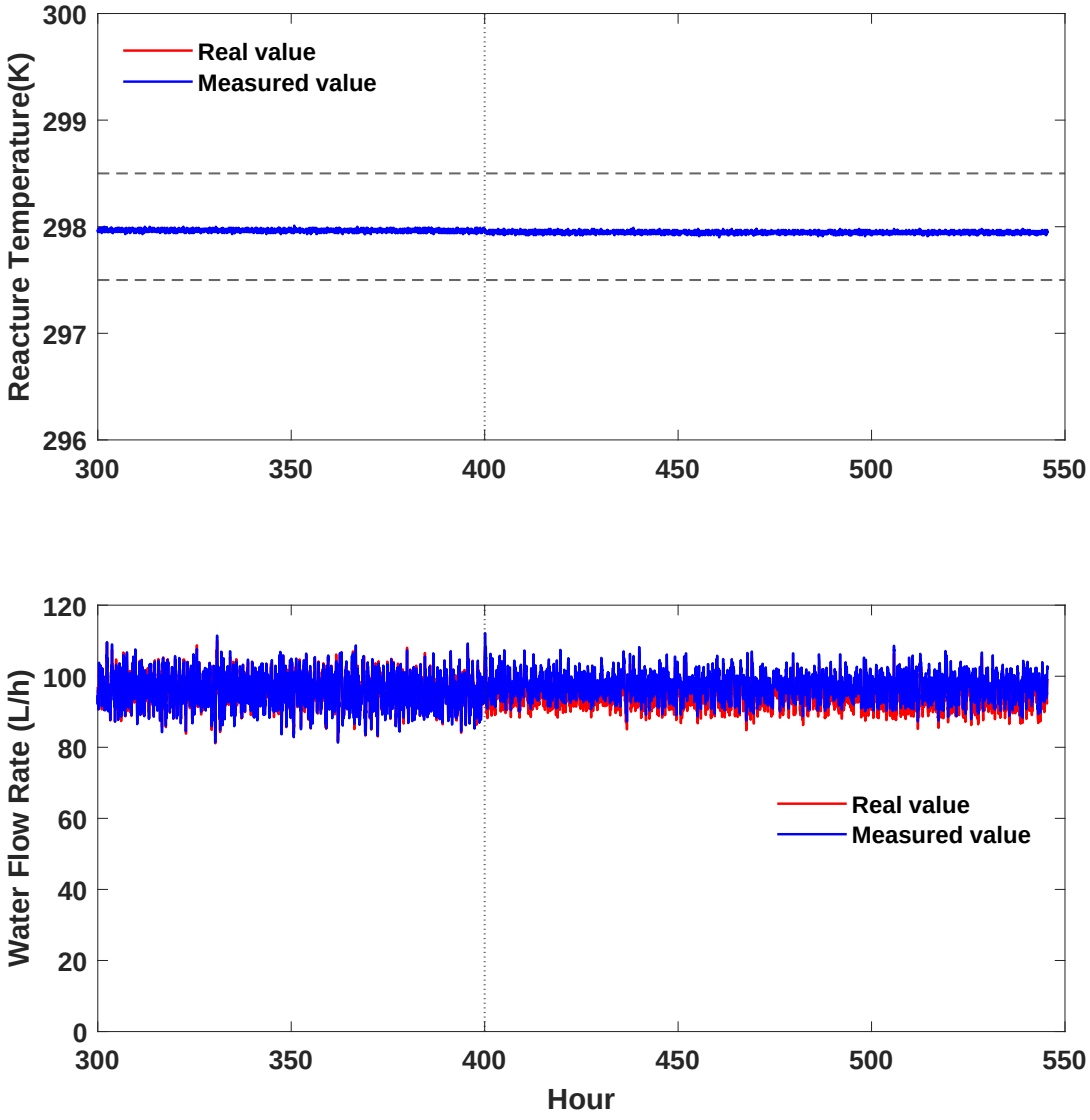


Figure A17: Reactor temperature and water flow rate profiles for process with actuator fault. Fault Onset: 400 h, Fault Magnitude: +2.5.

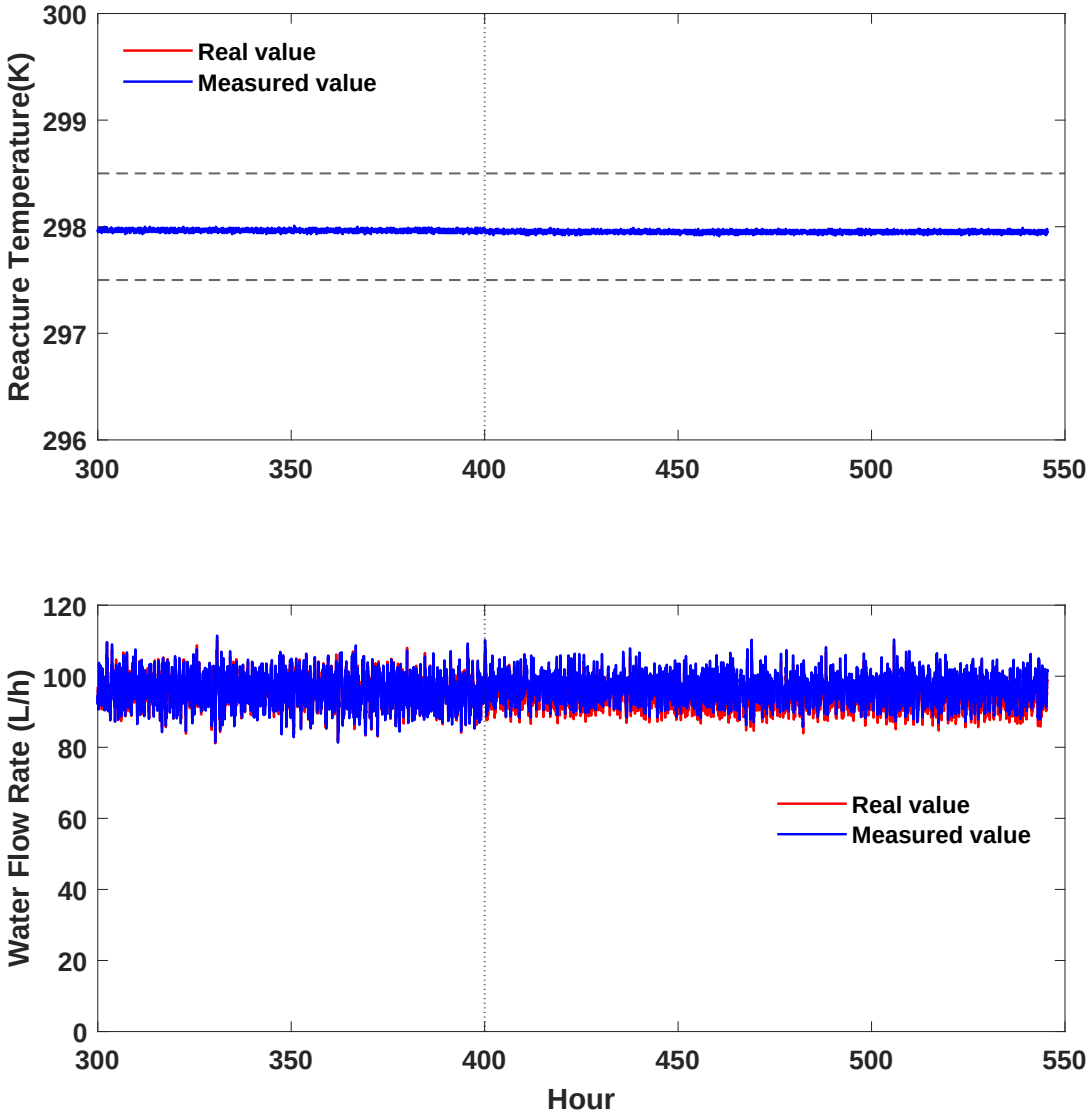


Figure A18: Reactor temperature and water flow rate profiles for process with actuator fault. Fault Onset: 400 h, Fault Magnitude: +2.0.

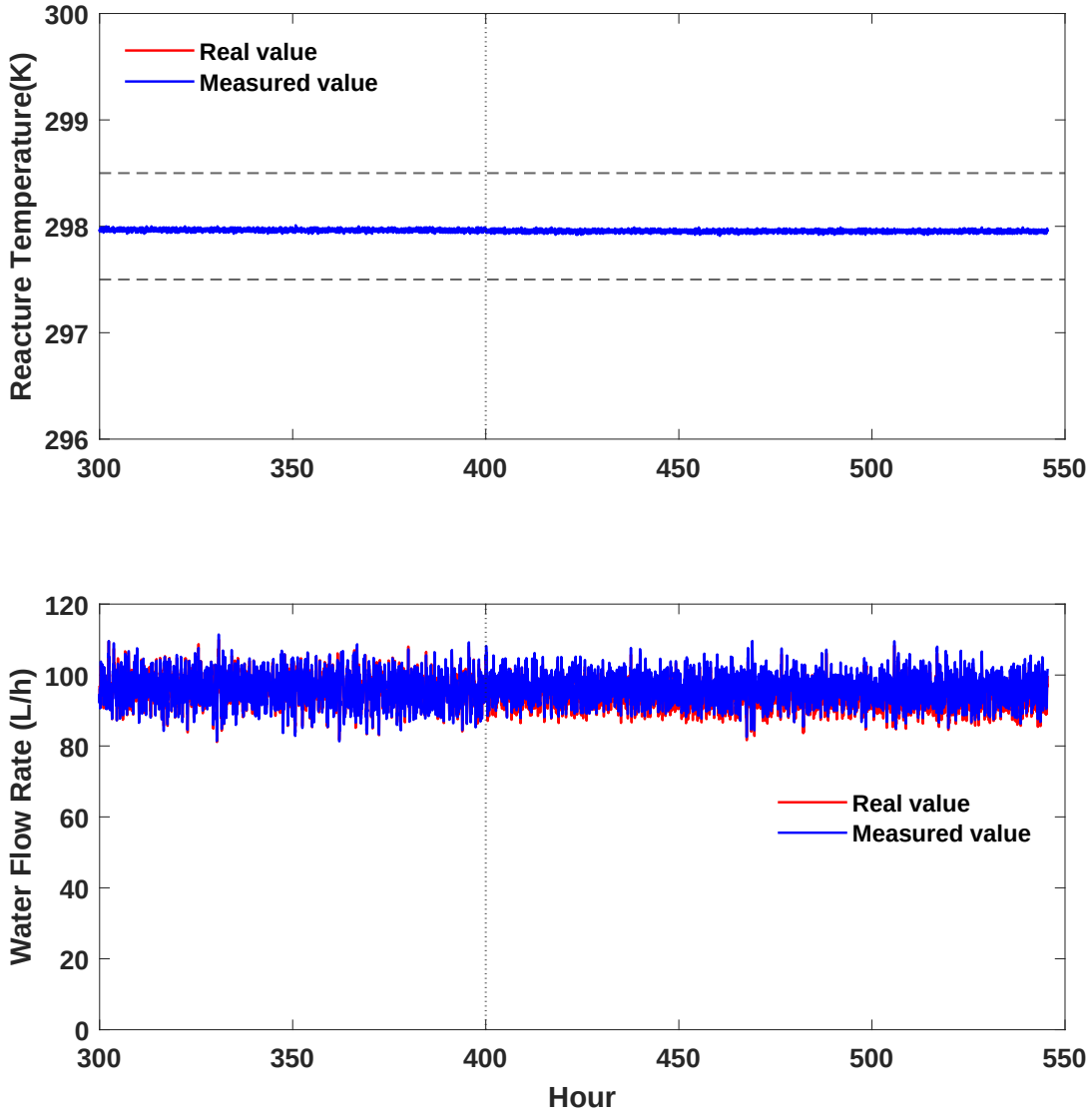


Figure A19: Reactor temperature and water flow rate profiles for process with actuator fault. Fault Onset: 400 h, Fault Magnitude: +1.5.

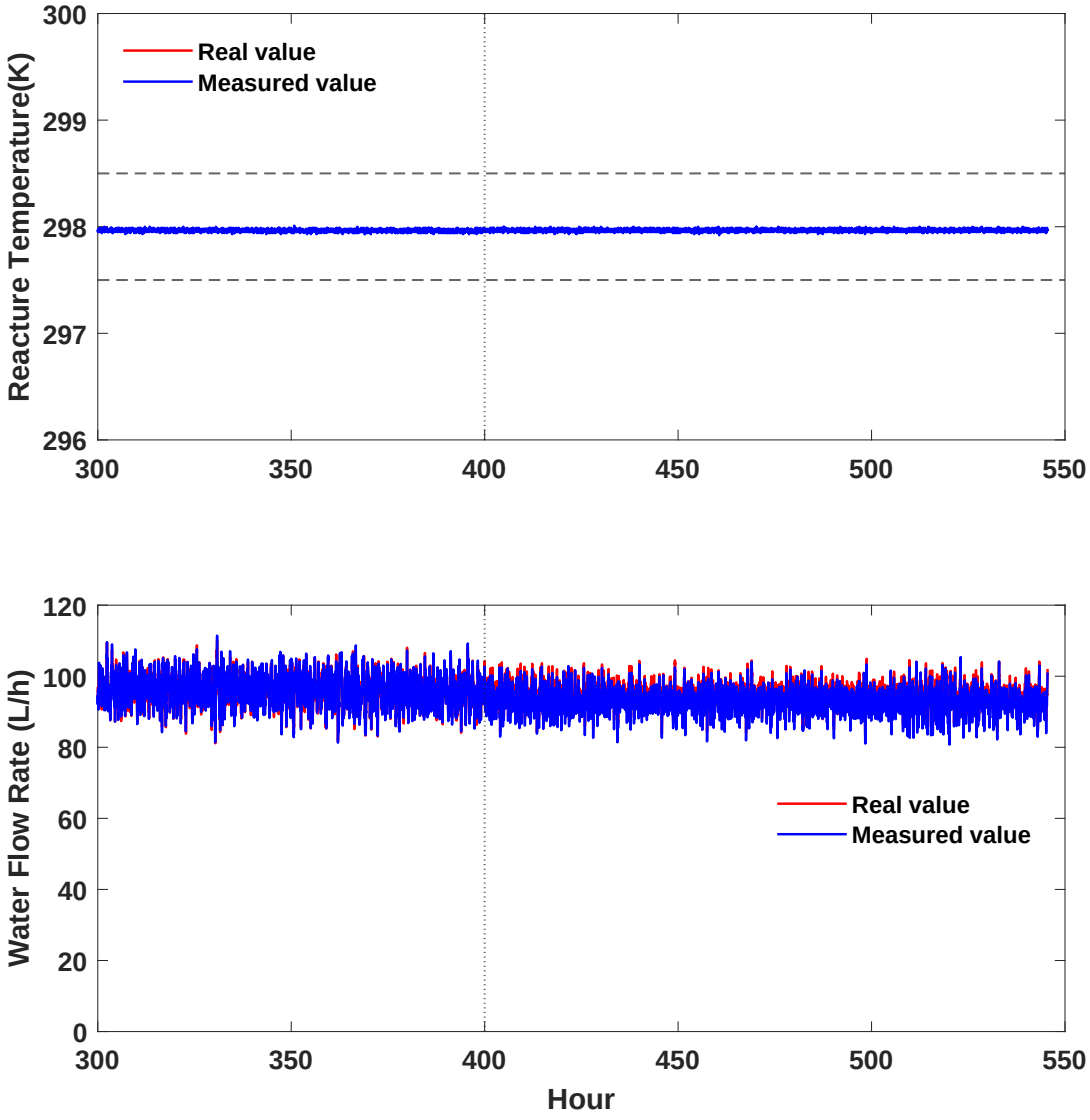


Figure A20: Reactor temperature and water flow rate profiles for process with actuator fault. Fault Onset: 400 h, Fault Magnitude: -1.5.

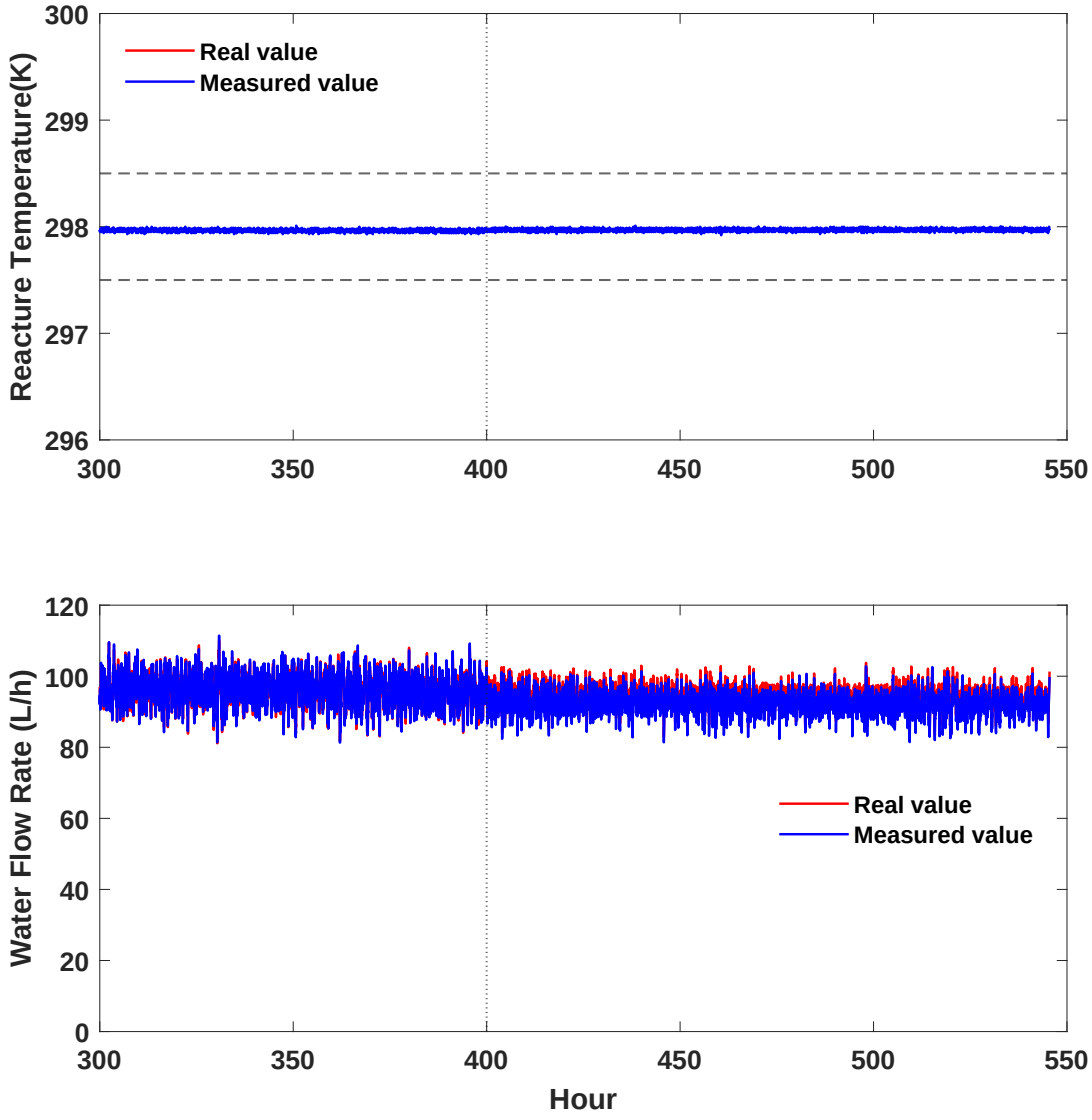


Figure A21: Reactor temperature and water flow rate profiles for process with actuator fault. Fault Onset: 400 h, Fault Magnitude: -2.0.

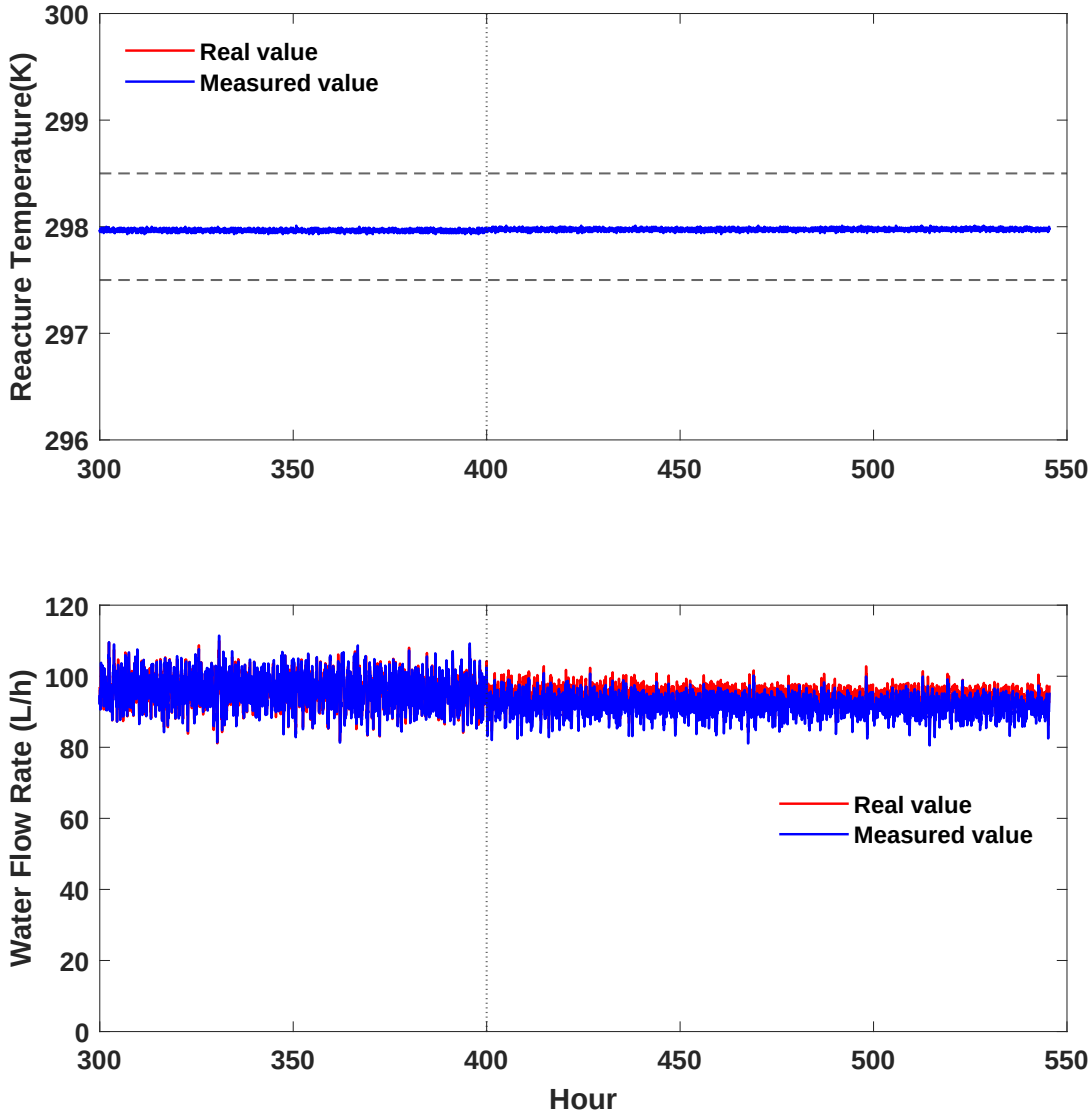


Figure A22: Reactor temperature and water flow rate profiles for process with actuator fault. Fault Onset: 400 h, Fault Magnitude: -2.5.

B.2 Sensor Fault: Bias in Reactor Temperature

This section includes the complete set of reactor temperature and water flow rate profiles with $\pm$ 0.5 K threshold on the set point for each time-specific models for the sensor fault case.

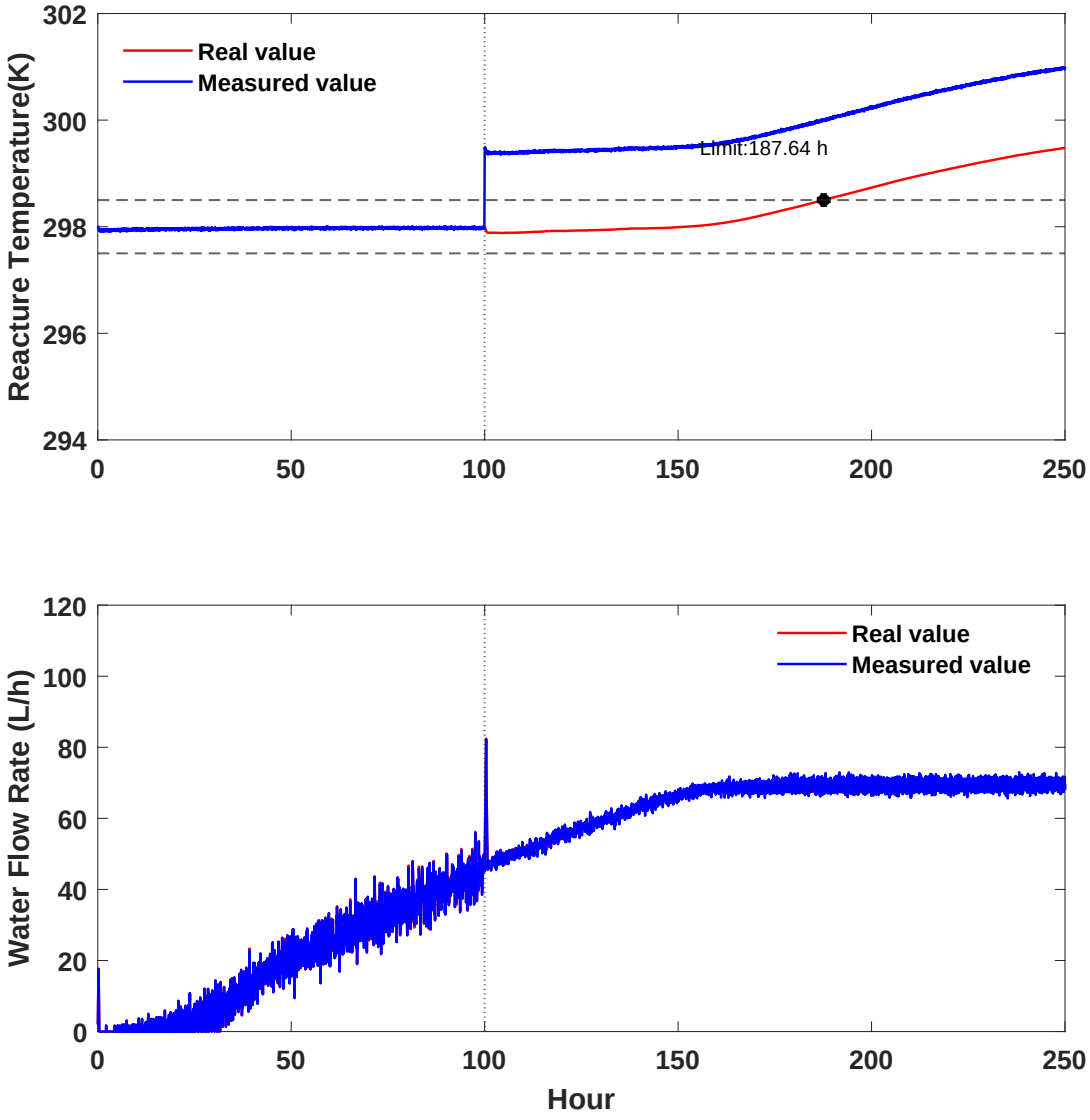


Figure A23: Reactor temperature and water flow rate profiles for process with sensor fault. Fault Onset: 100 h, Fault Magnitude: +1.5.

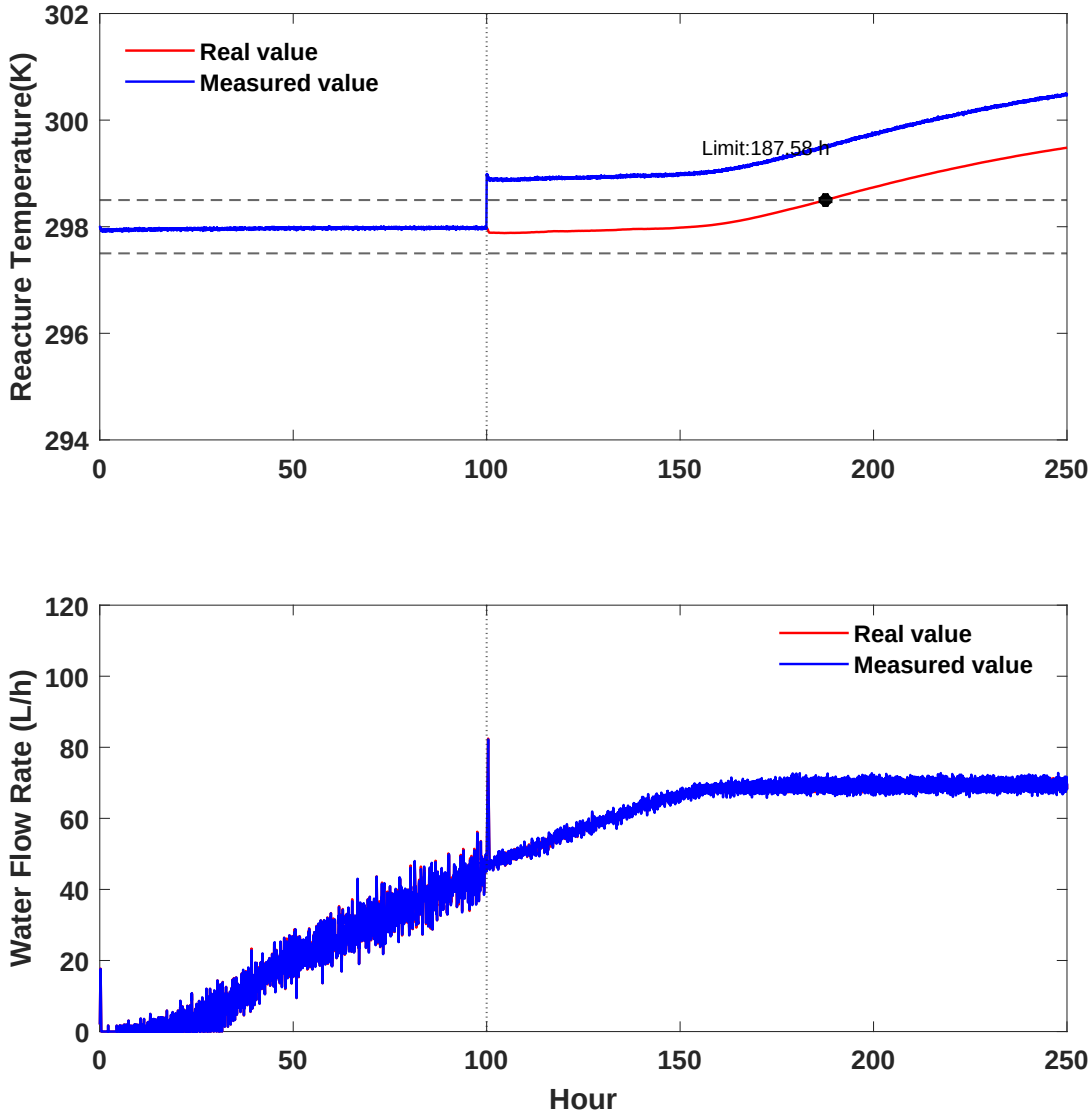


Figure A24: Reactor temperature and water flow rate profiles for process with sensor fault. Fault Onset: 100 h, Fault Magnitude: +1.0.

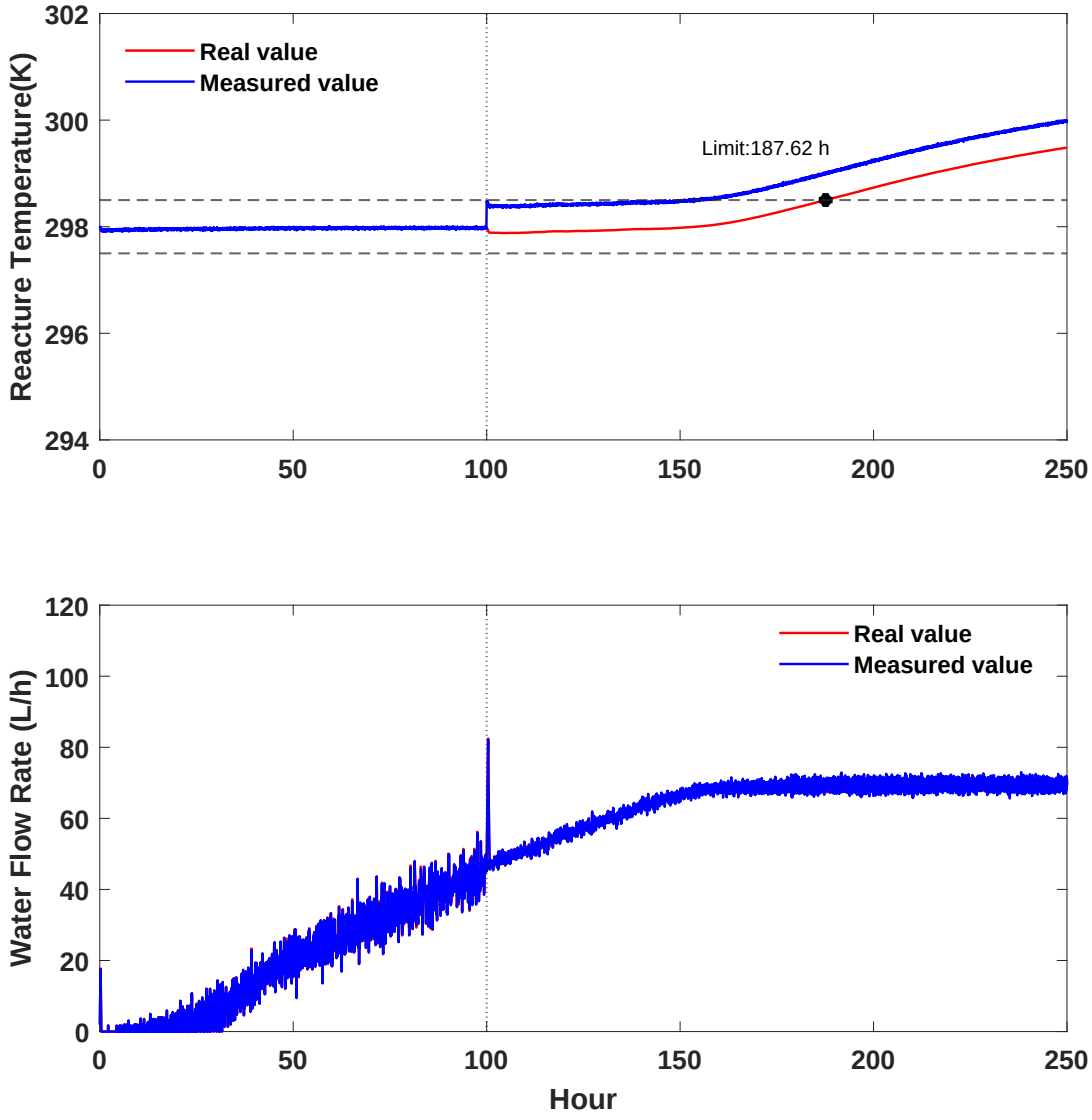


Figure A25: Reactor temperature and water flow rate profiles for process with sensor fault. Fault Onset: 100 h, Fault Magnitude: +0.5.

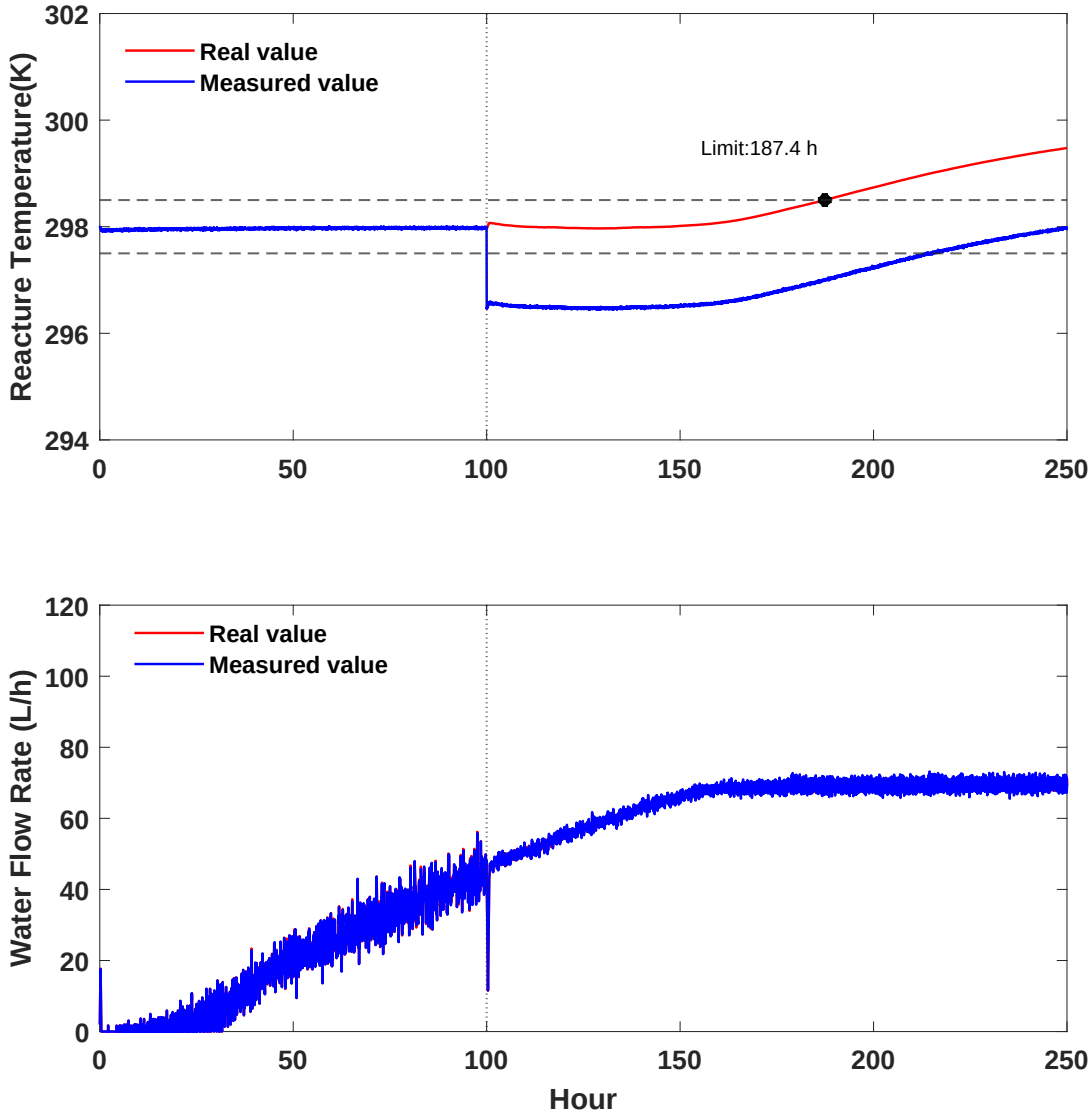


Figure A26: Reactor temperature and water flow rate profiles for process with sensor fault. Fault Onset: 100 h, Fault Magnitude: -1.5.

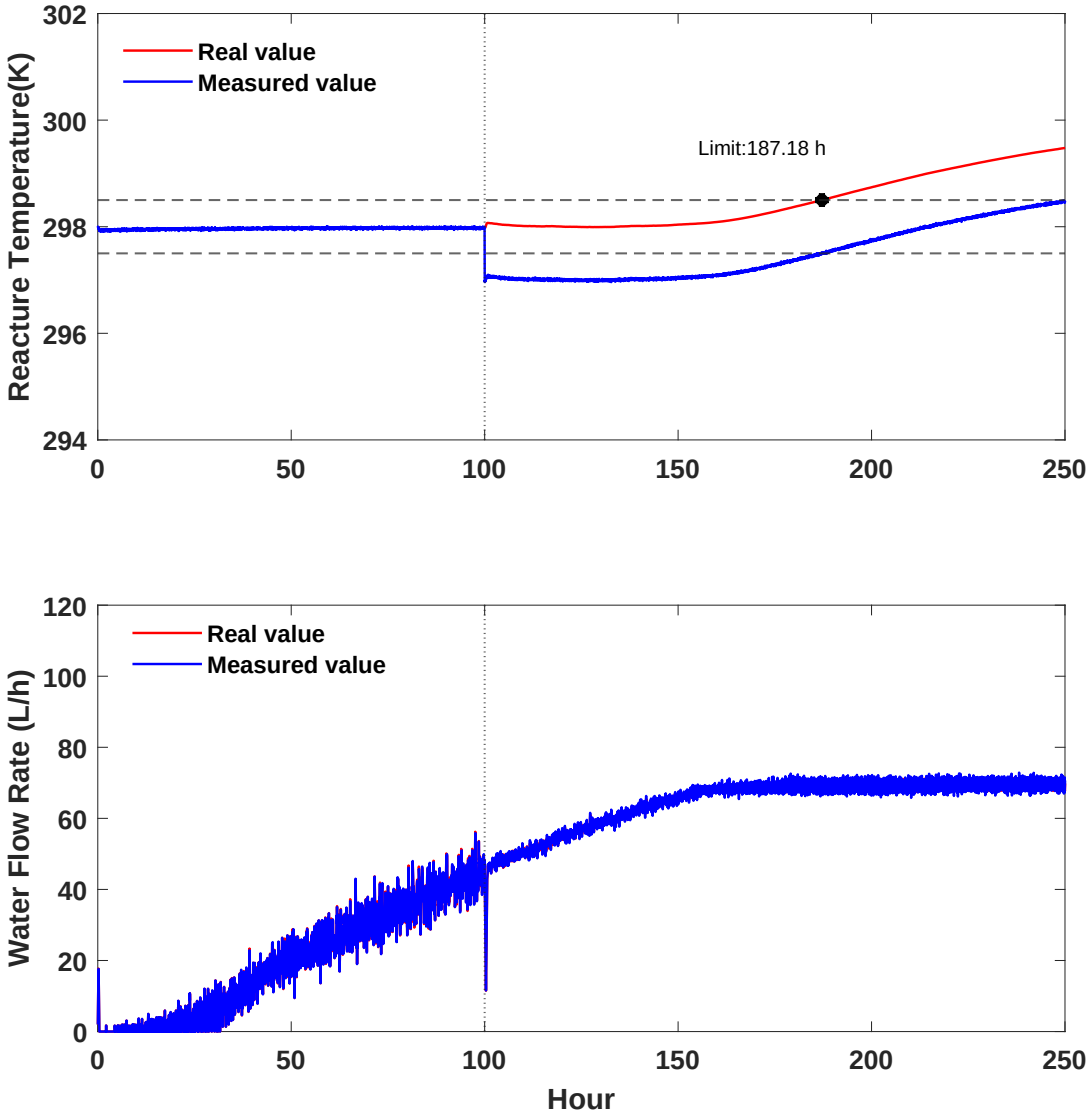


Figure A27: Reactor temperature and water flow rate profiles for process with sensor fault. Fault Onset: 100 h, Fault Magnitude: -1.0.

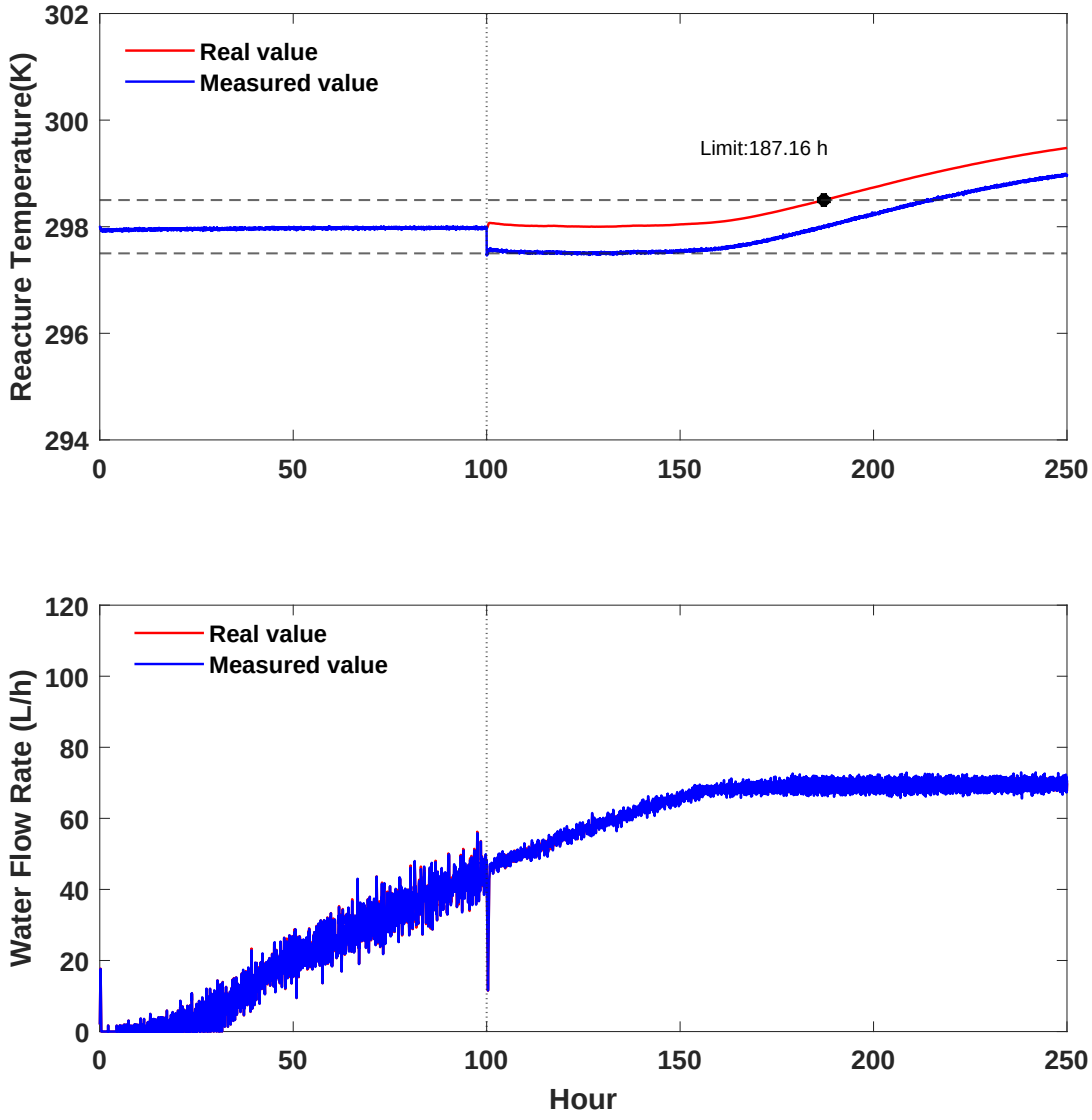


Figure A28: Reactor temperature and water flow rate profiles for process with sensor fault. Fault Onset: 100 h, Fault Magnitude: -0.5.

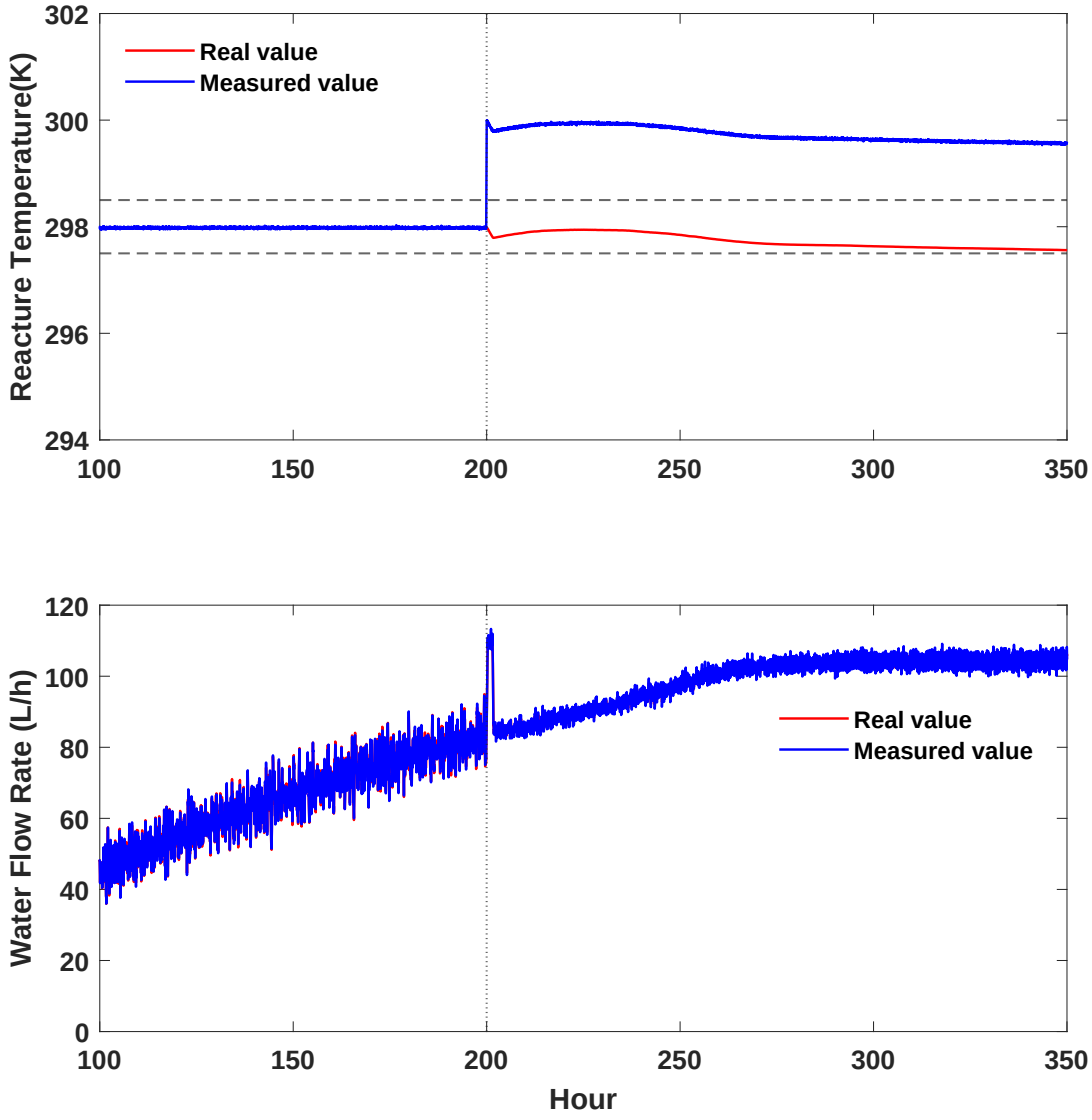


Figure A29: Reactor temperature and water flow rate profiles for process with sensor fault. Fault Onset: 200 h, Fault Magnitude: +2.0.

Set point is 298 K

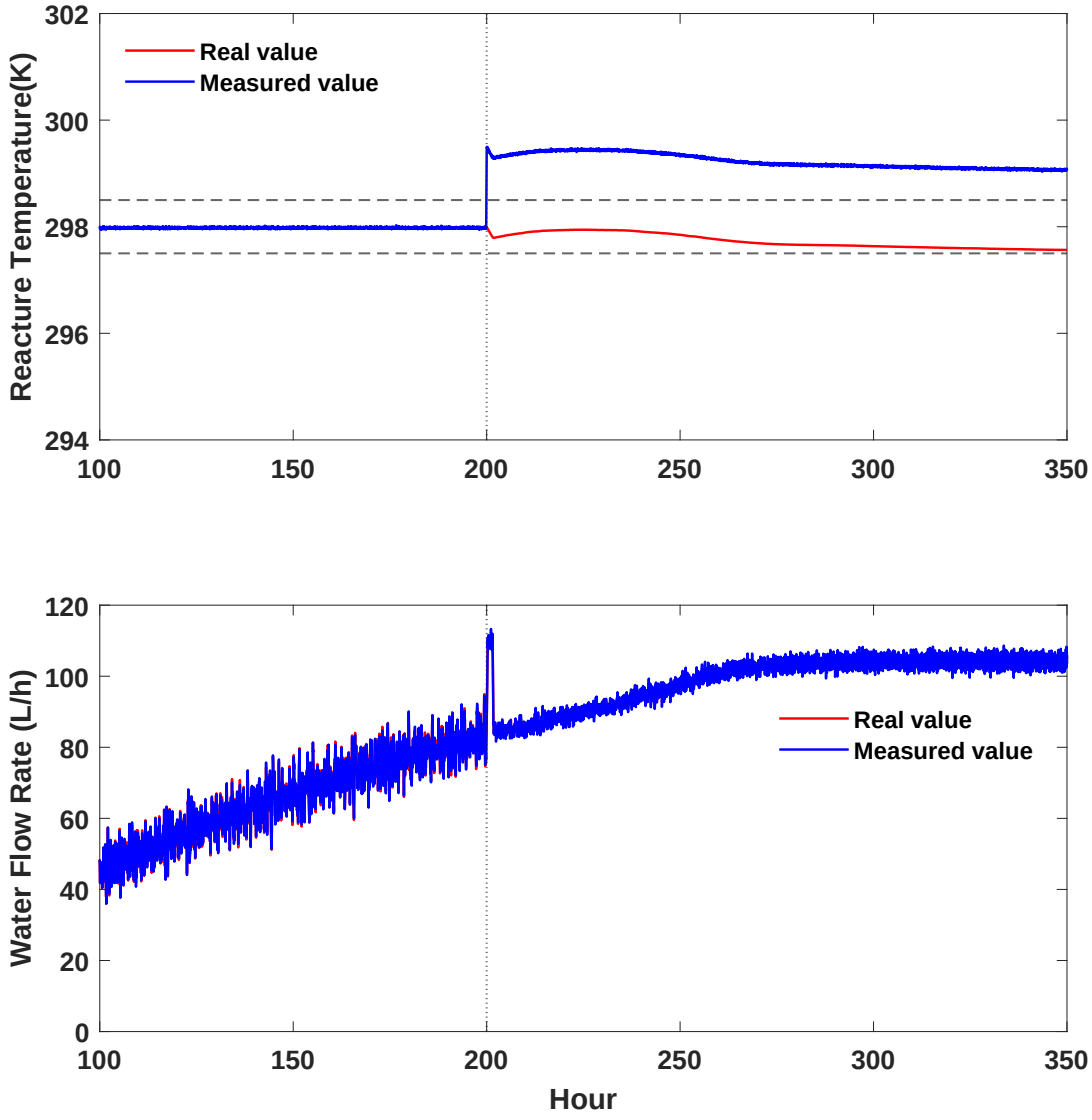


Figure A30: Reactor temperature and water flow rate profiles for process with sensor fault. Fault Onset: 200 h, Fault Magnitude: +1.5.

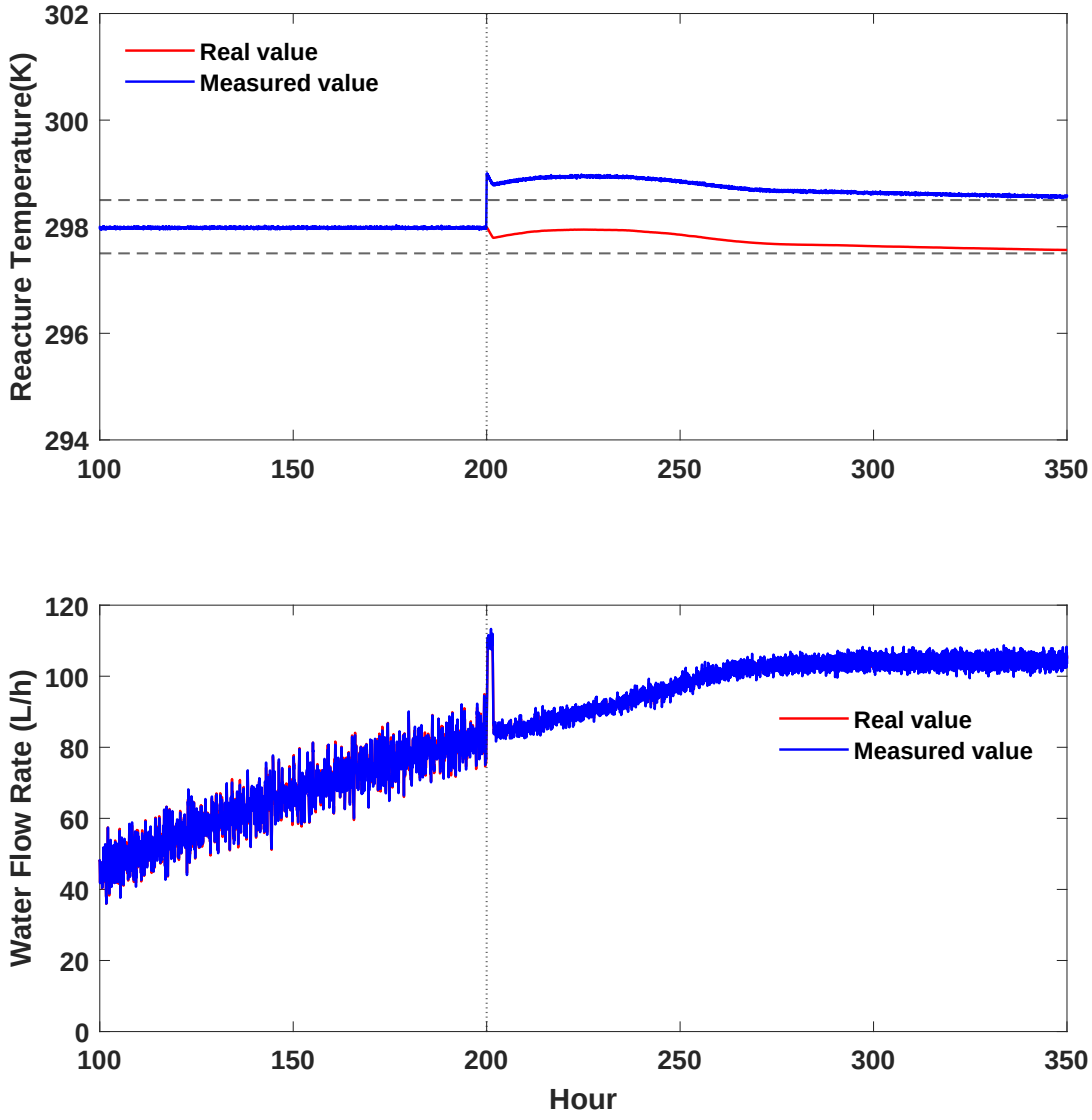


Figure A31: Reactor temperature and water flow rate profiles for process with sensor fault. Fault Onset: 200 h, Fault Magnitude: +1.0.

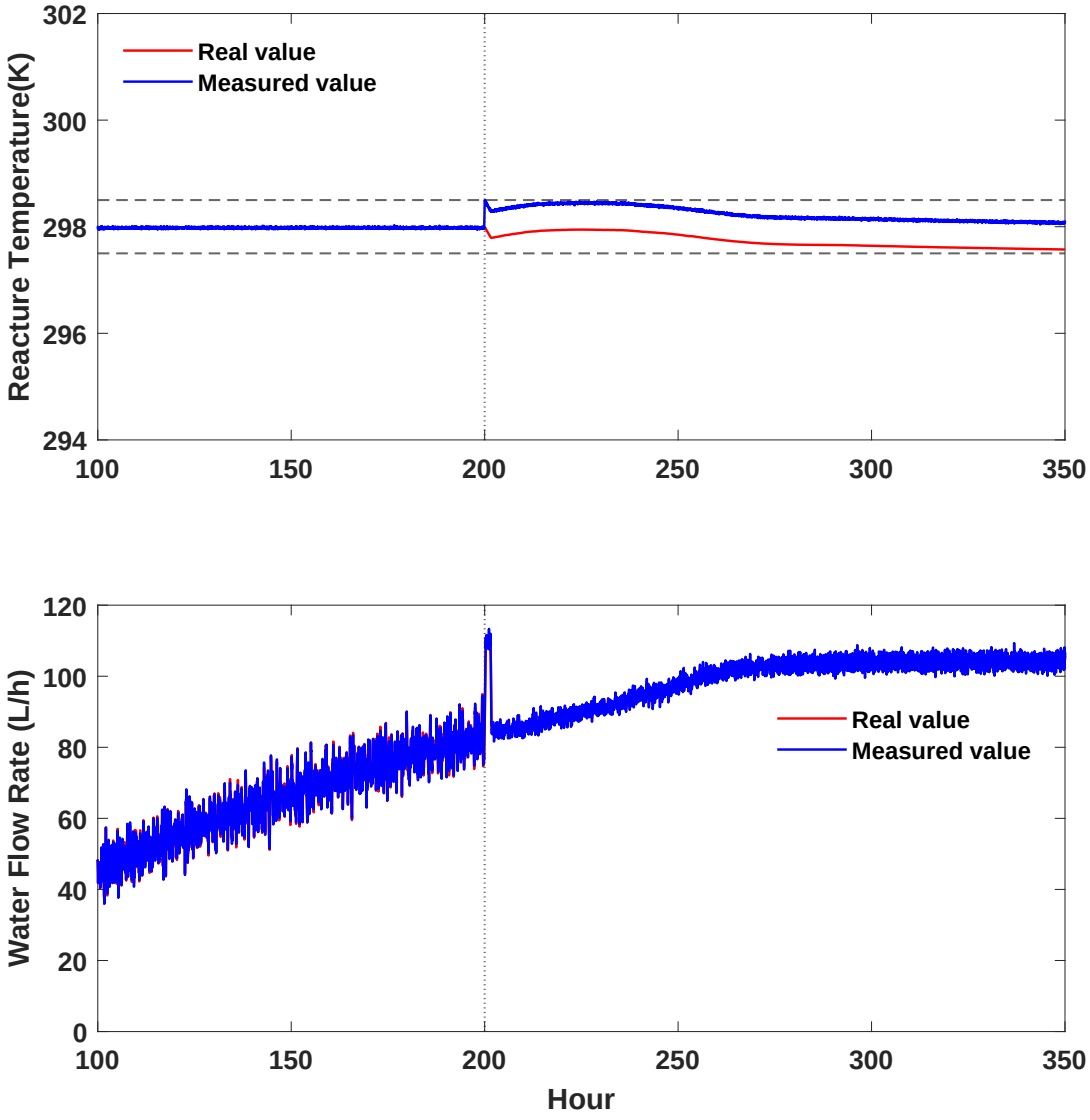


Figure A32: Reactor temperature and water flow rate profiles for process with sensor fault. Fault Onset: 200 h, Fault Magnitude: +0.5.

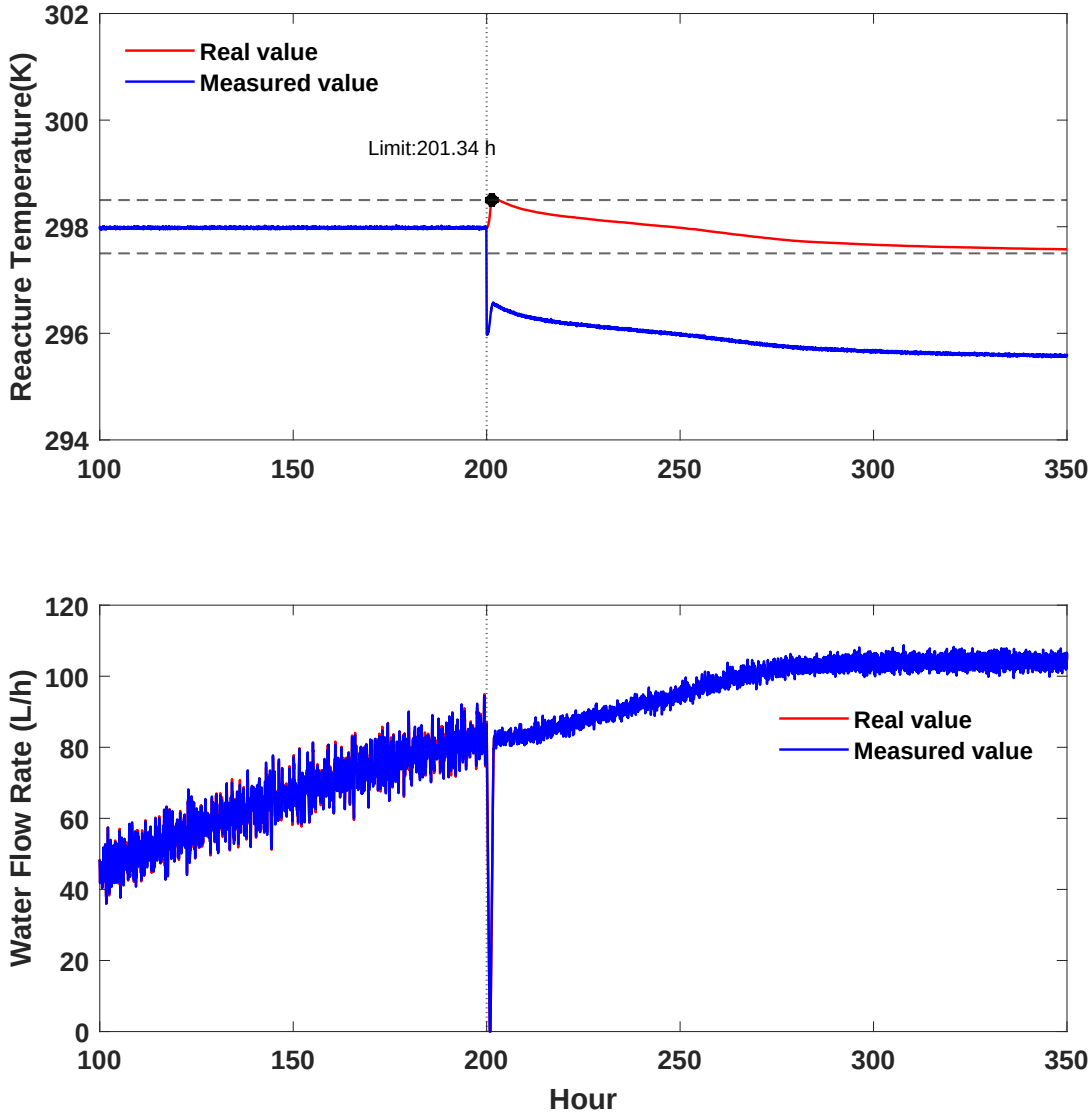


Figure A33: Reactor temperature and water flow rate profiles for process with sensor fault. Fault Onset: 200 h, Fault Magnitude: -2.0.

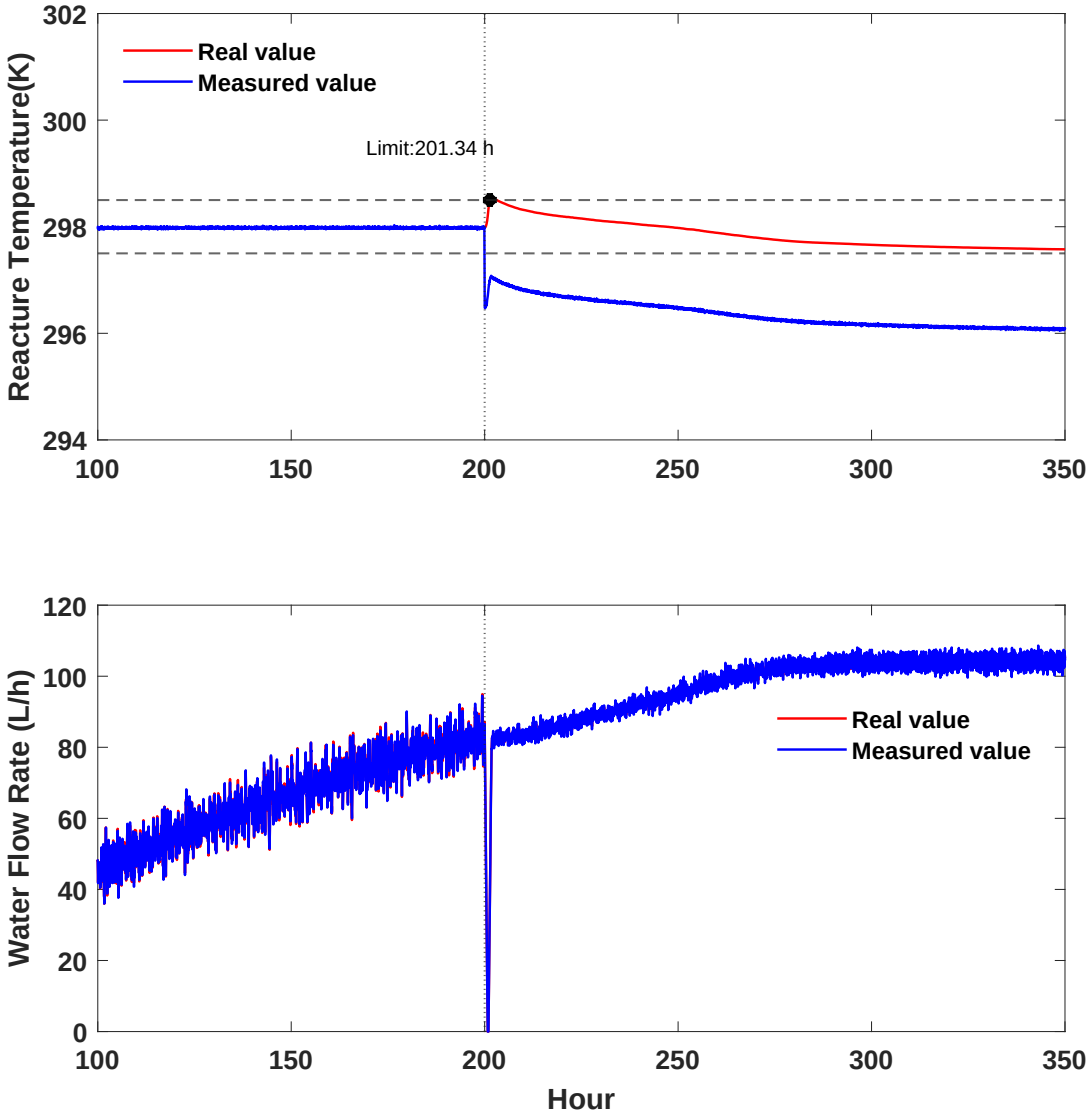


Figure A34: Reactor temperature and water flow rate profiles for process with sensor fault. Fault Onset: 200 h, Fault Magnitude: -1.5.

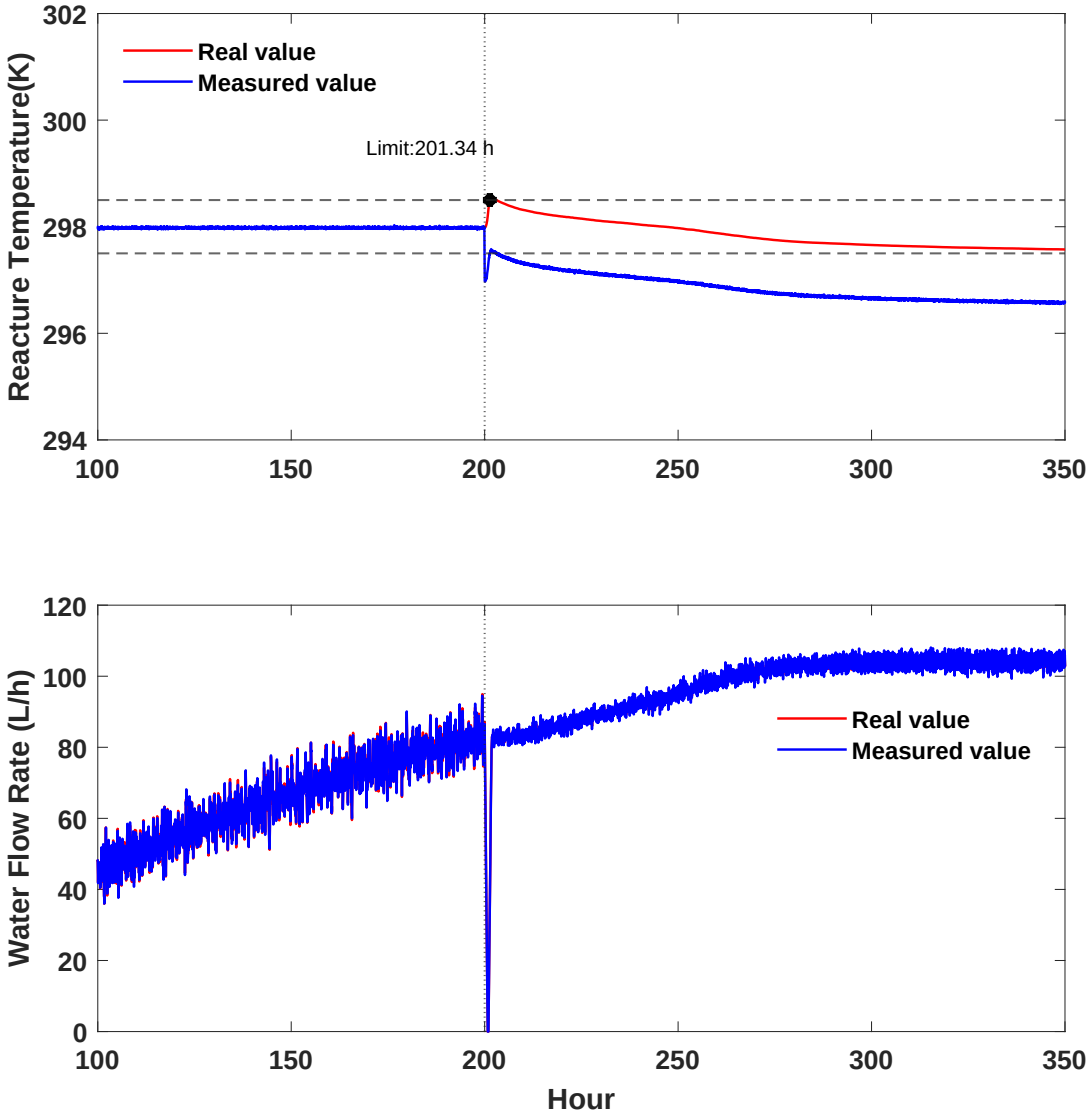


Figure A35: Reactor temperature and water flow rate profiles for process with sensor fault. Fault Onset: 200 h, Fault Magnitude: -1.0.

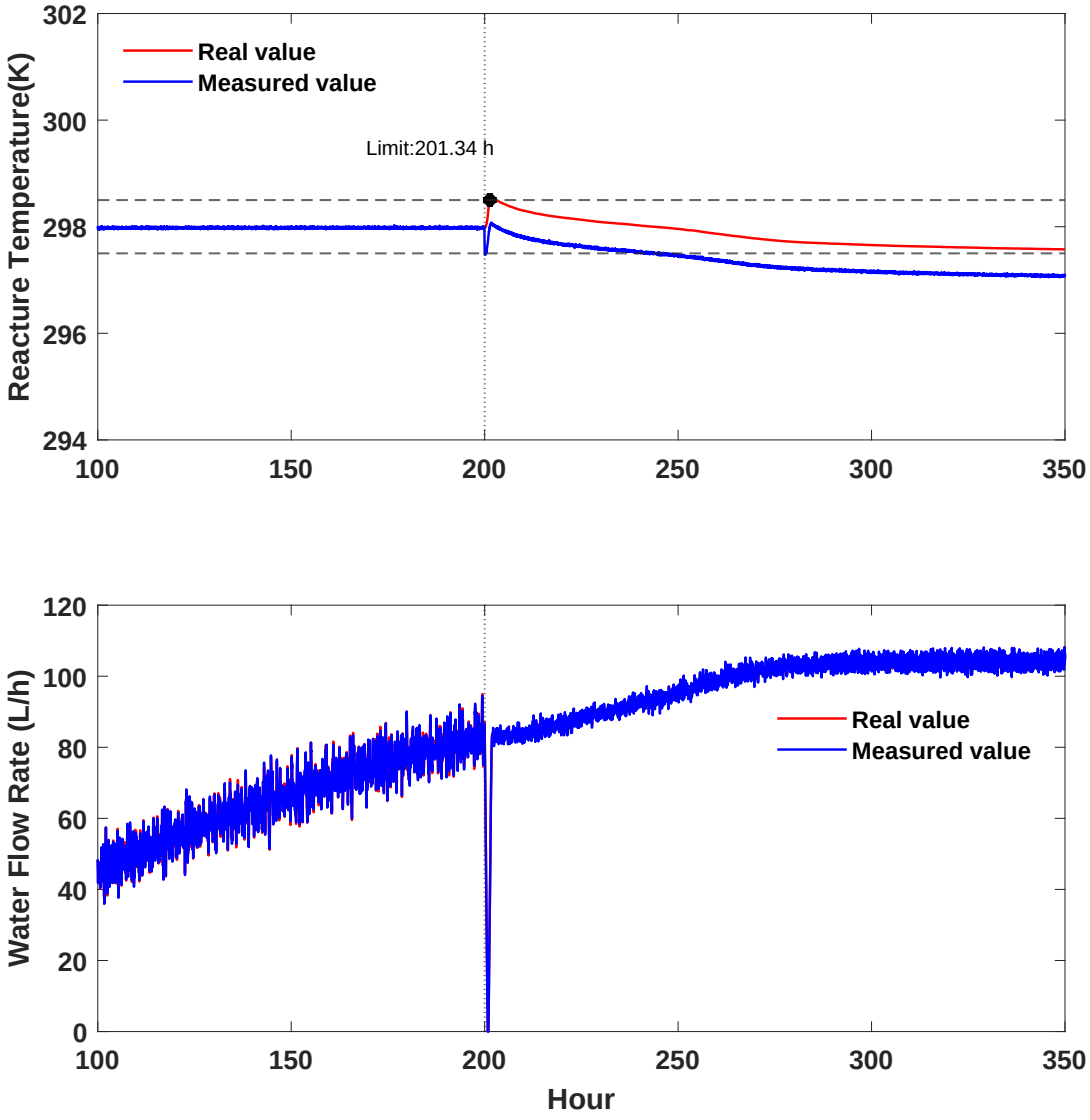


Figure A36: Reactor temperature and water flow rate profiles for process with sensor fault. Fault Onset: 200 h, Fault Magnitude: -0.5.

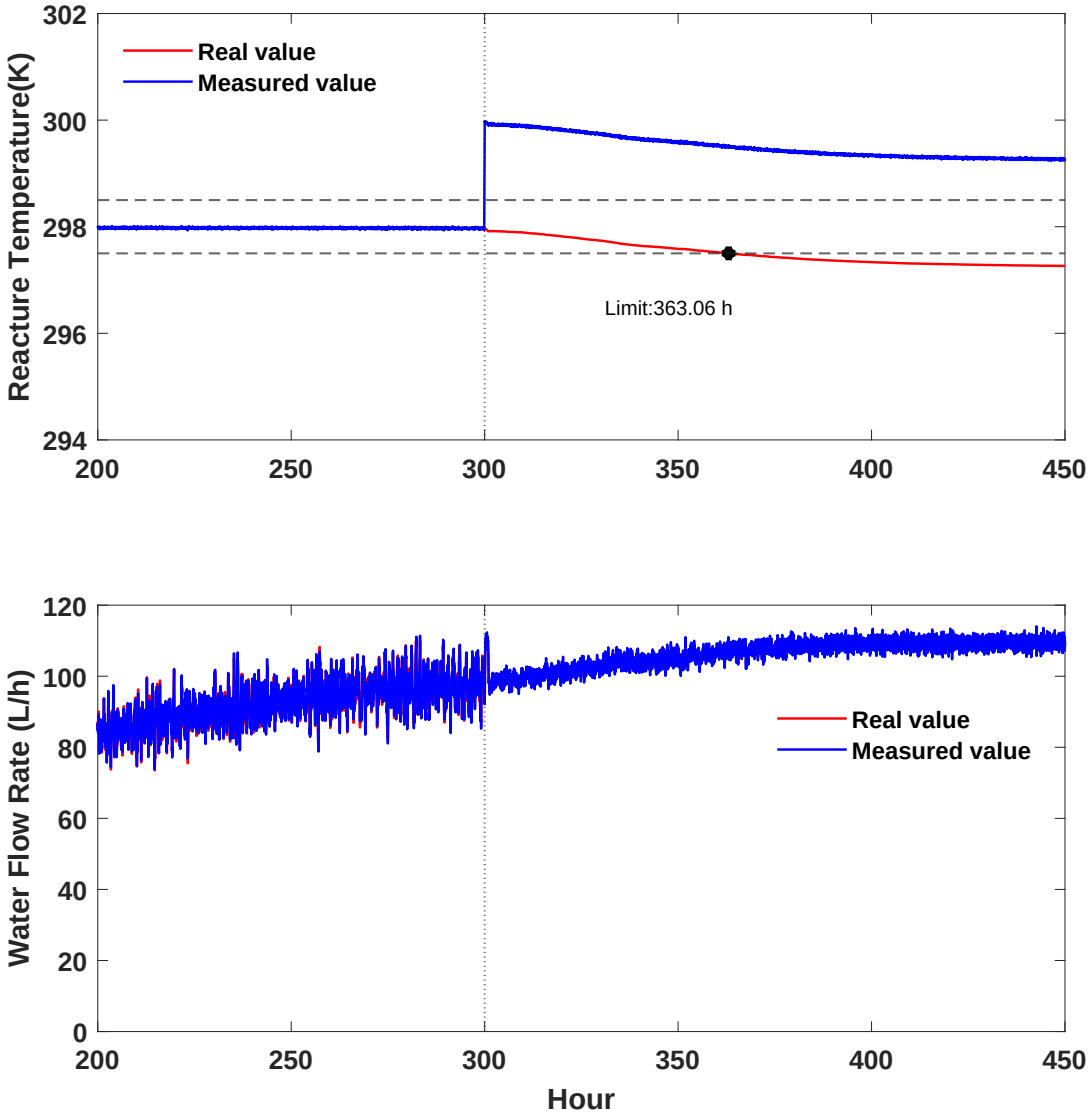


Figure A37: Reactor temperature and water flow rate profiles for process with sensor fault. Fault Onset: 300 h, Fault Magnitude: +2.0.

Set point is 298 K

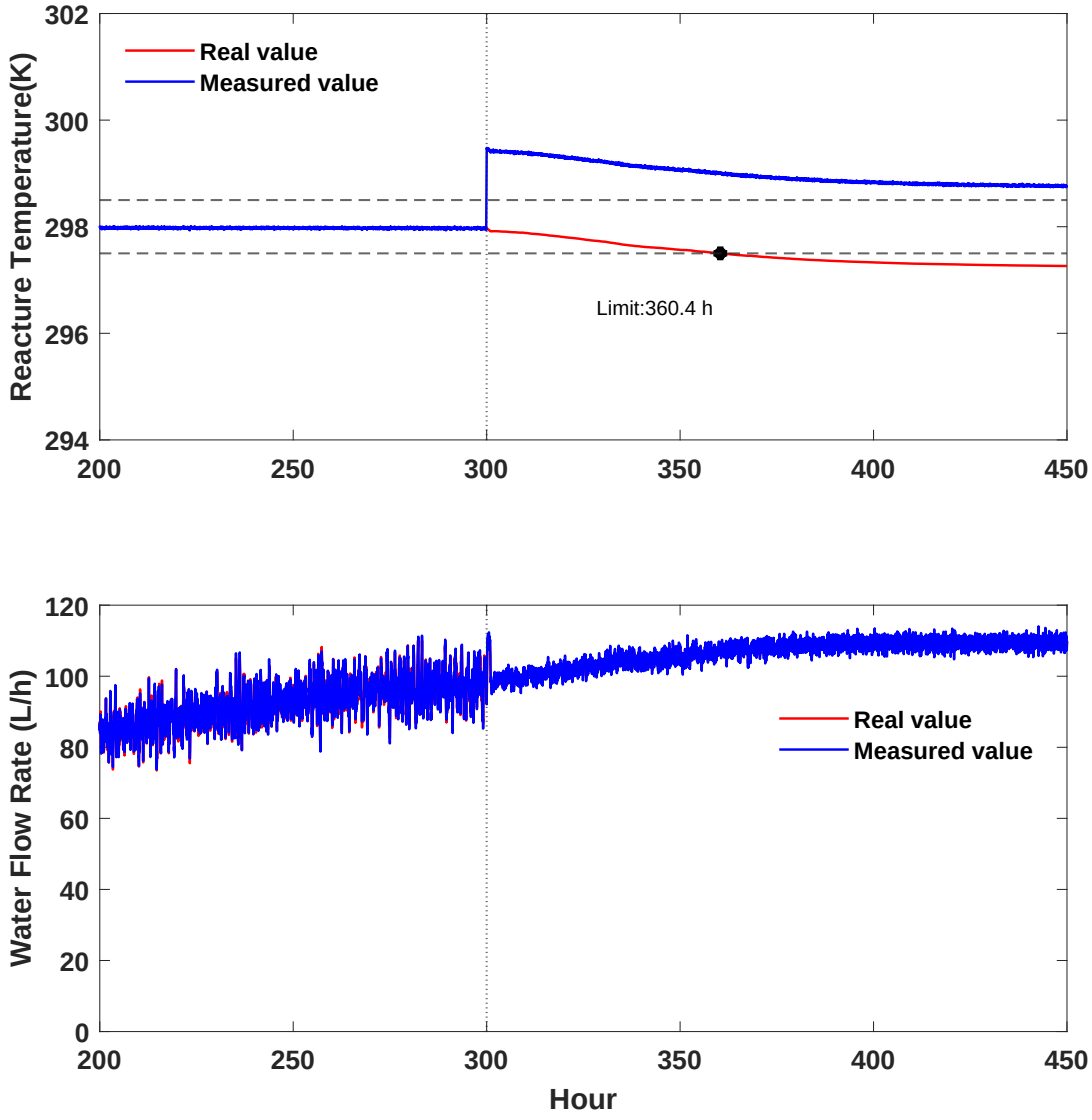


Figure A38: Reactor temperature and water flow rate profiles for process with sensor fault. Fault Onset: 300 h, Fault Magnitude: +1.5.

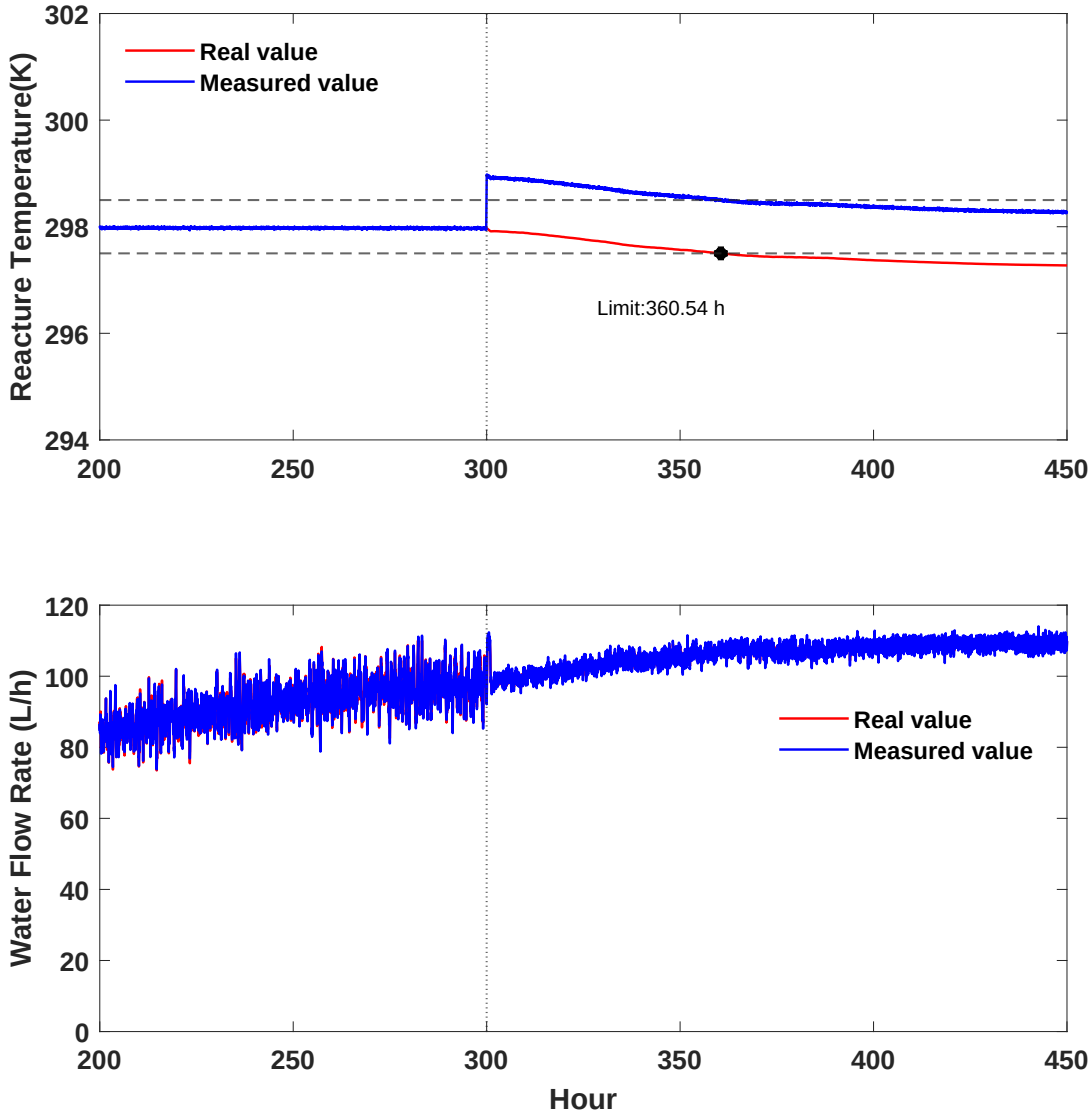


Figure A39: Reactor temperature and water flow rate profiles for process with sensor fault. Fault Onset: 300 h, Fault Magnitude: +1.0.

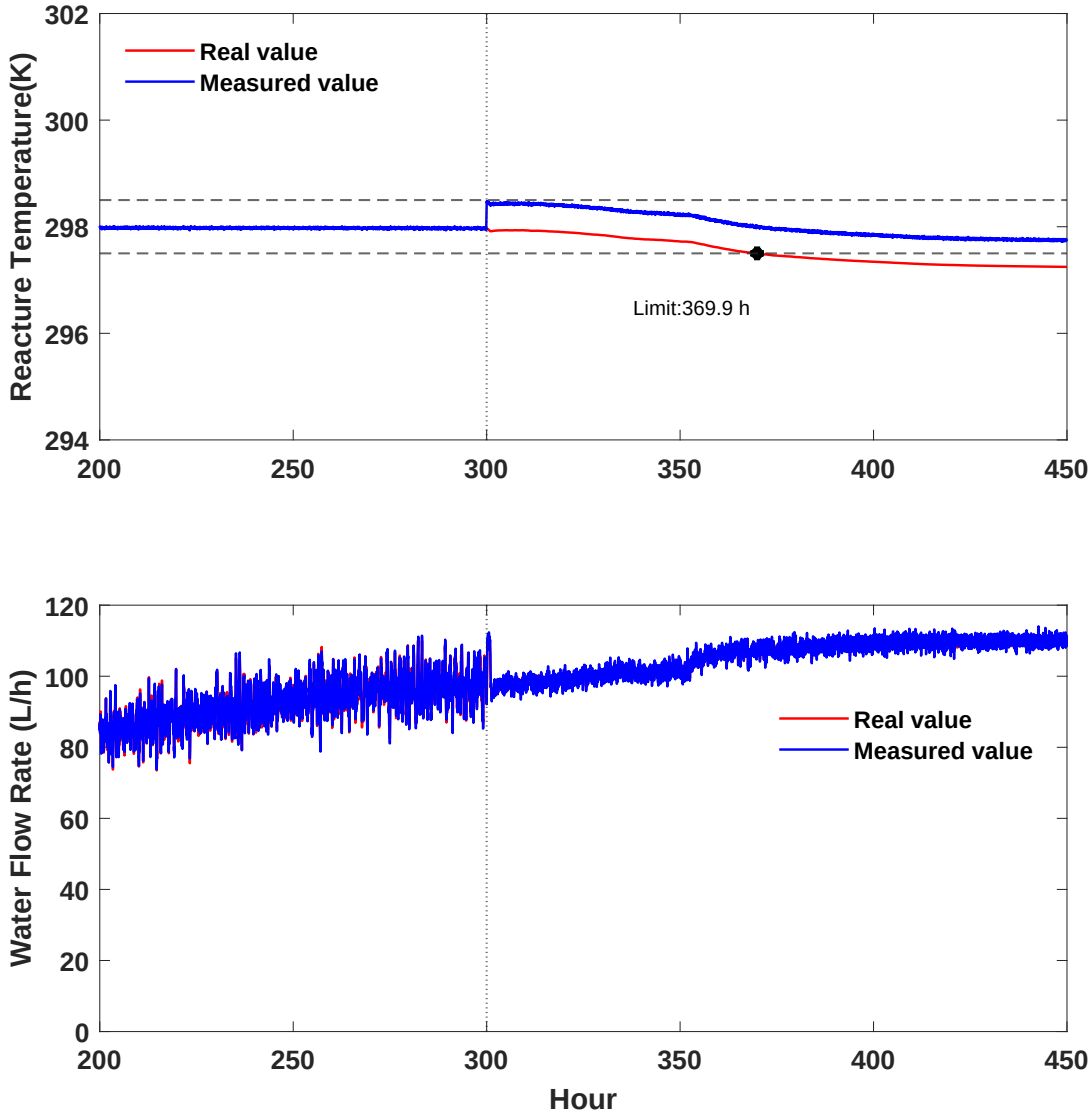


Figure A40: Reactor temperature and water flow rate profiles for process with sensor fault. Fault Onset: 300 h, Fault Magnitude: +0.5.

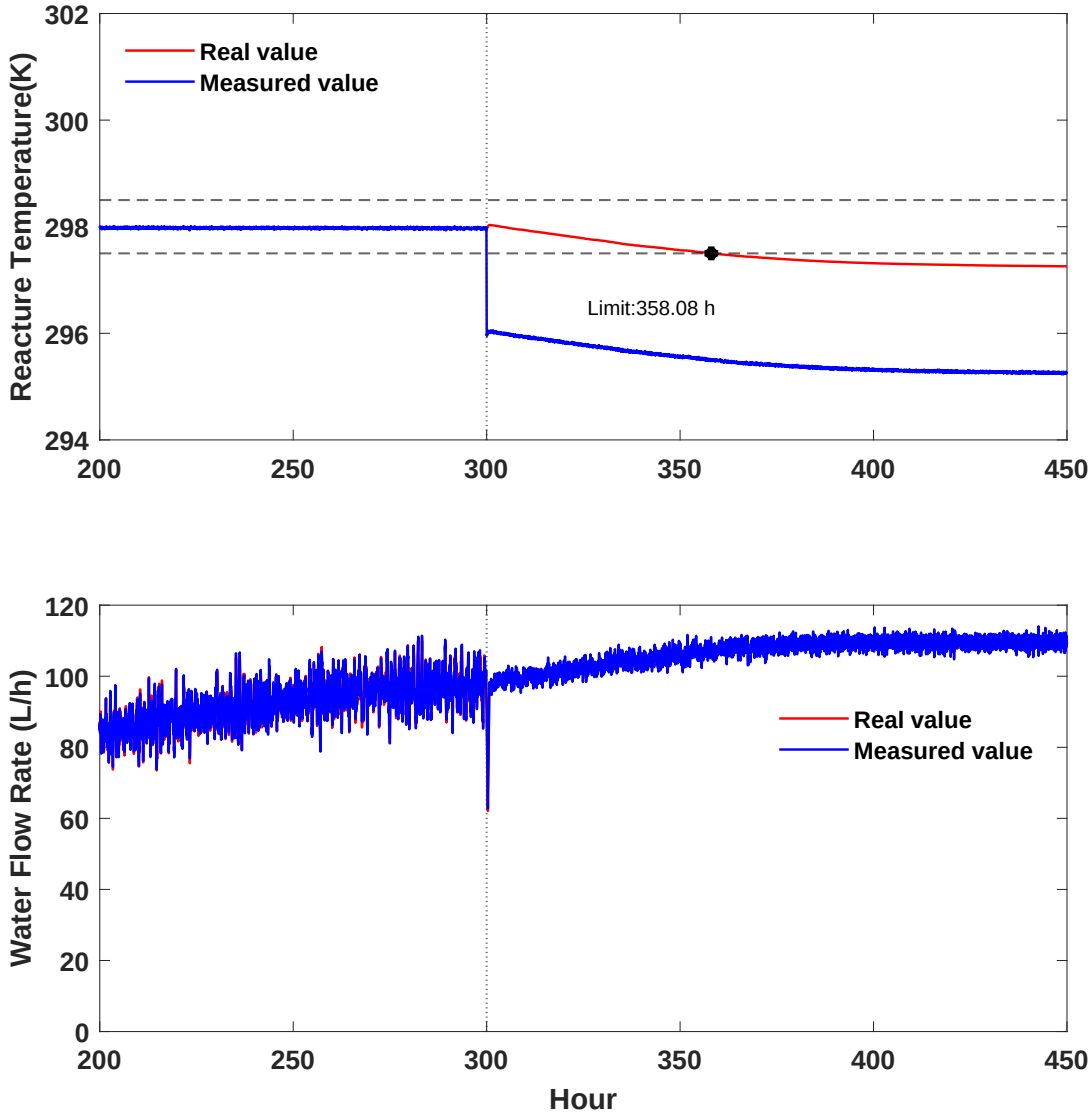


Figure A41: Reactor temperature and water flow rate profiles for process with sensor fault. Fault Onset: 300 h, Fault Magnitude: -2.0.

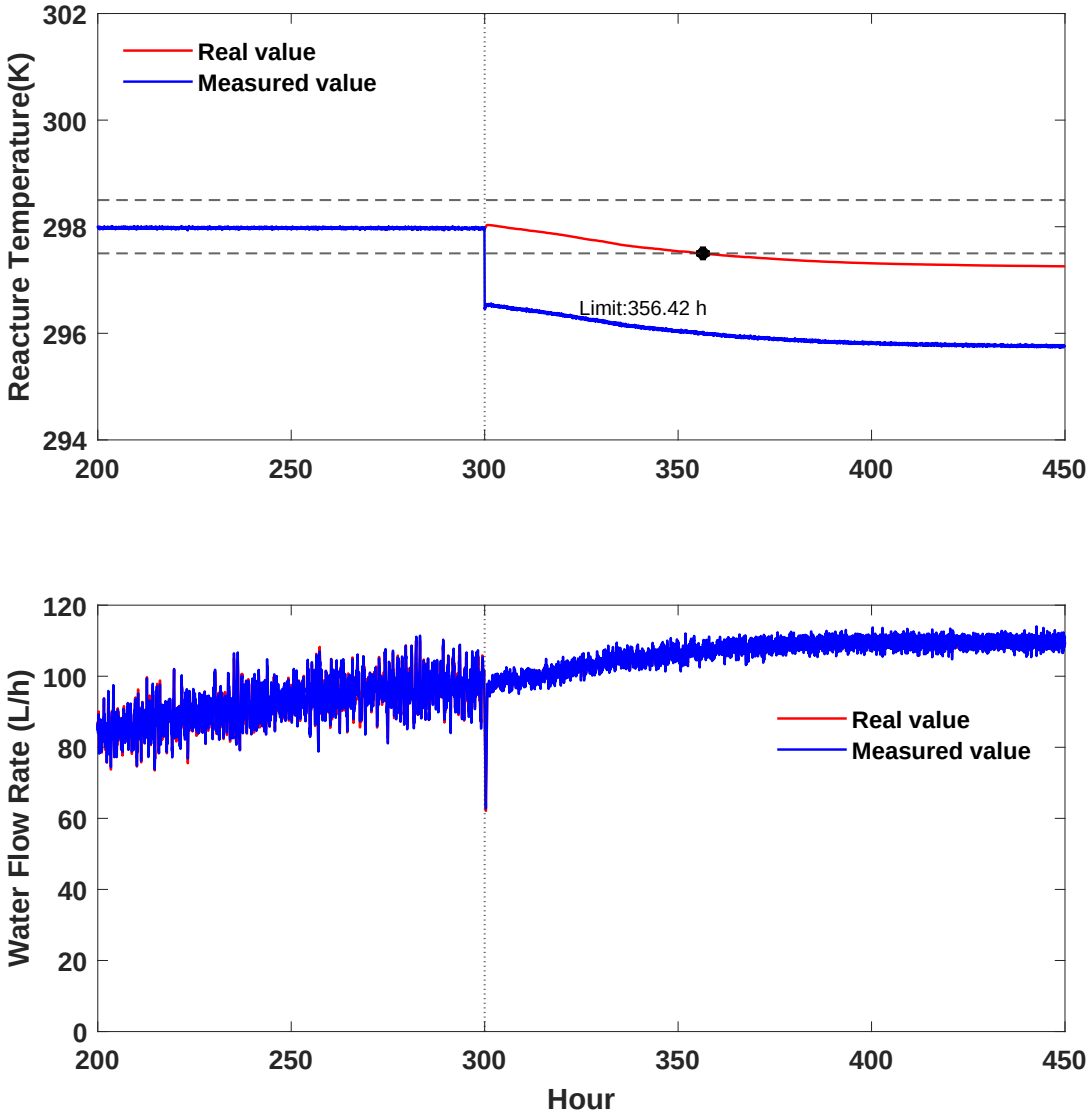


Figure A42: Reactor temperature and water flow rate profiles for process with sensor fault. Fault Onset: 300 h, Fault Magnitude: -1.5.

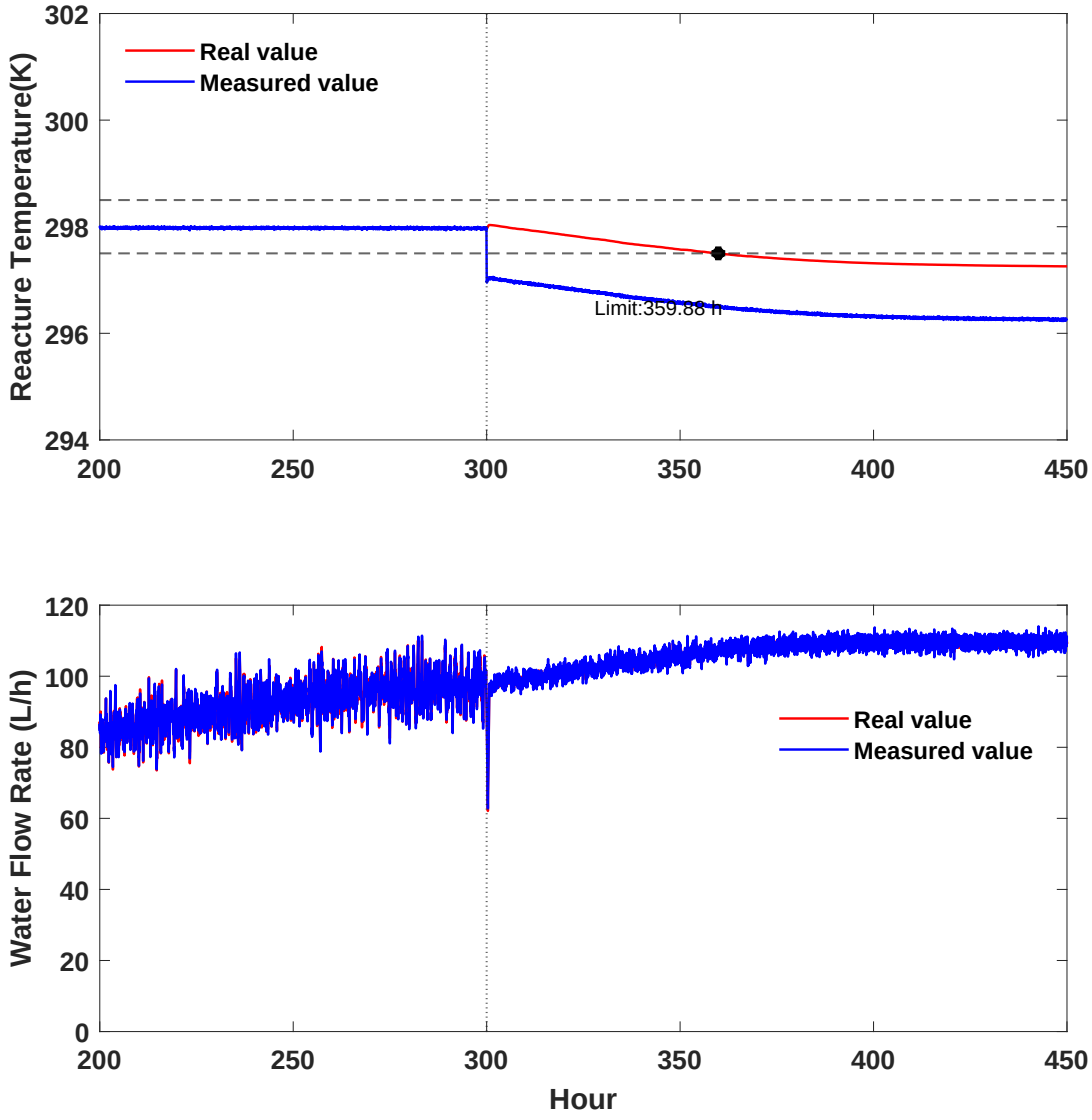


Figure A43: Reactor temperature and water flow rate profiles for process with sensor fault. Fault Onset: 300 h, Fault Magnitude: -1.0.

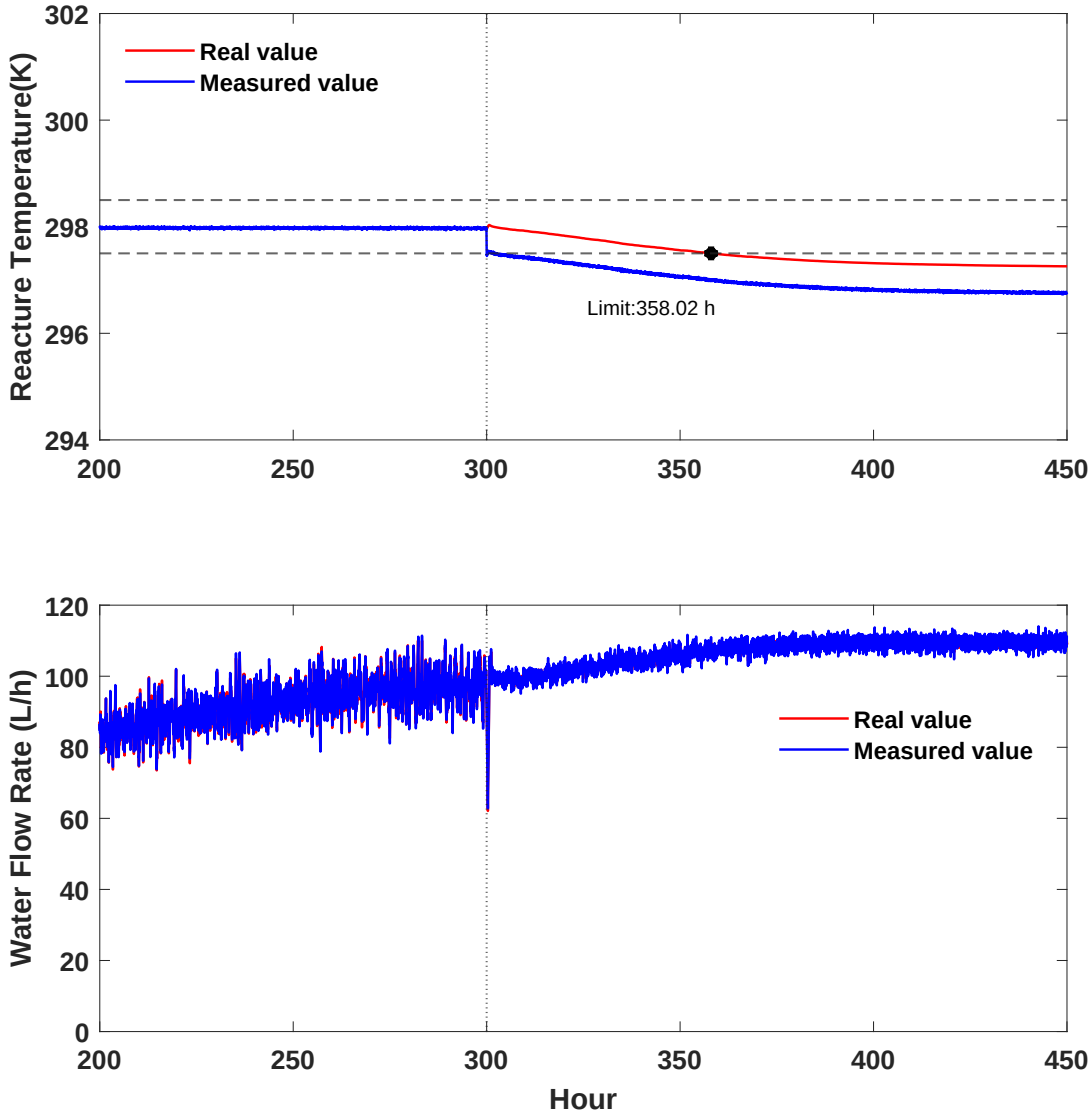


Figure A44: Reactor temperature and water flow rate profiles for process with sensor fault. Fault Onset: 300 h, Fault Magnitude: -0.5.

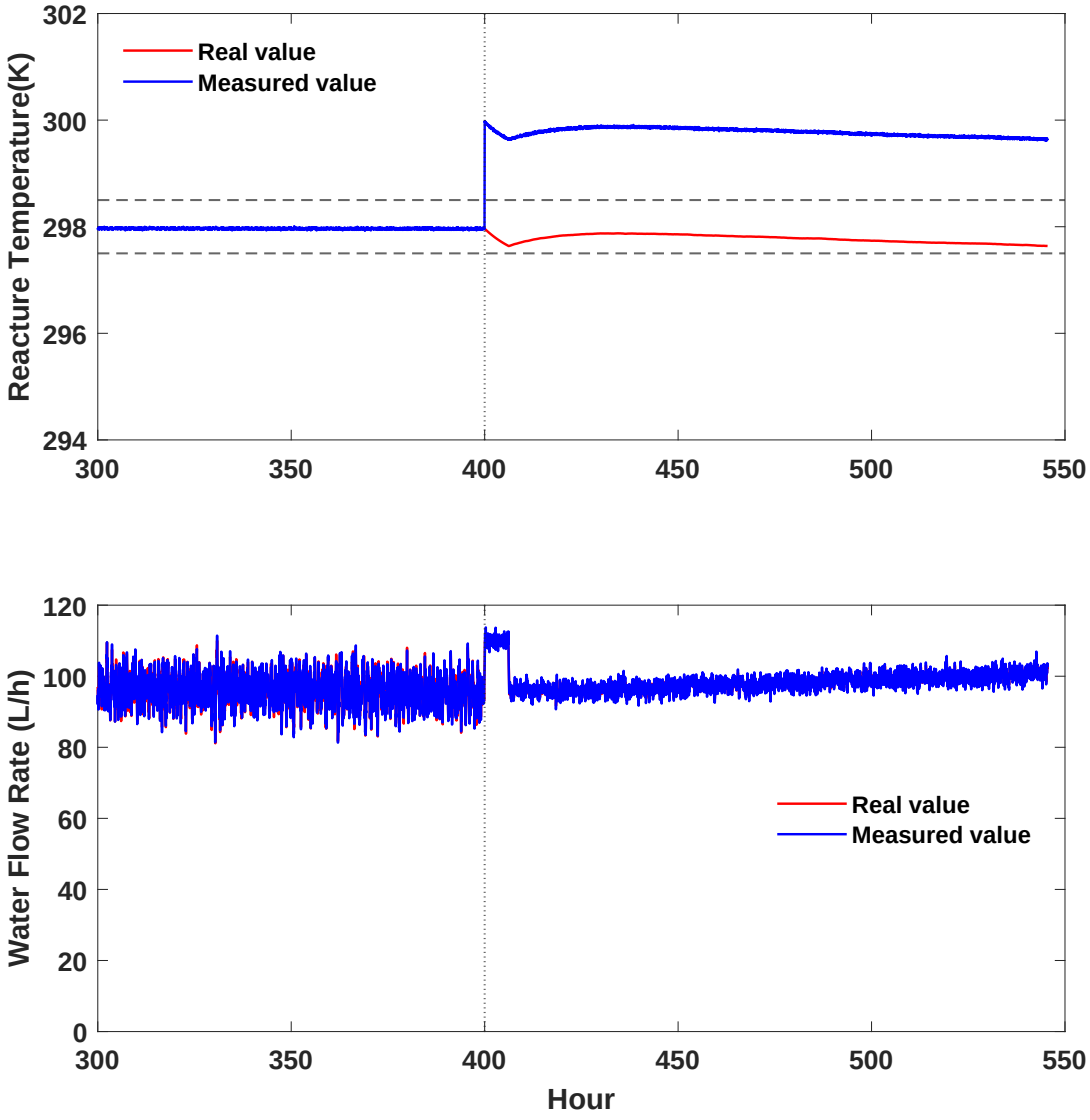


Figure A45: Reactor temperature and water flow rate profiles for process with sensor fault. Fault Onset: 400 h, Fault Magnitude: +2.0.

Set point is 298 K

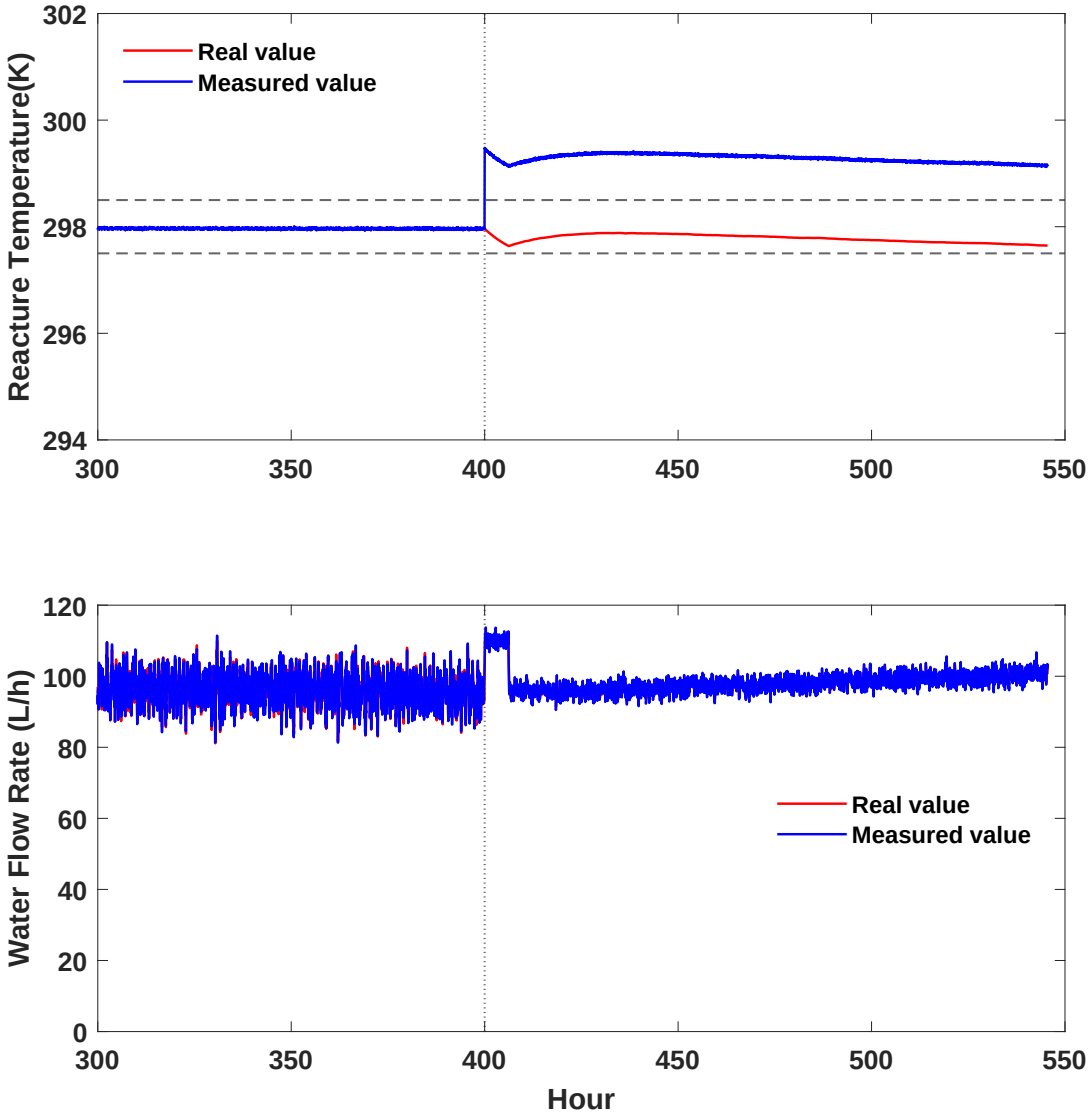


Figure A46: Reactor temperature and water flow rate profiles for process with sensor fault. Fault Onset: 400 h, Fault Magnitude: +1.5.

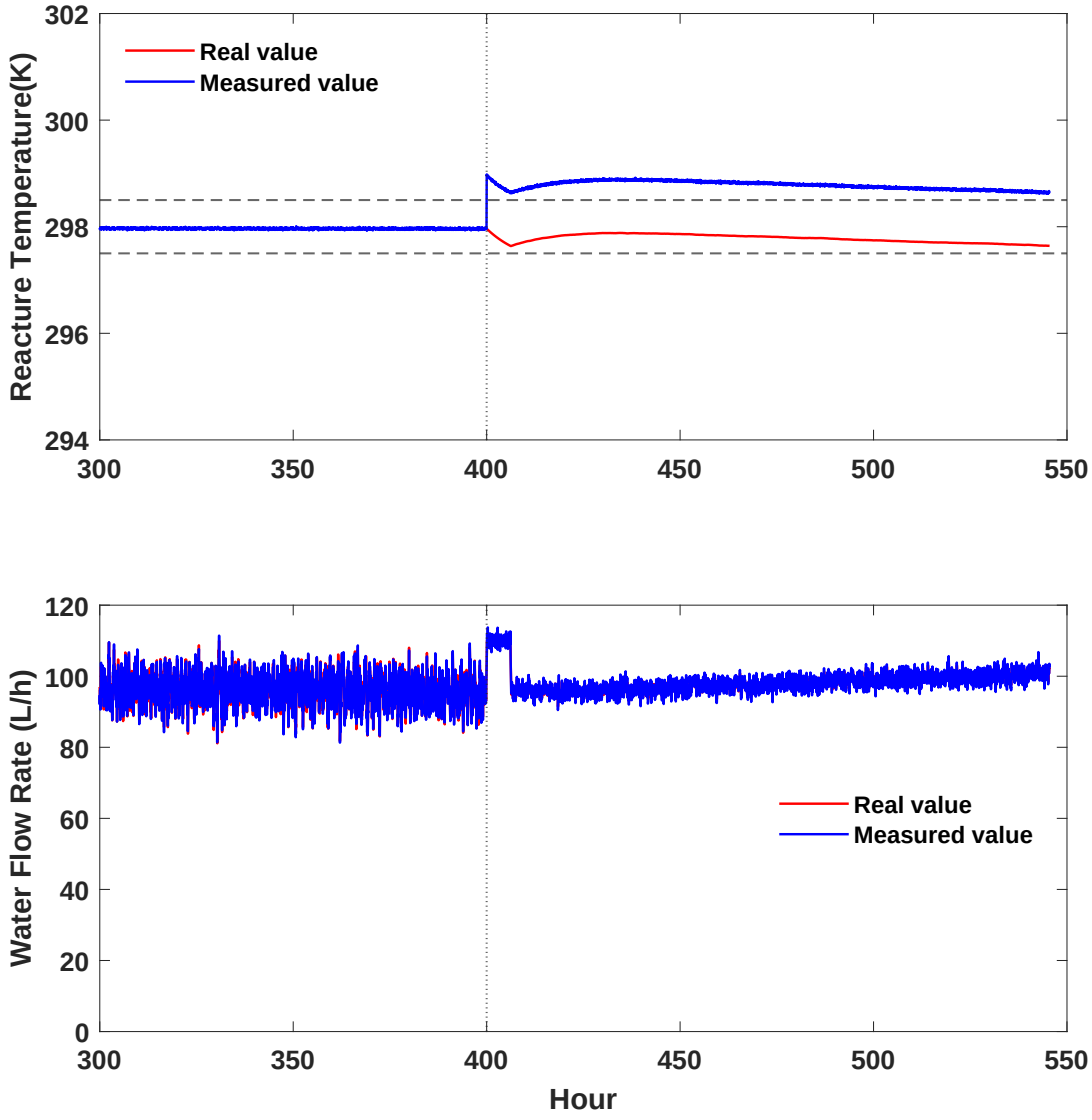


Figure A47: Reactor temperature and water flow rate profiles for process with sensor fault. Fault Onset: 400 h, Fault Magnitude: +1.0.

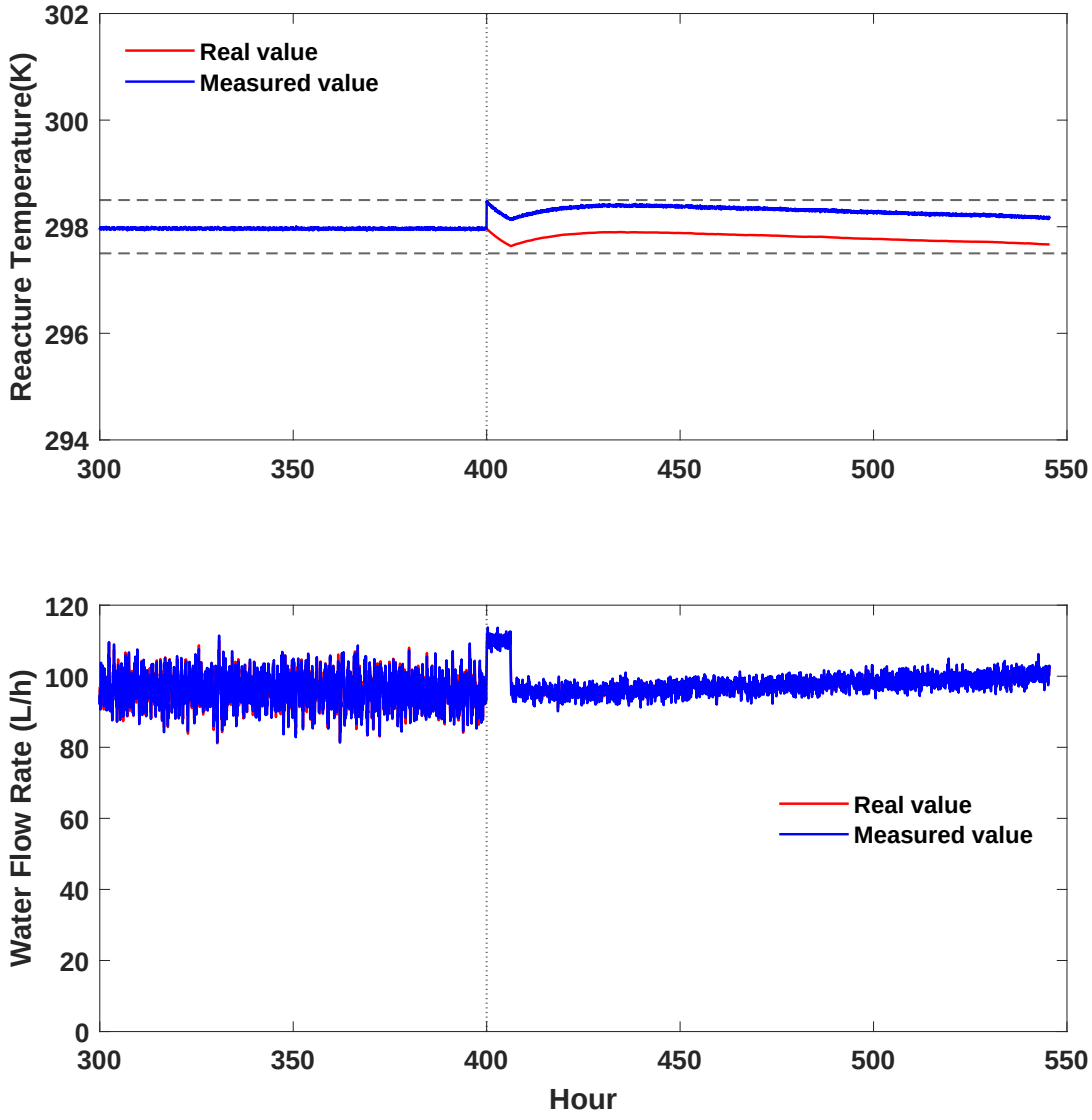


Figure A48: Reactor temperature and water flow rate profiles for process with sensor fault. Fault Onset: 400 h, Fault Magnitude: +0.5.

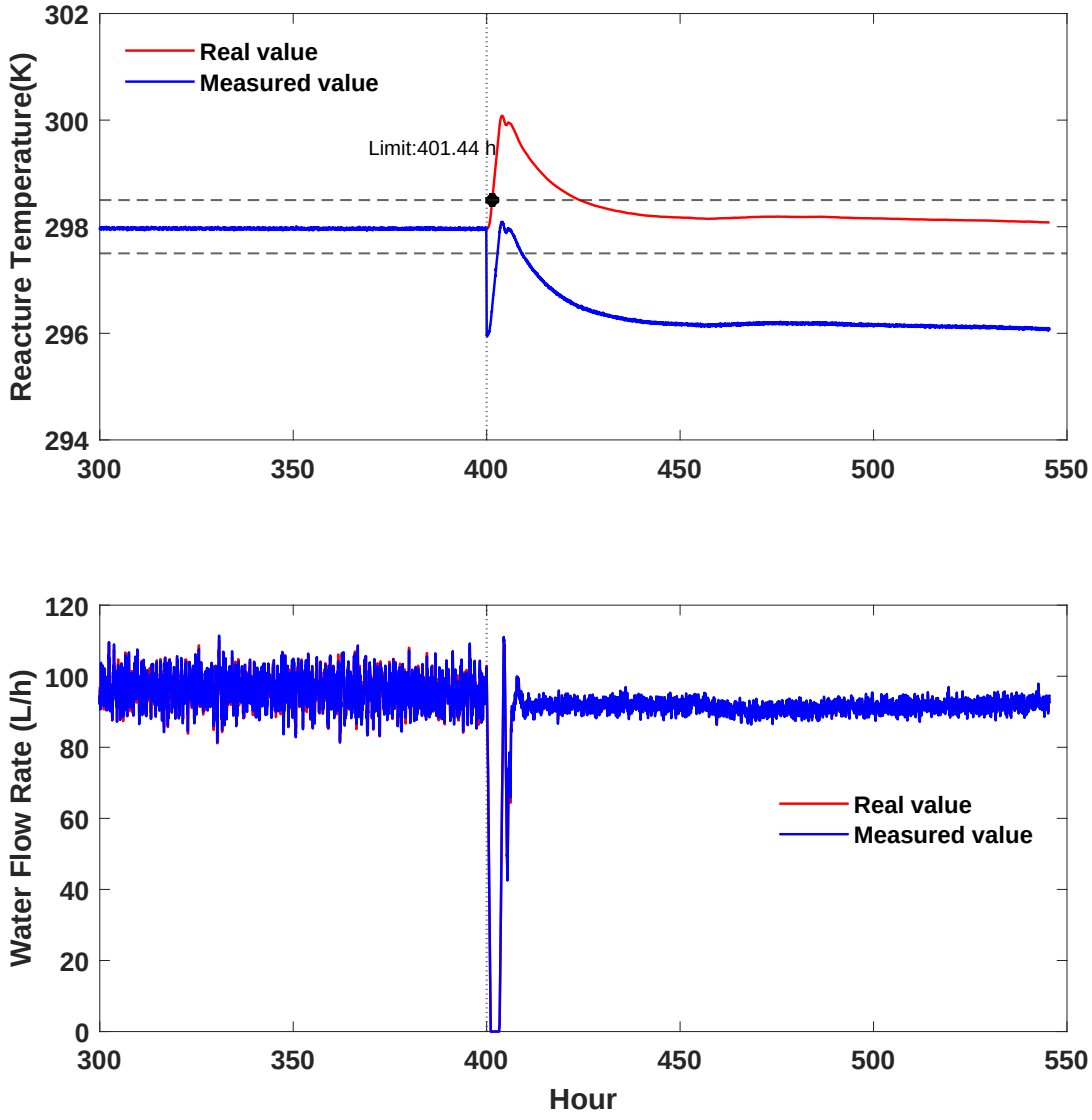


Figure A49: Reactor temperature and water flow rate profiles for process with sensor fault. Fault Onset: 400 h, Fault Magnitude: -2.0.

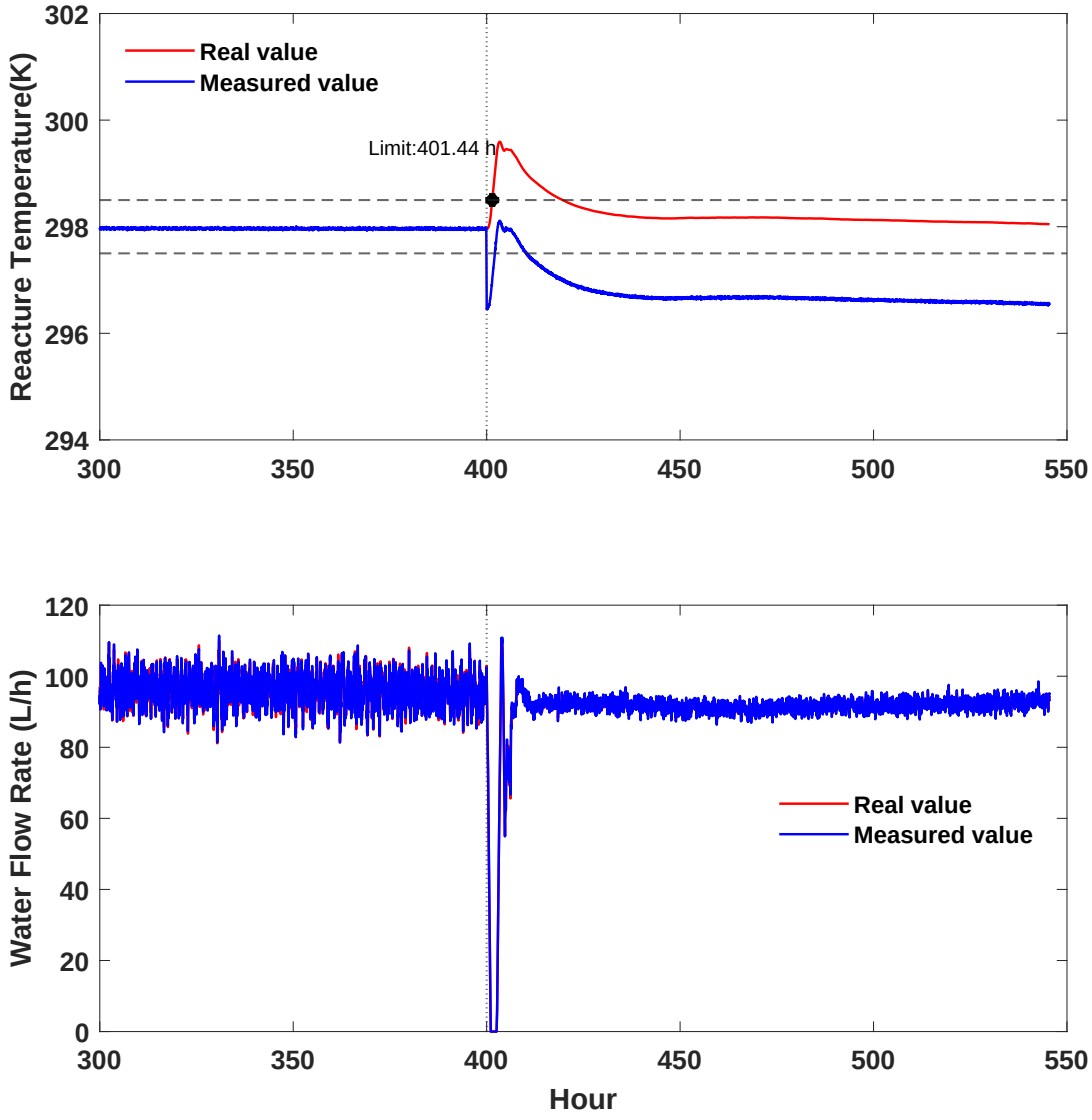


Figure A50: Reactor temperature and water flow rate profiles for process with sensor fault. Fault Onset: 400 h, Fault Magnitude: -1.5.

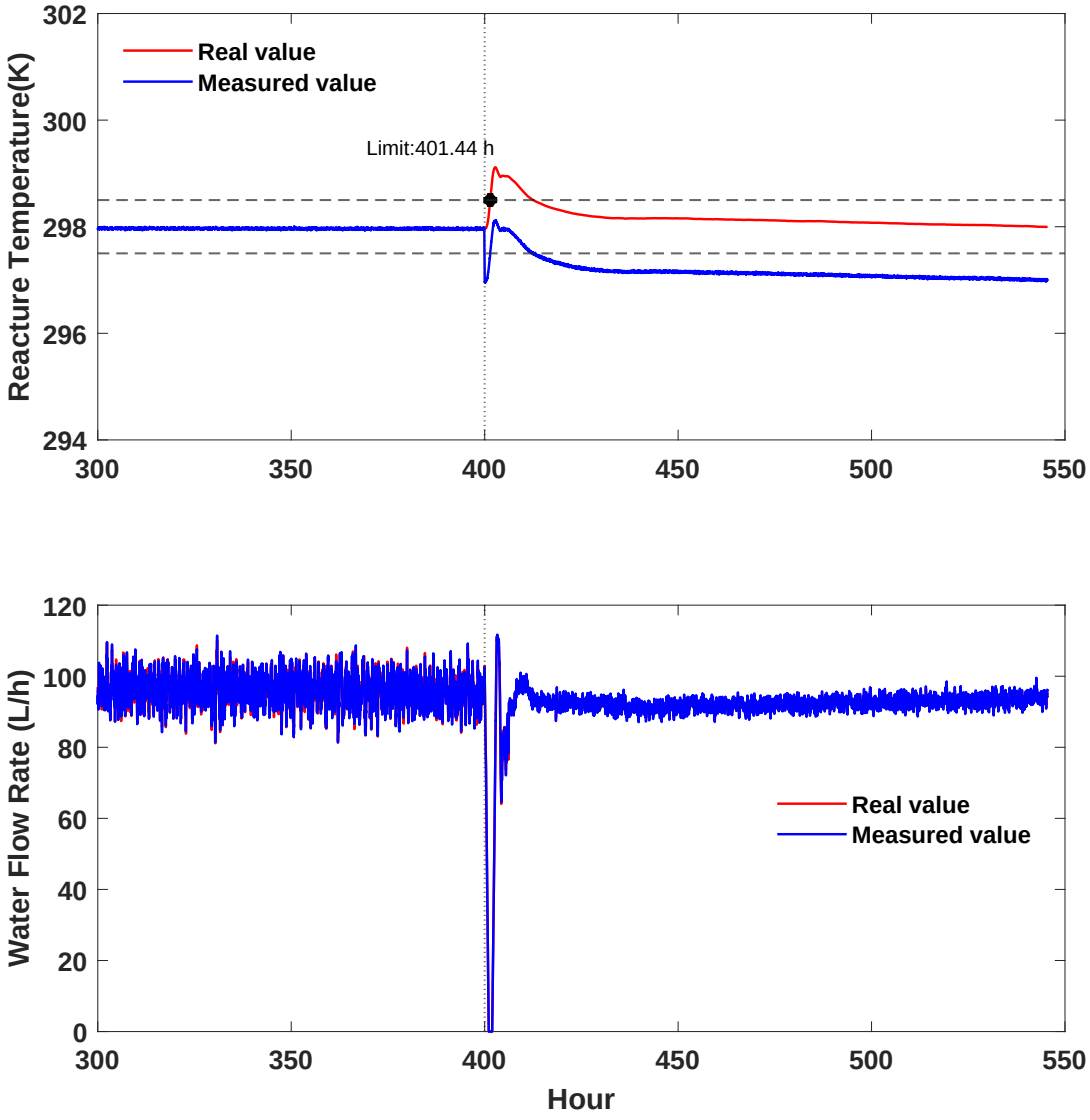


Figure A51: Reactor temperature and water flow rate profiles for process with sensor fault. Fault Onset: 400 h, Fault Magnitude: -1.0.

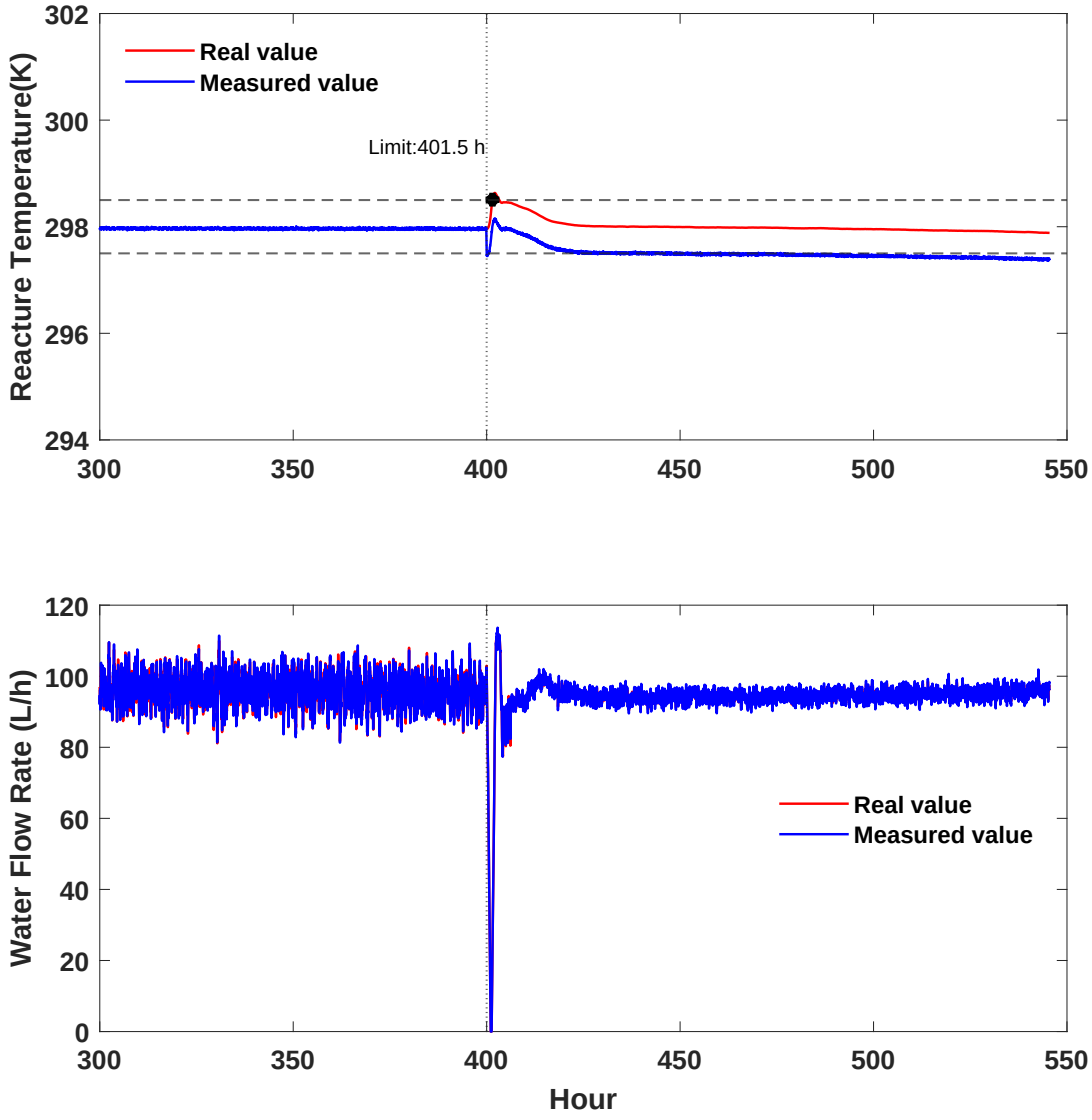


Figure A52: Reactor temperature and water flow rate profiles for process with sensor fault. Fault Onset: 400 h, Fault Magnitude: -0.5.