Towards Intelligent Control Loop Fault Diagnosis in Process Industries

A Thesis

Submitted in Partial Fulfillment of the Requirement for the Degree of

DOCTOR OF PHILOSOPHY

in

Chemical Engineering

By

Abhishek Bansal

196107101

Department of Chemical Engineering

Indian Institute of Technology Guwahati

Guwahati-781039, Assam, India

February, 2026

Department of Chemical Engineering

Indian Institute of Technology Guwahati

India

Certificate

It is certified that the work described in this thesis, entitled ***"Towards Intelligent Control Loop Fault Diagnosis in Process Industries"***, by **Mr. Abhishek Bansal**, for the award of the degree of Doctor of Philosophy, is an authentic record of the results obtained from the research work carried out under our supervision at the Department of Chemical Engineering, Indian Institute of Technology Guwahati, Guwahati, India, and this work has not been submitted elsewhere for a degree.

Signature

Prof. Prabirkumar Saha

Date: 25 May 20226

Signature

Dr. Resmi Suresh

Date: 26 May 2026

Department of Chemical Engineering

Indian Institute of Technology Guwahati

Guwahati – 781039, Assam, India

Indian Institute of Technology Guwahati

Department of Chemical Engineering

Statement

I hereby declare that the work presented in this thesis, entitled ***"Towards Intelligent Control Loop Fault Diagnosis in Process Industries"***, is the result of investigations carried out solely by me at the Department of Chemical Engineering, Indian Institute of Technology Guwahati, Guwahati, India, under the guidance and supervision of **Prof. Prabirkumar Saha** and **Dr. Resmi Suresh**.

I further declare that, in accordance with standard scientific practice, due acknowledgment has been given wherever the work presented relies on the contributions of other researchers.

Place: Guwahati

Date: 25 May 2026

Abhishek Bansal

Abhishek Bansal


Dedicated to My

Parents & Teachers


Abstract

With the rapid transition of process industries toward digitalization and automation, ensuring reliable and efficient operation of control systems has become increasingly critical. Modern industrial processes operate in complex and interconnected environments, where even minor disturbances or control loop faults can propagate across multiple units, resulting in productivity losses and quality degradation. Among these challenges, oscillations, valve stiction, and unidentified root causes of process anomalies remain persistent concerns that adversely affect process efficiency and stability. This thesis aims to address these challenges through the development of computationally efficient and interpretable approaches for intelligent fault diagnosis in process control loops.

The scope of this thesis broadly spans three areas: oscillation detection and characterization, stiction detection, and root cause analysis. The first part of this work introduces a neural network-based method for detecting oscillations using prominent features obtained from the Fast Fourier Transform (FFT) and FFT of the Autocorrelation Function (ACF) of dynamic process data. A sensitivity analysis is performed to evaluate the impact of the number of input features on the model's accuracy, precision, and recall. The proposed method achieves a reduction of up to 80% in the number of input features compared to existing techniques in the literature, thus reducing computational time without sacrificing performance. The proposed algorithm achieves a 96.63% accuracy and a recall of 0.96 for detecting oscillatory behavior. In addition, algorithms are proposed to quantify the oscillation period and the amplitude. The oscillation period is calculated based on the

frequency and amplitude obtained from the FFT of the ACF, giving an overall accuracy of 93.15% for regular and irregular oscillations. The performance of the method for predicting the amplitude of oscillation is presented for industrial data to validate its effectiveness in real-world scenarios.

Building upon this, the next part of the thesis focuses on identifying the underlying cause of oscillations, particularly those induced by valve stiction. Control valve stiction is a frequent nonlinear fault in process industries that causes stick–slip motion, inducing sustained oscillations in process variables. In this thesis, we propose a simple data-driven method for stiction detection by analyzing the dynamic interaction between the controller output (OP) and process variable (PV). The proposed method normalizes time-series signals and augments multiple time-shifted versions of signals to generate heatmaps highlighting regions of slow variations. A PV/OP heat ratio is computed from the filtered heatmaps and used as a feature for a threshold-based classifier. The classifier trained on a dataset of 58 industrial loops achieved an F_1-score of 0.696 and an area under the receiver operating characteristic curve (AUC) of 0.684. The optimal threshold obtained, when evaluated on a test set of 20 loops, achieved 95% accuracy. The method shows good generalization across different industrial loops and provides a practical and interpretable tool for valve stiction detection.

Finally, the work extends toward identifying the root cause of abnormalities in multivariable systems with interacting loops. In large chemical plants with interconnected control loops, root cause analysis (RCA) can be time-consuming due to the propagation of oscillatory and non-oscillatory faults. In this work, a topology-independent RCA method is proposed based on cross-correlation with weighted lags among process variables to identify the key variables responsible for anomalies in the system. The proposed method introduces the τ-metric, which incorporates the principle that a cause precedes its effect. The τ-metric systematically evaluates cross-correlations across all pairwise combinations of process variables, identifying the lag at which each pair reaches maximum correlation. By combining both

correlation strength and lag information into a weighted average, the τ-metric effectively pinpoints the variable that consistently leads others, thereby indicating the potential root cause of the fault. The proposed algorithm is validated using synthetic data generated in MATLAB Simulink for various processes with interconnected control loops. The study's results showed an overall accuracy of 94.41% for non-oscillatory faults and 88.65% for oscillatory faults for the case studies considered. Furthermore, the algorithm is also validated on synthetic data generated from two industrial case studies to exemplify the practical relevance of the approach.

Overall, this thesis advances intelligent control loop fault diagnosis in process industries through the development of data-driven, interpretable, and computationally efficient methodologies. The proposed approaches for oscillation detection, valve stiction identification, and root cause analysis collectively contribute toward improving process reliability and enhancing control loop performance. By addressing key challenges in monitoring and diagnosis under diverse operating conditions, this work provides practical and robust tools for achieving more reliable and autonomous process operations.

Acknowledgements

This doctoral journey would not have been possible without the guidance, support, and encouragement of many people who have accompanied me along the way. I am deeply thankful to all of them.

First and foremost, I owe my deepest gratitude to my supervisors, **Prof. Prabirkumar Saha** and **Dr. Resmi Suresh**, Department of Chemical Engineering, whose guidance, support, and encouragement have been central to the successful completion of this thesis. Their wisdom, patience, and dedication created an inspiring environment that constantly motivated me to give my best. They were always approachable and supportive, generously giving their time to address my doubts, provide constructive feedback, and refine my ideas. Their discipline, clarity of thought, and passion for research have profoundly shaped my approach to scholarship and instilled in me values that will guide me throughout my career. Their unwavering encouragement kept me motivated during the most challenging phases of my research, and it has been a privilege to work under their mentorship. I will remain forever grateful for the invaluable lessons I have learned from them.

I would like to sincerely thank my doctoral committee members, **Prof. V. S. Uppaluri Ramagopal**, **Prof. S. Senthilmurugan**, and **Prof. Harshal B. Nemade**, for their valuable suggestions and constructive comments during the various assessments of my Ph.D. program. I am also grateful to my comprehensive examination committee members, **Prof. Tamal Banerjee** and **Prof. Bishnupada Mandal**, for their insightful feedback and guidance, which helped me strengthen the foundation of my research.

I am thankful to the Heads of the Department of Chemical Engineering during the course of my doctoral program, **Prof. Anugrah Singh**, **Prof. Kaustubha Mohanty**, and **Prof. Subrata Kumar Majumder**, for their administrative support and encouragement. I would also like to extend my gratitude to all the faculty members, research scholars, and supporting staff of the Department of Chemical Engineering, IIT Guwahati, for their kind cooperation and assistance in various aspects throughout my research journey.

I gratefully acknowledge the Ministry of Human Resource Development (MHRD), Government of India, and the Indian Institute of Technology Guwahati for providing the financial support and fellowship throughout the duration of my Ph.D. program.

Special thanks to academic people, library facilities, chemical engineering department staff, security staff, and Core I-IV people for their necessary support in every aspect. I would like to acknowledge the Disang hostel, IIT Guwahati, for becoming almost my second home during my doctoral journey.

I would also like to express my heartfelt gratitude to Dr. Haribabu K, Department of Chemical Engineering, NIT Calicut. He was the one who first recognized my potential and inspired me to pursue research. I am especially thankful to my friends during my doctoral journey, Dr. Sapunii Sebastian, Dr. Nipu Kumar Das, Dr. Anil Kumar, Chandra Bhan, Plaban Jyoti Buragohain, Evenmore Mylliem, Antash Kishore Sinha, Suraj Goala, and Rigved Samant, whose companionship and support made this journey truly memorable. I am equally grateful to my friends, Manmohan Prasad, Samrat Dey, Pratyasha Priti Das, Piush Agarwal, Yash Agarwal, Akash Agarwal, Sandeep Singh, Hemant Agarwal, Saurabh Gupta, Digvijay Badghaiya, Nitin Shinde, and Bindu Sai M, for their unwavering friendship and constant encouragement, which have always been a source of strength and joy beyond academics.

I extend my heartfelt love and affection to my cousins, Vishal Bansal, Praveen Bansal, Rahul Bansal, Deepak Bansal, Vivek Bansal, Sashank Bansal, Sunita

Bansal, Sebei Bansal, and Priyal Bansal, whose constant support, care, and encouragement have always been a source of strength and happiness in my life. I respectfully acknowledge the loving memory of my grandfather, Ram Narayan Agarwala, my grandmother, Bhagirathi Devi, and my sister-in-law, Sikha Agarwal. Their values, love, and kindness laid a strong foundation for my life and character.

Last but not least, I convey my deepest gratitude and a profound sense of indebtedness to my mother, **Mrs. Saroj Devi**, and my father, **Mr. Uma Shankar Agarwal**, for their eternal love, invaluable support, and countless sacrifices throughout my life. Whatever I have achieved so far, and whatever I may achieve in the future, will always be because of their blessings, encouragement, and unwavering faith in me. I also extend my heartfelt thanks to my elder sister **Punam Agarwal**, her husband **Harish Agarwal**, and their family; **Sheetal Bahety** and her husband **Vikash Bahety**, along with their family; and **Priyanka Jajodia** and her husband **Mrinal Jajodia**, and their family, whose constant support and affection have strengthened my morale and stood by me in every situation. I remain equally grateful to many others whose names may not be mentioned here, but whose kindness and help I will always cherish. Above all, I thank the Almighty for blessing me with a wonderful family, supportive friends, inspiring teachers, and the strength and health to complete this journey.

"Every step of this journey was guided by a quiet faith that the plan was never prewritten, but shaped slowly through every choice, every effort, and every moment of perseverance. This work stands as a reflection of that belief."

Abhishek Bansal

Abhishek Bansal

Contents

List of Figures

List of Tables

Notations

List of Abbreviations

ACF	Autocorrelation Function
ADT	Average Detection Time
AUC	Area Under the Receiver Operating Characteristic Curve
CCT	Cycle Count Threshold
CSTR	Continuous Stirred Tank Reactor
DFN	Deep Feedforward Network
DNNs	Deep Neural Networks
FFT	Fast Fourier Transform
FFT–ACF	Fast Fourier Transform of the Autocorrelation Function
FN	False Negatives
FNR	False Negative Rate
FP	False Positives
FPR	False Positive Rate
GC	Granger Causality
ISDB	Industrial Stiction Benchmark Dataset
ML	Machine Learning
MV	Manipulated Variable
OP	Controller Output
ORI	Oscillation Ranking Index
PFI	Permutation Feature Importance

PV	Process Variable
RCA	Root Cause Analysis
RMSE	Root Mean Square Error
ROC	Receiver Operating Characteristic
SISO	Single Input Single Output
SP	Setpoint
TN	True Negatives
TP	True Positives

List of Symbols

L	Total number of samples in the signal
f_s	Sampling frequency of the signal
Δt	Average sampling interval between consecutive time points
δt_{shift}	Time-shift interval applied to the signal
α_{shift}	Scaling factor controlling the time-shift spacing
n_s	Total number of shifts considered in each direction
$\mathcal{D}_x$	Augmented dataset formed from time–signal pairs after time shifts
τ–metric	Time-lag-based metric combining correlation strength and lag information
$f_{\max}$	Frequency corresponding to the maximum magnitude in the frequency spectrum
f_i	Frequency component in the magnitude spectrum
T_i	Period of oscillation corresponding to frequency f_i
$S(f)$	Magnitude spectrum obtained from the FFT of the ACF
θ_m	Magnitude threshold applied to $S(f)$ for selecting significant frequency components
n_{cycles}	Number of cycles present in the signal
ξ_i	Oscillation Ranking Index (ORI) corresponding to frequency f_i

$A_{m,j}$	Average oscillation amplitude for the j^{th} segment in a signal
A_f	Overall mean oscillation amplitude across all segments in a signal
$A_{f,i}$	Predicted oscillation amplitude for the i^{th} time-series signal
A_i^{ref}	Reference (true) amplitude for the i^{th} time-series signal
μ_j	Mean value of the j^{th} segment in a signal
$\mathcal{M}_j$	Maximum value within the j^{th} segment in a signal
$\mathcal{N}_j$	Minimum value within the j^{th} segment in a signal
N_{bins}	Number of bins per axis used in the two-dimensional histogram (heatmap)
$H_x(i,j)$	Raw heatmap (2D histogram) representing counts of time–value pairs in each bin
μ_{H_x}	Mean bin count across all bins in the heatmap
$\tilde{H}_x(i,j)$	Normalized heatmap obtained by dividing each bin by the mean count
$\tilde{H}_{\text{PV}}$	Normalized heatmap corresponding to the process variable (PV)
$\tilde{H}_{\text{OP}}$	Normalized heatmap corresponding to the controller output (OP)
$h_x(t)$	Merged heat trace at each time t for signal x
$h_{\text{PV}}(t)$	Merged heat trace for the process variable (PV)
$h_{\text{OP}}(t)$	Merged heat trace for the controller output (OP)
$\hat{h}_{\text{PV}}$	Mode of $h_{\text{PV}}(t)$
$\hat{h}_{\text{OP}}$	Mode of $h_{\text{OP}}(t)$
$\tilde{h}_{\text{PV}}(t)$	Filtered and mode-normalized merged heat trace for process variable (PV)
$\tilde{h}_{\text{OP}}(t)$	Filtered and mode-normalized merged heat trace for controller output (OP)
PV_i^{sum}	Summed PV heatmap values over the i^{th} cycle
OP_i^{sum}	Summed OP heatmap values over the i^{th} cycle
Ratio_i	PV/OP heat ratio for the i^{th} cycle

R	Set of valid ratios
$\|R\|$	Number of valid cycles contributing to the ratio analysis
p_R	Percentage of valid cycles
μ_R	Mean of valid ratios
σ_R	Standard deviation of valid ratios
$\mu_R^{(k)}$	Loop-level mean PV/OP heat ratio for the k^{th} loop
$y^{(k)}$	Binary ground-truth label for the k^{th} loop ($y = 1$: Sticky, $y = 0$: Non-Sticky)
θ	Candidate threshold value for classifying loops based on μ_R
θ^*	Optimal threshold maximizing the F_1 score
F_1 score	Weighted harmonic mean of Precision and Recall
$\mathbf{X}$	Data matrix of sensor measurements
x_i, x_j	Time-series variables (signals) indexed by i and j
$C_{i,j}$	Maximum absolute cross-correlation between x_i and x_j over non-negative lags
$L_{i,j}$	Lag (non-negative) at which the maximum in $C_{i,j}$ is achieved
σ_j^2	Sample variance of the j^{th} variable in the time series dataset
G_c	Controller transfer function matrix
G_p	Process transfer function matrix
G_m	Sensor model transfer function
η	Noise power values for Gaussian white noise

Chapter 1

Introduction and Literature Review

Many industrial processes, such as refineries, power plants, and chemical manufacturing units, operate with a large number of process variables and interconnected control loops. For regulating key process variables such as temperature, pressure, and flow, these processes depend on automatic control loops to ensure safe, reliable, and optimal process operation. As shown in Figure 1.1, each control loop generally follows a feedback structure where the measured process variable (PV) is compared with a desired setpoint (SP). The difference between them, the control error, is input to a controller that outputs a signal which adjusts the final control element. This continuous feedback operation maintains the process variable at the optimal level and automatically accounts for disturbances.

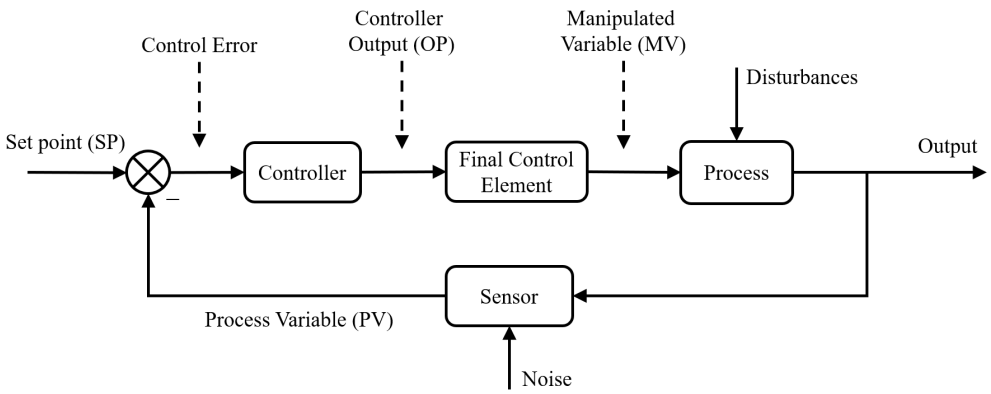


Figure 1.1: Basic structure of a feedback control loop [1].

Though the feedback action helps in maintaining the process stability and performance, the overall control loop performance depends on the proper operation of

each control loop component. Various sources of faults, such as improper controller tuning, control valve stiction, external disturbances, sensor drifts, etc., may result in process disturbance. These disturbances, either oscillatory or non-oscillatory, may lead to higher energy usage and suboptimal plant performance. In industrial systems with a large number of interconnected loops, these faults may propagate from one loop to another and significantly impact the product quality and overall plant efficiency.

The main sources of fault due to which control loop performance may degrade are briefly categorized into the following:

- **Sensor Faults:** Sensors are used for the measurement of process variables, which are then compared with the desired setpoint. The difference is then provided as feedback to the controller. The controller then takes control action to keep the process stable. Faults such as sensor drift, noise, or complete sensor failure may result in inaccurate feedback, causing the controller to take inappropriate actions. Thus, sensor faults may result in suboptimal process conditions, affecting both plant safety and efficiency [4].

- **Actuator Faults:** An improperly functioning control valve or actuator is a major source of fault in industrial processes. Issues such as tight packing, mechanical wear, backlash, hysteresis, etc., can introduce nonlinearities or dead zones in valve operations [5]. These faults result in oscillations in the process loop output, even when the controller tuning is correct. According to Desborough and Miller (2002), 32% of controllers classified as poor or average in an industrial survey by Honeywell showed problems in control valves [6].

- **Process-Related Issues:** With time, the process may undergo changes due to equipment aging, fouling, catalyst degradation, or changes in process operating conditions. External disturbances in feed composition, flow rate, or environmental conditions can also affect process behavior. Hence, the

performance of the process decreases if such disturbances are not taken into account during the process design. Also, a process design should be such that it minimizes interactions between loops, but in practice, control loops often interact with each other in large industrial systems. Hence, faults in one loop can propagate to other loops, causing disturbances across the plant [7].

- **Controller Tuning:** Controller parameters can directly affect the control loop performance and process stability. When a controller is tuned poorly, it may lead to sluggish or oscillatory behavior in process variables. In many plants, controller parameters are tuned only once during commissioning and left unchanged although the process has changed over time. Moreover, the controller, when tuned for worst-case conditions, may result in slow control responses, reducing overall loop performance [1]. Also, oscillations may arise in the controlled variable or the process output if the controller is tuned such that the loop operates close to instability. In addition, integral action can also contribute to oscillatory behavior, particularly in systems with time delays or nonlinear dynamics. Hence, proper controller tuning is vital to eliminate such oscillations and ensure stable and efficient loop operation [7].

- **Human and Operational Factors:** Control loop performance also gets affected when there is limited maintenance, a lack of skilled operators, and time constraints during commissioning. Also, sometimes the underperforming control loops are left unattended or are operated manually, which results in suboptimal process conditions. This causes long-term degradation that goes undetected until product quality or production stability is compromised.

Hence, faults in the control loop not only degrade the overall plant performance but also vary in their nature. Faults in the control loop may be gradual or abrupt, linear or nonlinear, and can develop as drift, bias, oscillations, or intermittent

failures. Such faults often interact and slowly evolve over time, making their detection challenging through routine observation alone. Hence, to maintain the plant performance, it is important to have a systematic monitoring and analytical tool capable of distinguishing normal process variations from performance deterioration. In this regard, Fault Detection and Diagnosis have emerged as a vital component of modern process control loop performance.

1.1 Fault Detection and Diagnosis

Fault Detection and Diagnosis (FDD) in a process industry plays an important role in identifying and addressing disturbances, thereby maintaining process stability. In general, real-time FDD has become increasingly important in the context of Industry 4.0, where large industrial plants require rapid identification and correction of faults for optimal plant performance.

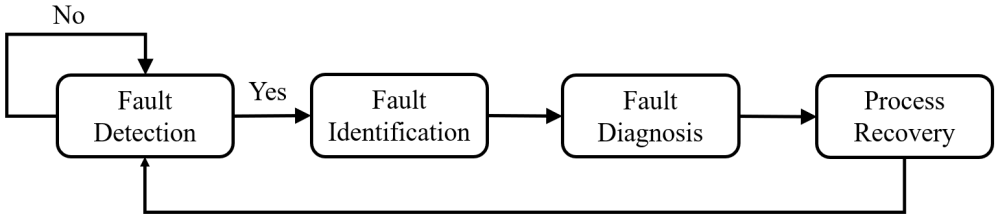


Figure 1.2: Framework for control loop fault management [2].

Fig. 1.2 presents the systematic approach generally used to maintain optimal operation in industrial control systems. Fault detection identifies if the variable is deviated from the desired control loop behavior, whereas fault identification separates the variables which are most relevant for finding the potential source of the disturbance. This allows the operator to look for the source of the fault in an efficient manner. In the next step, fault diagnosis includes finding the nature, location, and cause of the fault, such as a fault originating from sensors, actuators, controllers, or process nonlinearities. Finally, the last stage, i.e., process recovery, involves implementing corrective actions to restore loop performance. Together, these steps constitute a closed-loop framework for continuous monitoring, fault isolation, and recovery of industrial control performance [2, 8].

For the FDD framework to effectively address the faults, several desirable characteristics must be satisfied for efficient operation in industrial processes. Firstly, it should be able to quickly detect the presence of faults without generating excessive false alarms, and then diagnose faults to maintain a balance between responsiveness and robustness. In addition, FDD should be able to distinguish between different types of faults, even under noisy or uncertain conditions. It should also be adaptable to process design inaccuracies, measurement noise, and process change over time. Furthermore, novelty identifiability, the ability to recognize previously unseen faults, is critical for maintaining long-term reliability. Finally, a practical FDD system should provide valuable insights that can efficiently help and direct the operators' understanding of the fault origin and propagation, while maintaining easy and fast computational and modeling requirements [4].

Over the years, several FDD methods have been developed, and can be classified as knowledge-based, model-based, and data-driven approaches [4, 8–10]. Knowledge-based methods utilize qualitative reasoning, heuristic rules, or prior knowledge to detect and diagnose process disturbances. In model-based methods, mathematical representations of the process are used, which generate residuals, indicative of deviations from optimal process conditions. Recently trending, data-driven methods use statistical analysis and machine learning to extract fault characteristics directly from historical or real-time process data. Despite significant progress, practical implementation in complex industrial systems remains challenging due to nonlinearities, unmodeled dynamics, process uncertainties, and measurement noise.

In particular, for dealing with various fault detection problems, many useful methods have been developed and applied [4, 11–14]. While fault detection has been extensively studied and effective techniques exist for identifying abnormal behavior, fault diagnosis which involves determining the exact source and nature of the fault, remains significantly more challenging in practice [15]. A large portion of the existing literature focuses primarily on single-fault detection, with limited

attention given to the diagnosis of the underlying root cause.

However, it is a difficult challenge to address a wide range of process abnormalities with a single FDD method [16]. Hence, many studies are directed towards solving specific issues, such as handling nonlinearity, non-stationarity, autocorrelation, cross-correlation, and non-Gaussian data characteristics [17]. In addition, most process data have multiscale behavior, where important time and frequency-domain features are hidden by measurement noise and unknown disturbances [8, 18]. Therefore, many FDD frameworks are developed to focus on specific classes of faults rather than offering a universal solution.

Among these specific issues, the most common and well-documented problem in the process industry is oscillations in control loops [19]. Oscillatory behavior can lead to poor product quality, higher rejection rates, increased energy usage, and reduced throughput. Generally, the source of such oscillations is improper controller parameters, external disturbances, multivariable interconnections, or stiction in control valves. Therefore, accurately detecting and diagnosing the oscillatory behavior is important, as it reduces plant profitability. Studies have reported that 30–41% of control loops in process industries show oscillatory behavior [20, 21]. Also, modern process plants generally contain between 500 and 5000 control loops [6], and manual inspection of all loops is a challenging task, leading to many undetected oscillations [22, 23].

As reported, valve stiction is one of the major sources of oscillatory behavior in control loops that occurs due to a stick-slip effect within control valves [24]. Due to the presence of stiction, there is a delay in valve stem movement relative to the controller signal, often producing sustained oscillations in the process variable [25]. It is typically caused by tight packing, particulate buildup, lubricant degradation, or seal wear, and is usually quantified as a percentage of the valve's operating range [7]. Stiction is estimated to account for 20–30% of all oscillatory loops [20, 26, 27]. A study of over 26,000 PID loops showed that only 16% performed well, while 38% were rated as fair or poor, with valve stiction identified as a key source

of problem [28]. In general, the valve position is rarely measured or is unavailable, which makes detecting stiction a challenging problem. Moreover, distinguishing between the oscillatory patterns of stiction and the oscillations due to other sources is difficult. This has led to increased research interest in data-driven and machine learning–based stiction detection methods [29, 30].

Industrial control loops are often interconnected, due to which, the abnormal behavior in one loop can propagate and influence others [4, 31]. In addition, while oscillation or stiction detection can identify the nature of suboptimal performance, they do not reveal the source of the problem. This highlights the importance of root cause analysis (RCA), which aims to isolate and trace the underlying source responsible for observed loop disturbances. Identifying the true root cause is essential for implementing effective corrective actions, preventing fault propagation, and sustaining overall plant performance.

In summary, while fault detection has been widely studied, and significant progress has been made, identifying the exact causes of control loop performance problems is still difficult in real applications. Many industrial loops still show oscillations and valve-related faults that affect stable operation. Therefore, more focused research is needed on common faults such as oscillations, valve stiction, and their root causes. The following sections review the existing literature related to these specific problems.

1.2 Literature Review

The rising importance of plant safety, efficiency, and profitability has increased the need for effective methods to monitor and diagnose control loop performance. Critical challenges include the detection of plant-wide oscillations and the diagnosis of stiction in control valves which serve as the primary actuators in process industries such as petroleum refining, chemical manufacturing, and pulp and paper [30]. Stiction in control valves delays the stem movement in the valve relative to the

controller signal. This delay, caused by static friction, introduces nonlinearities and sustained oscillations in the process variable [25]. Therefore, for systems which is suffering from frequent valve actuation, early and accurate detection of stiction is important for maintaining process stability, reducing maintenance costs, extending valve lifespan, and ensuring operational safety [7, 30, 32]. Simultaneously, identifying root causes of performance degradation is essential for effective fault mitigation. In recent years, research studies have focused more on data-driven and machine learning-based methods, reflecting the broader Industry 4.0 transformation. With increasing use of sensor and process data availability [29], there is an increasing demand for robust, automated, and interpretable detection methods capable of operating across diverse process conditions. The reason for the shift is to improve diagnostic reliability while minimizing manual tuning and domain-specific modeling effort, ultimately enabling intelligent and scalable monitoring in complex industrial systems. This review organizes the evolving literature around three core methodological paradigms: 1) Signal Processing and Time-Series Analysis, 2) Model-Based and Statistical Frameworks, and 3) Data-Driven and Machine Learning Approaches. Each paradigm provides distinct tools for addressing the intertwined problems of oscillation detection, stiction diagnosis, and root-cause analysis.

1.2.1 Signal Processing and Time-Series Analysis Methods

This approach extracts diagnostic features directly from process signals such as PV and controller output (OP), using algorithmic transformations, without requiring an explicit process model. Its strengths are interpretability and direct applicability to industrial time-series data.

Over the years, several techniques have been proposed to address oscillation detection. Hägglund [33] used the control error for a time-series signal and analysed the Integrated Absolute Error (IAE) between successive zero crossings. Later, this idea was extended by Thornhill et al. [34] for offline detection of oscillations

in control loops. However, the performance of these approaches decreases in the presence of noise. To account for the noise in signals, Miao and Seborg [35] developed a method for suppressing the noise by utilizing the autocorrelation function (ACF) of the controlled variable or control error to extract a signal that preserves the oscillation frequency. The extracted signal is then used to calculate the decay ratio, which confirms the presence of oscillations if the ratio exceeds a defined threshold. This idea was further refined by Thornhill et al. [36] by analyzing the regularity of zero crossings in the ACF, and confirming the presence of oscillations if the period remains consistent over time.

Among the existing works, wavelet-based analysis tools [37–39] have been applied in industrial plants to detect multiple oscillations in measurement signals and diagnose root causes. The tool examines the input–output characteristics of valve and manipulated variable (MV)–process variable (PV) plots. The signals are converted into a time–frequency representation to suppress the effect of noise and enable pattern matching of time-resolved spectra to identify oscillatory behavior [37–39]. Later on, this idea was further extended by Bounoua et al. [40], in which integrated Detrended Fluctuation Analysis (DFA) with an Improved Empirical Wavelet Transform (IEWT) allowed real-time detection of coherent oscillations in control loops. This method overcomes issues such as mode mixing and over-decomposition in standard EWT, and makes it easier to separate true oscillations from background noise. Similarly, Salsbury and Singhal [41] used AutoRegressive Moving Average (ARMA) models to find oscillatory modes in single-input single-output (SISO) control loops. This method offered a computational advantage by relying only on zero-crossing counts instead of full data storage.

Further, Srinivasan and Rengaswamy et al. [42] developed an oscillation characterization approach based on modified Empirical Mode Decomposition (EMD). This method is capable of detecting multiple oscillation modes and gives estimation of period, amplitude, and strength while reducing false zero-crossings. The spectral envelope method [43] also demonstrated strong noise attenuation and

multi-oscillation detection capabilities but remains computationally demanding. Zakharov et al.[44] developed a correlation-based approach to detect oscillations by assessing the similarity between consecutive periods. Also, Wang and Zhao [45] introduced a multi-lag dynamic Slow Feature Analysis (SFA) method that improves oscillation detection and root-cause tracing in interconnected loops. Sharma et al. [46] proposed a computationally efficient oscillation detection technique based on Linear Predictive Coding (LPC). This method uses the roots of LPC polynomials and cross-correlation spectra for robust detection under noise and slow-varying conditions.

These same principles form the basis for classic stiction detection techniques. Large-scale industries consist of hundreds of control valves, and manual inspection of each loop for detecting stiction is a time-consuming task, particularly for routine monitoring, underscoring the importance of automated detection techniques. Several approaches without the need for a process model have been proposed, such as CURVE [47], AREA [48], CORR [49], HIST [50], and shape-based methods developed by Kano and Yamashita [51–53]. These methods shows good performance when the setpoint is constant under constant, but their accuracy decreases significantly when the reference signal varies. For example, Yamashita's method assumes a consistent relationship between the PV and valve position, an assumption that holds for flow loops but fails in level or integrating processes.

To account for variation in the reference signal, Brásio et al. [54] proposed a finite-difference transformation method, but relies on valve position data, which is rarely available in industrial environments. Dambros et al. [55] proposed the SLOPE and ZONE methods in which the shapes of OP and PV signals are compared with predefined triangular or sinusoidal patterns. These shape-based methods show good performance for clean, high-resolution data, but they are highly sensitive to noise, sampling rate, and signal distortions. Later on, attempts were made to account for these issues through sampling-rate adjustment frameworks [56, 57]. Pozo Garcia et al. [58] found the limitations of individual algorithms and proposed

a hybrid detection strategy combining multiple techniques to improve robustness under varying operating conditions. Furthermore, Thornhill et al. [59] found that stiction introduces odd harmonics in control signals, a property later leveraged by Aftab et al. [60, 61] using Hilbert–Huang Transform (HHT) and multivariate empirical mode decomposition (MEMD) for adaptive detection. These methods enhance reliability under noise and nonlinear disturbances, though at the expense of increased computational complexity and implementation effort.

Similarly, signal processing forms the foundation for many root cause analysis techniques. Techniques such as cross-correlation function (CCF) [62] extract causal relationships directly from process data. The spectral envelope method [43, 63] has also been used for frequency-domain causal analysis and oscillatory fault identification. They are effective in isolating frequency-specific relationships, but its scope is limited to oscillatory disturbances and cannot generalize to non-oscillatory faults.

1.2.2 Model-Based and Statistical Methodologies

This paradigm involves constructing explicit representations, either physical, statistical, or probabilistic, of the process or its disturbances to serve as benchmarks for performance or templates for diagnosis.

For stiction characterization, Hammerstein models describe stiction behavior as a static nonlinear block, followed by a linear dynamic block that captures process dynamics [64]. In addition, versions such as HAMM1–3 [7] have been developed, but their effectiveness was limited by sensitivity to parameter initialization and ambiguity in handling integrating loops [65]. Several other improvements includes robustness metrics [65], hybrid time–frequency analyses [66], and bootstrap-based parameter estimation [67] to improve reliability.

Bacci di Capaci et al. [68, 69] did a comparative study of linear models (AutoRegressive model with eXternal input (ARX), AutoRegressive Moving Average with eXternal input (ARMAX), Extended AutoRegressive model with eXter-

nal input (EARX),Extended AutoRegressive Moving Average with eXternal input (EARMAX)) and nonlinear models (Kano's and He's) for control loop analysis. Their study reports that simpler models do well under clean operating conditions, whereas extended models show greater robustness in nonstationary environments. In addition, Jeremiah et al. [70] presented an integrated detection–compensation scheme using an adaptive neuro-fuzzy inference system (ANFIS) in conjunction with dual-mode model predictive control, while El-Ferik et al. [71] recently developed a robust hybrid framework combining signal processing and machine learning for real-world valve systems. With many advancements, model-based approaches still face challenges related to loop-specific parameter tuning, excitation dependency, and limited scalability in large, complex industrial settings.

Classical statistical and regression-based techniques are widely used because of their simplicity, interpretability, and low computational cost. Zheng et al. [72] developed a K-means clustering-based moving window approach for detecting and quantifying valve stiction. Recent studies, such as Damarla et al. [73], applied a linear regression method to extract OP—PV dynamics within sliding windows. In addition, Liu and Wang [74] used this concept and adaptively adjusted the window size according to curve shape indices and signal characteristics. However, these methods rely on manually selected thresholds and show limited robustness across different industrial datasets.

For root cause analysis, several techniques have been reported in the literature. Several methods that come under knowledge-based methods, such as signed directed graphs (SDG) [75], adjacency matrices (AM) [76], and fault trees (FT) [77], show good performance in structured systems with well-defined causal relationships. However, they depend strongly on prior process knowledge and causal models, which limit their applications in complex and evolving industrial environments.

Statistical methods for causality have gained more attention recently. Many automated root cause detection techniques have been proposed in the last two

decades [78–81]. Granger causality (GC) [82–84] methods are simple and depend on multivariate linear regression. It is used for linear dependencies, which limit its application in nonlinear and time-varying industrial processes. Transfer entropy (TE) [85, 86] is used for causality analysis in nonlinear systems by quantifying directional information flow between variables. TE methods need large datasets, and the computational cost of estimating joint probability density functions (PDFs) increases with data dimensionality. This makes real-time causal modeling infeasible; furthermore, during process upsets, the true causal structure may become distorted or obscured, reducing the reliability of TE-based diagnostics.

Bayesian network–based methods have also been used to extract probabilistic dependencies among process variables [87–90]. These methods use conditional probabilities to describe the fault propagation path. But these methods are computationally extensive as they depend on accurate estimation of PDFs, making them sensitive to data quality and availability. To address these limitations, Kumari et al. [91] developed a Hierarchical Bayesian Model (HBM) that incorporates informative priors and key process variables. Though this improves diagnostic accuracy and computational efficiency, the construction of reliable priors remains a significant challenge, especially for complex industrial systems. Later on, Kumari et al. [92] proposed a modified Bayesian Network (mBN) method to handle cyclic causal structures by converting weak causal links into temporal ones. Since this method relies on transfer entropy, its computational cost is high. To enhance scalability, a Direct Transfer Entropy (DTE)–based multiblock Bayesian Network was later proposed [93], improving accuracy and efficiency in cyclic loop discovery. However, dividing the process into blocks introduces additional complexity and requires careful segmentation strategies.

1.2.3 Data-Driven and Machine Learning Frameworks

Driven by increased data availability and computing power, this methodology uses algorithms to learn diagnostic patterns directly from data, minimizing the need

for first-principles modeling or manual feature engineering.

In the last two decades, there has been significant growth in machine learning (ML) based approaches for industrial fault detection. Unlike traditional rule-based methods, which require modelling of systems, ML-based methods can automatically learn patterns from data. This enables robust and adaptive fault diagnosis and makes ML particularly well-suited for oscillation detection in control loops, where complex and nonlinear interactions often challenge conventional analytical approaches.

Early ML-based studies focused on quantifying stiction-induced oscillations in control valves [94–96]. Later, Dambros and Trierweiler [97] and Dambros and Farenzena [98] achieved accurate oscillation detection using features extracted from the Fast Fourier Transform (FFT) of time-domain signals. However, these models relied heavily on handcrafted frequency-domain features, making them sensitive to noise and less reliable under varying industrial conditions. Moreover, accurate estimation of oscillation parameters such as period and amplitude remains essential, as not all oscillatory loops in large-scale plants require immediate intervention. Quantifying these parameters helps prioritize maintenance actions based on oscillation severity, reducing unnecessary shutdowns and optimizing plant performance [97].

A deep learning framework that combines Prony-based Infinite Impulse Response (IIR) modeling and Deep Neural Networks (DNNs) is developed by Sharma et al. [99]. This method shows better frequency and amplitude estimation when compared to Support Vector Machine (SVM) and EMD-based methods. Arbabi Yazdi et al. [100] further extended ML-based oscillation diagnosis using classifiers such as SVM, k-Nearest Neighbors (KNN), and Naïve Bayes to identify root causes, including valve stiction, poor tuning, and external disturbances. More recently, Wang et al. [101] developed a Convolutional Neural Network (CNN)-based visual framework utilizing models like MobileNet-V1 and EfficientNet-B0. Their approach gives strong robustness to noise, nonstationarity, and overlapping

oscillations, but shows fluctuations due to sparse intra-class feature distributions. In recent years, stiction detection research has increasingly focused on data-driven and machine-learning methods, reflecting the broader Industry 4.0 transformation. Both unsupervised and supervised learning techniques have recently gained more attention. Daneshwar and Mohd Noh [102] applied fuzzy c-means clustering to show the difference between stiction-induced oscillations and external disturbances. Miskin et al. [103] developed a Principal Component Analysis (PCA) method, and Teh et al. [104] then combined nonlinear PCA (NLPCA) with autocovariance analysis (AC), achieving better accuracy. More recently, Shang et al. [105] introduced a dynamic stiction index based on slow feature analysis and the Hurst exponent, providing improved sensitivity to nonlinear trends.

Artificial intelligence (AI) and deep learning methods have shown great advancement in recent years for control valve stiction detection. Amiruddin et al. [106] used a multi-layer neural network trained on transformed OP—PV data to classify stiction conditions. In another study, Venceslau et al. [96] estimated Choudhury's stiction parameters using a multilayer perceptron (MLP). Damarla and Huang [107] developed a Learning Vector Quantization Neural Network (LVQNN) trained on both simulated and industrial datasets, achieving improved generalization performance.

Several studies have used image-based signal transformations for CNNs. Henry et al. [108, 109] converted OP–PV signals into unthresholded recurrence plots (URPs) and used AlexNet for the classification task. This method shows enhanced performance through transfer learning. Memarian et al. [110] combined Markov Transition Fields (MTF) with CNN architectures for robust detection under noisy conditions. In addition, Dambros et al. [97] and Kamaruddin et al. [111] utilized pixel-based encodings and butterfly-shaped CNN structures to detect both oscillation and stiction phenomena.

SVMs have also been successfully applied, which includes methods such as training an SVM classifier on OP and PV signals to distinguish stiction from other

oscillatory behaviors by Yazdi et al. [112]. Guan et al. [113] used phase-space reconstruction and a recurrence-based feature analysis framework to show strong generalization to industrial datasets. Vazquez et al. [114] used valve position and controller output data to train a CNN. More recently, with the rise of multimodal artificial intelligence, vision–language models (VLMs) have been explored for interpretable stiction analysis. StictionGPT [115] integrates a large vision–language model with few-shot adaptation, utilizing shape descriptors and textual prompts to achieve higher generalizability and transparency in decision-making.

Deep learning methods have also been used for root cause analysis as they can capture nonlinear and temporal dependencies in high-dimensional data [116–118]. Method like Recurrent neural networks (RNNs) works well for time-series modeling, as they can capture sequential correlations between variables. Nevertheless, traditional RNNs often suffer from vanishing or exploding gradients when handling long time sequences, limiting their capacity to retain long-term dependencies. While recent advancements such as LSTM and GRU architectures account for some of these issues, most existing deep learning models focus primarily on fault prediction or time-series forecasting rather than explicit root cause identification or fault propagation analysis. As a result, interpretability and causal understanding remain key challenges for deploying such models in real-world industrial environments.

Even though both traditional and AI-based methods have improved a lot, existing methods continue to face challenges. Issues such as sensitivity to parameter tuning, reliance on handcrafted or high-dimensional features, substantial data requirements for training, and limited robustness across varying process conditions limit the application of most of the studies.

1.3 Research Gaps

Based on the comprehensive literature review, several research gaps have been identified across the areas of oscillation detection, valve stiction diagnosis, and root cause analysis in process control systems.

1. When applied to real industrial data, the presence of noise in process data and nonstationary operating conditions often reduces the reliability of most existing oscillation detection techniques. Also, traditional rule-based methods depend on manual thresholding and heuristic parameters, which limit their adaptability across different loops and process conditions.

2. Intermittent, multi-frequency, or overlapping oscillations, which are often present in industrial processes, limit the applicability of many existing methods. Although signal decomposition and wavelet-based techniques can address some of these challenges, their high computational cost makes them unsuitable for real-time or plant-wide monitoring.

3. Machine learning and deep learning methods have shown high accuracy but depend on a large set of frequency-domain or wavelet-based features. This increases the computational cost as well as limits their practicality for online applications.

4. For valve stiction diagnosis, both model-free and model-based techniques have been developed, but most of them work only under constant set-point conditions. As a result, their performance decreases when reference signals vary or when dynamic disturbances occur in the process. In addition, model-based methods, such as Hammerstein or Preisach models, need extensive parameter tuning and are sensitive to specific process dynamics. Similarly, model-free and shape-based methods are affected by noise and sampling variations. Furthermore, many data-driven algorithms depend on valve position measurements, which are rarely measured in practice, limiting their use in

large-scale plants.

5. While deep learning and image-based frameworks have been recently developed, they involve complex signal transformations, such as recurrence plots or Markov transition fields, and require large amounts of labeled data for training. These approaches can be difficult to interpret and have yet to demonstrate consistent scalability across different plant conditions.

6. Many existing RCA methods also face limitations as they depend on process causal maps, process topology, or expert knowledge, which are not always available in industrial processes. Knowledge-based techniques such as signed directed graphs and fault trees are difficult to update as process conditions evolve. Data-driven methods, including Granger causality and transfer entropy, are computationally demanding and struggle with nonlinear, high-dimensional data. Although spectral envelope methods work well for oscillatory faults, they do not extend to non-oscillatory scenarios.

7. Bayesian and deep learning–based RCA methods require either detailed prior information or very large datasets, and their ability to find true causal dependencies remains limited. Overall, there is a need for simple, robust, and computationally efficient data-driven methods that are capable of performing oscillation detection, stiction diagnosis, and root cause identification without explicit process models or topology information.

1.4 Research Objectives

The objective of this thesis is to develop simple, computationally fast, and data-driven methods for effective detection, characterization, and diagnosis of control loop faults in process industries. The study focuses on three major aspects of loop performance: oscillation detection and characterization, control valve stiction detection, and root cause analysis.

1. The first objective is to develop a method for detecting and quantifying oscillations in control loops. The method should be computationally fast and robust under noisy or nonstationary conditions. The method should be able to classify oscillations as regular and irregular oscillations and accurately determine their period and amplitude. This will enable automated, intelligent, and help operators to take quick maintenance decisions to keep processes stable and efficient.

2. The second objective is to develop a simple, model-free, and interpretable data-driven framework for detecting control valve stiction using routine process data. The will ensure reliable detection across a wide range of industrial loops without requiring valve position measurements or model-based assumptions. This objective aims to make the method adaptable, scalable, and suitable for real-time process monitoring.

3. The third objective is to develop a scalable and computationally efficient method for RCA in multivariate industrial processes. The primary aim is to utilize only the process data to directly pinpoint the possible sources of process disturbances without requiring prior knowledge or causal maps. This will thereby enable efficient and automated diagnosis of both oscillatory and non-oscillatory faults in large-scale industrial environments.

The three research objectives are designed with the idea of providing a systematic and data-driven approach for diagnosing control loop faults in process industries. When oscillations occur in one or more control loops and propagate through the plant, the first objective classify which variables are oscillatory or non-oscillatory. The second objective is then used to determine if the oscillations are due to stiction and classify control loops as sticky or non-sticky. Finally, the third objective identifies the source control loop responsible for the disturbance. These steps form an integrated framework that will support operators in timely fault diagnosis and taking corrective action, thereby enhancing process reliability, efficiency, and

overall plant performance.

1.5 Outline of the Thesis

The focus and contributions of the thesis are outlined below for each chapter. The chapters are organized sequentially to cover oscillation detection and characterization, valve stiction detection, and root cause analysis, followed by the concluding remarks and future scope.

A neural network–based method for oscillation detection in process control loops is presented in Chapter 2. In this work, the time series is pre-processed, and domain-informed feature engineering is used to improve detection accuracy while keeping computational demands low. Dominant features obtained from the FFT and the FFT of the ACF of time-series process data are used as inputs to the neural network. A sensitivity analysis is performed by varying the number of input features to the neural network to assess the impact of feature selection on model performance. The proposed method shows high accuracy and recall at a lower number of input features, making it suitable for online implementation. The method also includes algorithms to estimate the oscillation period and amplitude, and its effectiveness is demonstrated using industrial data.

Chapter 3 extends the work done in Chapter 2, identifying the underlying cause of detected oscillations. This work focuses on the detection of control valve stiction that leads to sustained oscillations and reduced process efficiency. A simple, model-free, and data-driven approach analyzes the dynamic relationship between the OP and PV. The time-series data is first normalized, and then time-shifted signal heatmaps are generated to identify stick–slip behavior and introduce a PV/OP heat ratio as a reliable indicator of stiction. The heat ratio is then used as a feature for a threshold-based classifier and shows strong generalization when evaluated on an industrial test dataset, providing an accurate and computationally efficient solution for stiction detection in control loops.

Chapter 4 introduces a RCA framework designed to identify the sources of process faults in multivariate systems with interconnected control loops. Oscillatory or non-oscillatory faults occurring in one loop can propagate to others depending on the interconnection. In this work, a weighted cross-correlation-based method is used to determine the source variable responsible for system performance degradation without relying on plant topology or causal maps. The algorithm computes cross-correlations across all pairwise combinations of process variables and the lag at which each pair reaches maximum correlation. The combination of both correlation strength and lag information into a weighted average (τ-metric) pinpoints the variable that consistently leads others, potentially indicating the root cause of the fault. The method is validated using the synthetic data generated in MATLAB Simulink and simulated industrial case studies, demonstrating high accuracy in detecting the root causes of oscillatory and non-oscillatory faults.

Finally, Chapter 5 provides the concluding remarks drawn from the studies presented in Chapters 2 through 4. Based on the experience and insights gained during the course of this research, several recommendations for future work are also presented to guide further advancements in oscillation detection, stiction diagnosis, and root cause analysis in industrial control systems.

Chapter 2

Oscillation Analysis in Process Control Loops

2.1 Introduction

Oscillations in process control loops are a widespread problem and often indicate a more severe problem than irregular variability [19]. Process control loops that experience oscillations can result in many problematic outcomes, including subpar product quality, elevated rejection rates, heightened energy usage, and decreased throughput. These issues can arise from excessive controller gains, external oscillation disturbances, interactions in multivariate systems, and friction within control valves. The first step in resolving oscillation issues is efficient detection and diagnosis of oscillatory behavior, which directly affects the profitability of the process plant. Oscillation in control loops of process industries affects about 30 to 41% of all loops [20, 21]. Typically, process industries have between 500 and 5000 control loops [6], which makes it impossible to visually inspect and diagnose the whole plant and as a result, only a part of these loops can be investigated [22], leading to undetected oscillations [23]. The increasing focus on plant safety and profitability has accelerated the demand for methods to identify and diagnose plant-wide oscillations. As a result, the implementation of automated oscillation

detection techniques has become crucial to overcome this challenge.

Various methods have been introduced over the years to detect oscillations in control systems. Hägglund [33] developed an oscillation detection technique based on the magnitude of the IAE between successive zero crossings of the control error for a single time series. Thornhill et al. [34] extended this approach for off-line oscillation detection in control loops. However, the performance of this method deteriorates when the error signals are noisy. To overcome this limitation, Miao and Seborg [35] proposed an alternative method that examines the ACF of the controlled variable or control error. This approach extracts a signal with the same oscillation frequency, but with reduced noise. The technique further analyzes the decay ratio of this signal, triggering an oscillation detection if the ratio exceeds a certain threshold. In another approach, Thornhill et al. [36] utilized the regularity of zero crossings in the ACF to detect oscillations across the entire plant. Oscillation detection is confirmed if the period remains consistent over time.

Among the existing works, a wavelet analysis tool, used in industrial chemical plants, detects multiple oscillations in measurement signals and diagnoses root causes by analyzing valve input-output characteristics and MV – PV plots. This method, converting signals into a time-frequency view, overcomes noise issues and identifies key characteristics using pattern matching of time-resolved frequency spectra [37–39]. Time2Wave software provides visual inspection and quantitative similarity measures between wavelet transforms, offering deep insights into process behavior and oscillation detection. Bounoua et al. [40] further advanced this direction by combining DFA with an Improved Empirical Wavelet Transform for real-time detection of coherent oscillations in industrial control loops. Their method addresses critical limitations of the standard Empirical Wavelet Transform (EWT), notably mode mixing and over-decomposition, by more effectively separating genuine oscillatory components from background noise, thereby improving the clarity and reliability of oscillation diagnosis. Salsbury and Singhal [41] discuss

ARMA models, focusing on detecting oscillatory modes in a SISO feedback control loop. They present a new method for estimating ARMA poles using higher-order crossings, where autocorrelation lags from the counts of crossing events are used to obtain autoregressive parameters. Poles are then determined using a root finding algorithm, providing insights into control performance. This approach offers an advantage by not requiring storage of large data batches, as only crossing counts need to be retained. Srinivasan, Rengaswamy et al. [42] proposed an advanced oscillation characterization scheme based on modified EMD. Zero-crossing information in each oscillating loop is evaluated, providing detailed information about the time period, detection of multiple oscillation modes, amplitude, strength, and time instances of oscillations. This approach keeps the shape of the signal by removing only the non-constant mean: Step 1 involves removing the low-oscillation mode applying a modified EMD procedure, while Step 2 estimates the cumulative area of the separated signal to reduce false zero-crossings. The spectral envelope method [43], is frequency-sensitive, detecting multiple oscillations while also attenuating noise components, making it particularly effective in analyzing noisy or corrupted signals. Using an algorithm-based approach, Zakharov and Jämsä-Jounela [44] designed and tested an algorithm to evaluate the correlation between two periods of oscillation, detecting oscillation when the correlation is high. Wang and Zhao [45] introduced a framework based on multi-lag dynamic SFA to extract, detect, and isolate oscillatory behavior in coupled control loops. This technique not only improves oscillation detection accuracy but also facilitates root-cause tracing with minimal human supervision. Furthermore, Sharma et al. [46] proposed a novel oscillation detection technique using LPC. In this method, the roots of the LPC polynomial are analyzed for oscillation detection, and cross-correlation spectra are used for quantification. The approach demonstrates strong robustness to both noise and slow-varying trends and serves as a computationally efficient alternative to EMD-based techniques.

As seen, most of the techniques proposed in the literature for oscillation detection

are usually data-driven and rely on rules based on criteria similar to those used in visual inspection and mathematical concepts. These rules generally compute a parameter (such as IAE, decay ratio, or regularity), which is then evaluated by an if/else statement. When applied to actual industrial data, these methods reveal low efficiency [119] as industrial data is often noisy and subject to disturbance, leading to irregular frequency and amplitude, intermittent oscillations, or multiple oscillations in the same time series. To be effective, rule-based methods must incorporate all these influences into the algorithm, making it complex, extensive, and challenging to implement. ML has emerged as a powerful tool for detecting faults and has been commonly used in the last two decades. Unlike traditional methods, machine learning models can adapt themselves according to given examples, making them highly efficient for specific tasks without the need for specific rules. This makes it easier to apply machine learning to oscillation detection without requiring rules for each influence, leading to a more robust approach. Several machine learning techniques have been proposed for quantifying stiction in control valves which causes oscillation [94–96]. The works proposed by Dambros and Trierweiler [97], Dambros and Farenzena [98] can accurately detect oscillations. Still, they heavily rely on multiple input features extracted from the FFT of time-domain signals. This can result in flawed detection, especially in the presence of noise, limiting its industrial applications. Moreover, improved identification of the period and amplitude is also necessary. This is particularly important because in large-scale industrial plants, numerous control loops may exhibit oscillatory behavior, but not all require immediate intervention. Accurate quantification of oscillation parameters such as period and amplitude helps prioritize loops based on oscillation severity, enabling targeted maintenance and avoiding unnecessary shutdowns [97]. A deep learning-based approach integrating Prony-based IIR filter modeling with DNNs was later introduced by Sharma et al. [99]. This method enabled accurate detection and quantification of both single and multiple oscillations, estimating frequency and amplitude more effectively than conventional SVM and EMD-based

techniques. Arbabi Yazdi et al. [100] further extended ML-based detection by employing classifiers such as SVM, KNN, and Naïve Bayes to classify the root cause of oscillations—including valve stiction, poor tuning, and external disturbances. Most recently, Wang et al. [101] proposed a CNN-based visual framework using lightweight architectures like MobileNet-V1 and EfficientNet-B0. Their model demonstrated strong robustness against noise, nonstationarity, and overlapping oscillatory patterns, though performance fluctuations were observed due to sparse intra-class feature distributions.

This work presents a domain-specific pipeline for oscillation detection and characterization in industrial signals by enhancing a deep feedforward network (DFN) with frequency-domain feature engineering and selection strategies grounded in process knowledge. While FFT, ACF, and neural networks are established individually, our approach integrates them through several methodological innovations. Features are independently derived from both the FFT and the FFT of the Autocorrelation Function (FFT-ACF), enabling the model to capture both spectral strength and temporal coherence—crucial for handling noisy or irregular signals. To ensure feature relevance and reduce dimensionality, we introduce a cycle count-based thresholding method that filters out spurious peaks based on signal duration. Around each dominant peak, a localized window is applied to construct a compact and interpretable feature set. For oscillation period estimation, we propose the Oscillation Ranking Index (ORI), which prioritizes significant components by considering both frequency and spectral magnitude. A frequency separation heuristic is further used to eliminate closely spaced components, preserving distinct oscillation modes. For amplitude estimation, a time-domain segmentation approach aligned with the estimated period is employed, from which statistical features such as mean, maximum, minimum, and absolute deviation are extracted. Overall, the proposed methodology enables robust detection, classification, and ranking of oscillatory behavior in process control loops—supporting timely fault diagnosis and optimal plant operation.

2.2 Synthetic Data Availability

For training and validating the proposed method, a synthetic dataset provided by Dambros et al. [97] is used. Although the ultimate goal is to apply the proposed ML-based technique to real industrial data, training a data-driven algorithm using industrial data is challenging. Industrial data is often confidential and scarce, making it difficult to gather sufficient data for effective training. Additionally, labeling such data is both labor-intensive and subjective. Even with accurate labels, the data might not cover the full range of parameters and operation of processes.

To overcome these challenges, synthetic data provided by Dambros et al. [97] was utilized for both training and validation of the model. The model's performance was subsequently evaluated using both synthetic and real industrial datasets [120]. The provided synthetic dataset closely mimic industrial data and captures a wide range of process dynamics, configurations, and characteristics. The dataset contains examples of oscillatory and non-oscillatory behaviors with varying lengths, noise levels, and disturbances of different amplitudes. It also includes oscillatory time series with sinusoidal, triangular, or square waveforms (each with varying periods), waveforms smoothed to simulate process filtering, and time series with oscillations of varying frequencies and intensities.

The synthetic dataset consists of dynamic process data for 110,000 variables, where each variable represents a time-series signal. The signals are divided into three categories: non-oscillatory (43,705 variables), oscillatory (33,174 variables), and irregularly oscillatory (33,121 variables). Class 0 represents non-oscillatory variables, Class 1 corresponds to oscillatory variables, and Class 2 to irregularly oscillatory variables. Irregularly oscillatory variables exhibit oscillations with inconsistent frequencies and/or fluctuating amplitudes, often changing in a non-periodic manner over time. Since the dataset is synthetic, the classification of each variable is known with certainty, making it a multi-class classification task. The signal lengths in the dataset vary from 200 to 24,064 time points.

2.3 Methodology for Oscillation Detection and Parameter Estimation

The following presents a methodology for oscillation detection that utilizes frequency-domain features, and an approach for predicting period and amplitude of oscillation applicable to both regular and irregular oscillatory behaviors. Figure 2.1 illustrates the proposed framework for oscillation detection and parameter estimation.

2.3.1 Proposed Method: Neural Network Architecture and Frequency Domain Feature Integration for Oscillation Detection

A three-layer feedforward neural network was employed for oscillation detection, with 400, 100, and 20 neurons in each layer. This architecture, along with its associated hyperparameters (batch size of 10,000, Hard Sigmoid activation function, and Adam optimizer), was directly adopted from the work of Dambros et al. [97], where a random search strategy was used to select the best configuration from 100 sampled hyperparameter sets, drawn from a total search space of 11,907 possible combinations. Since this study utilized the same dataset as Dambros et al. [97],adopting their architecture ensured consistency and enabled a fair and direct comparison of detection performance. The network was implemented in Python and trained on a dataset of 100,000 variables. The test dataset includes 10,000 variables, consisting of 3,985 non-oscillatory, 3,027 oscillatory, and 2,988 irregularly oscillatory variables. The model outputs a 3-dimensional continuous vector. For classification purposes, we convert these to discrete class predictions (0, 1, or 2) by selecting the dimension (i.e., output node) with the highest value. The primary objective of the neural network is to classify input signals into three categories: non-oscillatory, oscillatory, and irregularly oscillatory. The proposed

methodology integrates frequency domain features extracted from both the FFT of the time-series signal and the FFT of its ACF. Peaks in the FFT spectrum indicate potential oscillations, making it an effective tool for identifying strong and regular periodic behavior. The ACF, on the other hand, quantifies the similarity of a signal with time-lagged versions of itself, enabling the detection of repeating patterns even in noisy or weakly periodic data. A key property of the ACF is that it exhibits oscillatory behavior when the original signal is periodic, with its period preserved in the autocorrelation structure. Applying FFT to the ACF (i.e., FFT-ACF) enhances the visibility of this periodicity in the frequency domain. Theoretically, this fusion of FFT and FFT-ACF captures both frequency-domain energy and time-domain regularity, enriching the feature space with complementary information. This dual representation improves the model's ability to discriminate between different oscillation types while maintaining robustness to noise. Dominant frequency-domain features are carefully selected from both FFT and FFT-ACF outputs and fed to the neural network for training. This not only improves oscillation detection accuracy but also reduces computational cost by focusing on the most informative spectral components. The following steps outline the framework and key components of this integrated approach:

Step 1: Preprocessing and Frequency Domain Transformation: The input time-series data is normalized to ensure that each time series has a zero mean and uniform scaling. Specifically, each signal is adjusted by subtracting its mean and scaling by the inverse of its range (max–min). This reduces bias and addresses scale discrepancies across the dataset, enabling effective model training. Normalization ensures that no feature dominates the others, leading to better accuracy of oscillation detection. Following this preprocessing stage, two distinct frequency-domain representations are generated for comprehensive analysis:

- **FFT of the normalized data:** FFT is applied to the normalized time-series data with a fixed length of 2^{13} (8192 points). This provides a suitable balance between frequency resolution and computational efficiency. For each

FFT computation, only the first half of the resulting magnitude spectrum is utilized, as the second half is a mirrored representation of the first. This spectra is referred to as FFT spectra from here onwards.

- **FFT-ACF of data:** The ACF is first computed from the original time-series data without normalization. The resulting ACF is then normalized, similar to the preprocessing step for the raw data. FFT is subsequently applied to the normalized ACF, transforming it into the frequency domain. This process enhances the visibility of periodic patterns that may be obscured in the raw data due to noise, thereby providing a clearer representation of oscillatory behaviors. Only the first half of the resulting magnitude spectrum is retained for further analysis. This spectra is referred to as the FFT-ACF spectra from here onwards.

Step 2: Peak Detection and Feature Selection: Instead of using all the frequency domain features, we perform feature selection to prioritize dominant peaks in the frequency spectra. A dominant peak is defined as the frequency component in the magnitude spectrum (from either FFT or FFT-ACF) with the highest amplitude. This peak typically reflects the most prominent periodic behavior in the signal. However, to ensure that the identified frequency corresponds to a genuine oscillation and not a low-frequency trend, we employ a cycle count threshold (θ_c), which sets a minimum requirement on the number of oscillation cycles present in the signal duration. The procedure adopted is as follows:

- For each frequency spectra, the frequency ($f_{\max}$) corresponding to the maximum magnitude is identified, and its associated time period ($T = \frac{1}{f_{\max}}$) is calculated. The number of cycles, n_{cycles}, is defined as the ratio of the total duration of the signal to the time period (T), given by $n_{\text{cycles}} = \frac{L}{f_s T}$, where L is the number of samples and f_s is the sampling frequency.

- If the calculated number of cycles falls below a specified cycle count threshold (θ_c, CCT) (e.g., 2 cycles), the next significant peak in the frequency spectrum

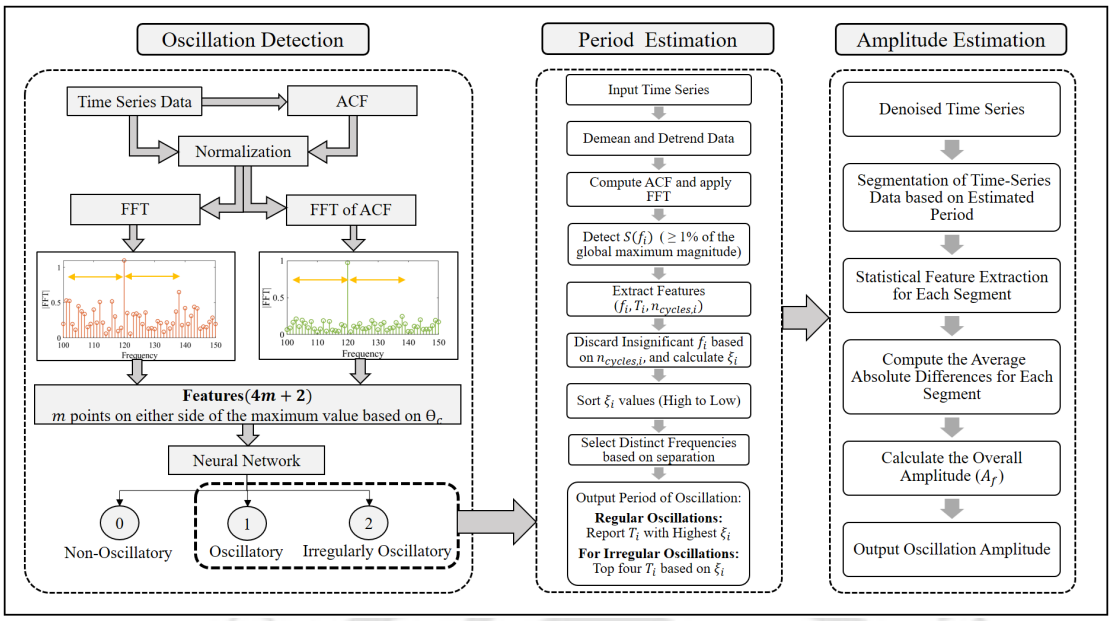


Figure 2.1: Proposed Framework for Oscillation Detection and Estimation.

is considered.

- This iterative process continues until a peak that satisfies the threshold criterion is identified, ensuring that only significant oscillation is selected.
- Once a valid peak is selected, m points on either side of the peak are included, resulting in $2m + 1$ features being extracted from each frequency spectra.

The rationale behind this thresholding strategy is further discussed in Section 2.3.4.

Step 3: Feature Integration and Neural Network Classification: The $2m + 1$ significant features identified in the previous step from both FFT spectra and FFT-ACF spectra are integrated to create a comprehensive feature set, giving a total of $4m + 2$ features. This combined feature set is then used as input for the feedforward neural network model and trained to classify the signals into three categories: non-oscillatory, oscillatory, or irregular oscillatory.

The proposed method is evaluated using standard metrics, including accuracy, confusion matrix, and classification reports, applied to the test dataset. Performance comparisons are made across various combinations of m values (e.g., 5, 10, 15, 50, 100, 200) and θ_c values. This dual evaluation assesses the impact of both feature set sizes and θ_c on oscillation detection performance.

In addition, preliminary results for various values of m were obtained using only the FFT spectra, yielding $2m+1$ features, and similarly for the FFT-ACF spectra. While using the feature from the FFT spectra alone showed good accuracy, the features from the FFT-ACF spectra successfully identified oscillations in some cases where the former failed. However, the combined use of both feature sets significantly improved detection accuracy, demonstrating the complementary strengths of these methods. These results that demonstrate the performance of the algorithm using only FFT features and only FFT-ACF features are provided in the Appendix B.

2.3.2 Approach for Estimating Period of Oscillation

This method outlines the steps for predicting period of oscillation in process data, addressing both regular and irregular oscillations through frequency-domain analysis. Before any analysis, each time-series dataset undergoes preprocessing to eliminate trends and bias. The data is demeaned by subtracting the mean to center it around zero and then detrending is performed by removing a linear trend from each time series. Specifically, a first-degree polynomial (straight line) is fitted to the data using least squares, and then subtracted. This operation eliminates slowly varying linear components that can bias frequency-domain analysis. This ensures that non-stationary components, such as trends, do not interfere with prediction of the period of oscillation.

Step 1: ACF and Fourier Transform: After preprocessing, for each time-series dataset, the ACF is computed to capture periodic patterns. Only the first half of the ACF is used for subsequent analysis. FFT of fixed length 2^{13} is applied to the ACF, transforming it into the frequency domain to highlight the frequency components of oscillations. The resulting magnitude spectrum, denoted as $S(f)$, represents the magnitudes of the frequency components, where each magnitude $S(f_i)$ corresponds to its frequency component f_i. This spectrum is analyzed to identify significant oscillatory frequencies.

Step 2: Frequency Selection and Feature Extraction: Significant frequency components in $S(f)$ represent potential oscillations. A magnitude threshold (θ_m) is applied to $S(f)$ to select the f_i whose $S(f_i)$ exceeds this threshold. The threshold is defined as a fraction (1%) of the global maximum magnitude in $S(f)$. Only those f_i with $S(f_i)$ greater than or equal to θ_m are considered for further analysis. The following features are extracted for each selected f_i:

- **Period of Oscillation (T_i)**: The period of oscillation, calculated as the inverse of the frequency component, $T_i = \frac{1}{f_i}$.
- **Number of Oscillation Cycles ($n_{\mathbf{cycles},i}$)**: The total number of cycles in the time series corresponding to frequency f_i is computed as $n_{\text{cycles},i} = \frac{L}{f_s T_i}$, where L is the number of samples in the time series, f_s is the sampling frequency (in Hz), and T_i is the period of oscillation corresponding to the frequency component f_i.

Step 3: Filtering and Period of oscillation: Frequency component f_i with low $n_{\text{cycles},i}$ values are discarded, as they represent insufficient oscillatory behavior. The minimum $n_{\text{cycles},i}$ threshold is set to 1.5, and the maximum threshold is determined by the length of the time series, specifically $\frac{L}{6}$. The minimum threshold of 1.5 cycles was chosen to ensure that a frequency component completes at least one full oscillation and has sufficient repetition to be considered reliable, avoiding false positives from transient or incomplete cycles. The upper threshold of $\frac{L}{6}$, where L is the length of the time series, was selected to exclude extremely high-frequency components that may result from noise. The following steps are applied for estimation of period of oscillation:

- **Oscillation Ranking Index (ORI, ξ_i)**: A ranking metric is used to assess the significance of the filtered frequency components and is defined as:

$$\xi_i = f_i^2 \cdot S(f_i)^2 \tag{2.1}$$

where $S(f_i)$ represents the magnitude of the spectrum at the frequency component f_i. This metric quantifies the significance of the selected frequency components by considering both the frequency value and its corresponding magnitude. By combining f_i^2 and $S(f_i)^2$, the ORI balances the influence of both the frequency and its spectral power. This makes it particularly effective for identifying dominant frequencies that are both high in value and strong in magnitude. The squared frequency term f_i^2 suppresses low-frequency components that may exhibit large magnitudes due to drifts or long-term fluctuations, which are not indicative of true oscillatory behavior. Simultaneously, squaring the spectral magnitude $S(f_i)^2$ ensures that components with strong energy are emphasized. This formulation effectively penalizes spurious low-frequency peaks while promoting high-frequency, repetitive oscillations that are more representative of dominant dynamics in industrial systems. In irregular or weakly oscillating signals, ORI provides better discrimination by balancing temporal density and spectral strength, allowing for more robust identification of relevant oscillatory patterns compared to simpler metrics based solely on magnitude or linear frequency terms.

- **Frequency Separation:** The ξ_i values are sorted in decreasing order, and the corresponding f_i values are compared. Only the frequency with the higher ξ_i is retained if the difference between the frequencies is less than 5%. This ensures that the selected frequencies are sufficiently distinct.

- **Selection of Dominant Period of Oscillation:** After applying the frequency separation, the remaining valid frequencies are prioritized based on the value of ξ_i to find the most significant period of oscillation.

  - For regular oscillations, the period of oscillation with the highest ξ_i value is reported.

  - For irregular oscillations, the top four periods of oscillation are reported based on the value of ξ_i.

2.3.3 Amplitude Prediction Method

Amplitude of oscillation is predicted from time series based on the analysis of segmented data and statistical feature extraction.

Step 1: Data Preprocessing: Initial preprocessing steps include denoising the data using a wavelet-based approach. Specifically, the signal is decomposed using the Daubechies 1 (db1) wavelet at level 1, and adaptive soft thresholding is applied to remove noise while preserving key signal features. The method leverages interval-dependent thresholding, allowing localized denoising suited for dynamic process signals with non-uniform characteristics. This ensures reliable amplitude estimation and accurate analysis in subsequent steps.

Step 2: Segmentation of Time-Series Data: The denoised time-series data is segmented into smaller intervals based on the period of oscillation. If the true values of the period of oscillation are already available, they can be used. Alternatively, the oscillation period can be estimated using the method described in section 2.3.2. The number of segments (N) is taken as the number of cycles determined using the formula $N = \frac{L}{f_s T_\mathrm{i}}$, where L is the signal length, f_s is the sampling frequency, and T_i is the estimated oscillation period. This segmentation allows for a localized examination of oscillatory behavior within each segment.

Step 3: Statistical Feature Extraction: For each segment j (where $j = 1, 2, \ldots, N$), the following key statistical features are computed from the original time-series data:

- **Mean Value (μ_j)**: The average value of each segment j.
- **Maximum Value ($\mathcal{M}_j$)**: The peak value within each segment.
- **Minimum Value ($\mathcal{N}_j$)**: The trough value within each segment, which, along with the maximum value, defines the amplitude range.
- **Absolute Deviation**: The differences between the mean and maximum/minimum values are calculated. Specifically, the absolute differences $|\mu_j - \mathcal{M}_j|$ and $|\mu_j - \mathcal{N}_j|$ are analyzed.

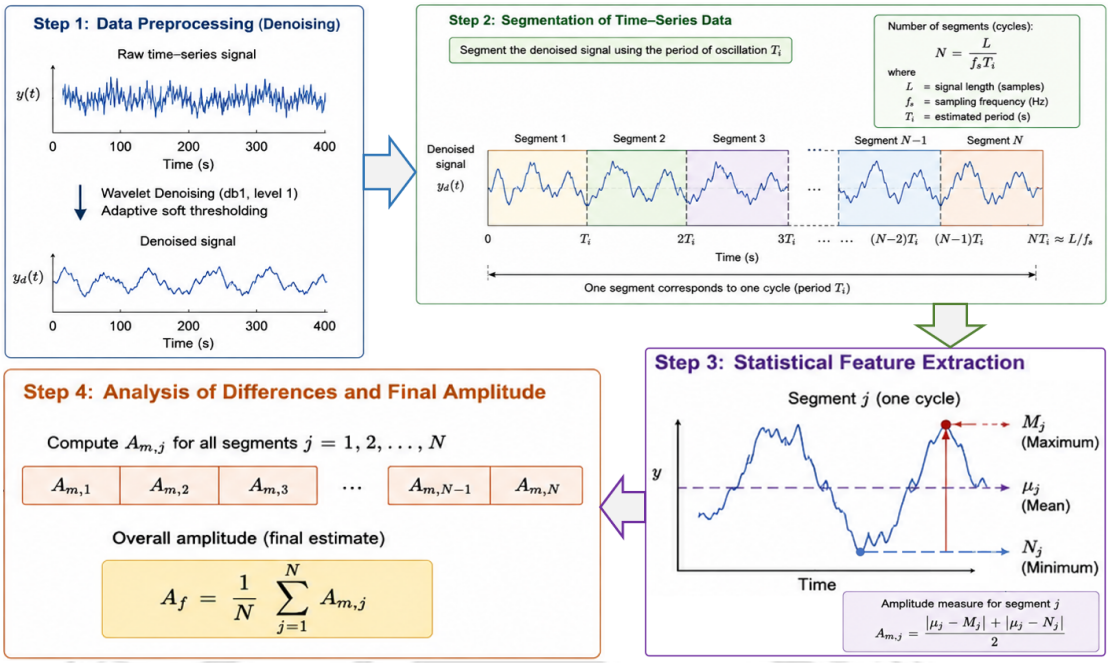


Figure 2.2: Overall framework of the proposed amplitude prediction methodology, including signal preprocessing, segmentation based on oscillation period, statistical feature extraction, and amplitude estimation.

Step 4: Analysis of Differences: The differences calculated in Step 3 are analyzed to reflects the amplitude dynamics across segments. The average of the absolute differences, defined as:

$$A_{m,j} = \frac{|\mu_j - \mathcal{M}_j| + |\mu_j - \mathcal{N}_j|}{2} \tag{2.2}$$

indicates the oscillation amplitude for each segment. Finally, the overall amplitude is reported as the mean of the amplitude measures:

$$A_f = \frac{1}{N} \sum_{j=1}^{N} A_{m,j} \tag{2.3}$$

where N is the number of segments. The overall workflow of the proposed amplitude prediction method is illustrated in Fig. 2.2.

2.3.4 Theoretical Basis for Cycle Count Threshold θ_c

To improve the reliability of frequency-domain features, a cycle count threshold θ_c is introduced to filter out frequency components that do not exhibit sufficient

periodicity over the duration of the signal. This threshold suppresses spurious spectral peaks caused by slow trends or noise, which are especially prevalent in industrial process data.

Let L denote the number of samples in the signal, and f_s the sampling frequency in Hz. For a candidate frequency component f, the number of observable cycles within the signal is given by:

$$n_{\text{cycles}} = \frac{L \cdot f}{f_s} \tag{2.4}$$

By imposing a lower bound θ_c on n_{cycles}, only components that complete at least θ_c full cycles over the signal duration are considered significant. Rearranging, the minimum detectable frequency is:

$$f_{\min} = \frac{\theta_c \cdot f_s}{L} \tag{2.5}$$

This expression shows that θ_c defines a tunable lower cutoff in the frequency domain. Increasing θ_c restricts detection to higher-frequency oscillations by requiring a greater number of cycles within the signal duration. This enhances robustness by filtering out low-frequency drifts and trends, but may inadvertently exclude valid low-frequency oscillations. Conversely, lower θ_c values allow the detection of long-period or weak oscillations but increase the risk of including spurious low-frequency components.

While a fixed θ_c offers consistency and interpretability across the dataset, it also introduces a signal-length-dependent minimum detectable frequency. As such, very low-frequency oscillations may not satisfy the cycle count requirement in shorter signals. Although this trade-off was acceptable for the objectives of this study, future work may explore adaptive strategies such as scaling θ_c with signal length or adjusting it based on spectral characteristics, to improve flexibility while maintaining robustness.

2.4 Results and Discussion

2.4.1 Training Neural Network

The three-layer feedforward neural network, as described in Section 2.3.1, was employed for oscillation detection, consisting of 400, 100, and 20 neurons in each layer. The model was trained using a dataset of 100,000 signals, which includes 39,720 non-oscillatory variables, 30,147 oscillatory variables, and 30,133 irregularly oscillatory variables. The test dataset contains 10,000 variables, with 3,985 non-oscillatory variables, 3,027 oscillatory variables, and 2,988 irregular oscillatory variables. The performance of the trained neural network on synthetic and industrial test datasets is discussed in the subsequent sections. Specifically, Section 3.4.1 presents illustrative case studies using a small set of representative time series to demonstrate the model's interpretability and accuracy in parameter estimation. Section 2.4.3 provides a large-scale quantitative evaluation on synthetic test data, assessing classification accuracy under different feature and threshold configurations. Finally, Section 2.4.4 examines the model's robustness on a real-world industrial benchmark dataset.

2.4.2 Illustrative Example

The performance and applicability of the proposed oscillation detection and parameter estimation method are demonstrated here using three different datasets representing three types of time-series data: non-oscillatory, regular oscillatory, and irregular oscillatory. This section highlights how the method differentiates between these behaviors and estimates the key parameters, such as the period of oscillation and amplitude.

As shown in Figure 2.3, the first subplot has no significant periodic patterns and is the selected non-oscillatory variable (Class 0). The second subplot depicts the regular oscillatory variable (Class 1) which is evident from the consistent oscillatory pattern, while the third subplot presents the irregular oscillatory variable

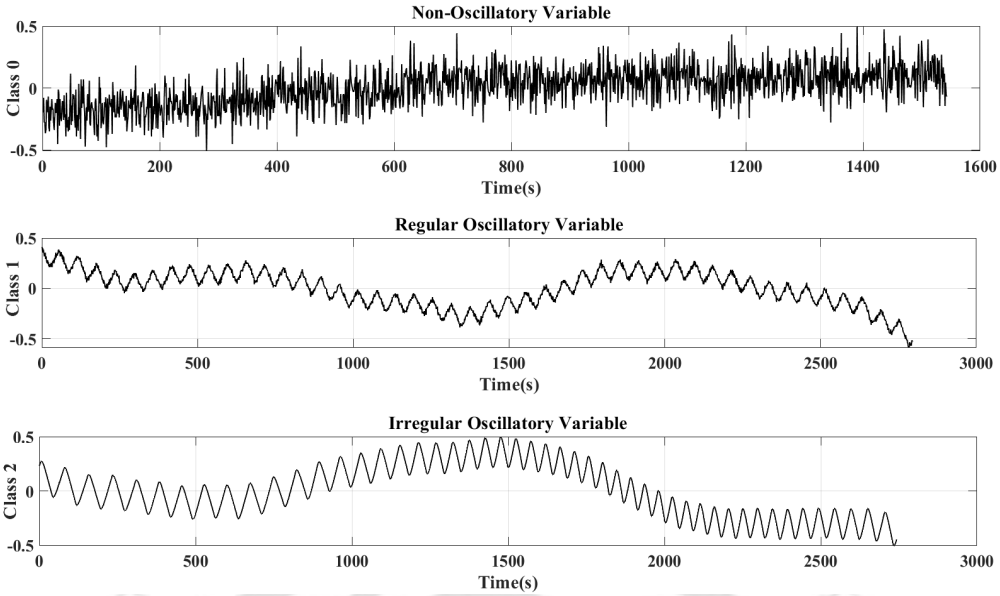


Figure 2.3: Visualization of Time Series Data: Class 0, Class 1, Class 2 (Class 2), which exhibits varying period of oscillation.

The frequency-domain analysis is presented in Figure 2.4, where the normalized FFT and FFT-ACF of the three datasets are compared. For the non-oscillatory data, no prominent frequency peaks are detected in either the FFT of the time-series or the FFT-ACF. Minor fluctuations due to noise appear, but no dominant peaks are observed, confirming the absence of oscillations. For the regular oscillatory data, the FFT shows a clear dominant frequency peak at approximately 0.01672, which corresponds to the known period of oscillation, and additionally, some peaks corresponding to lower frequencies are present that do not represent the dominant oscillation. The FFT-ACF yields a smoother spectrum, with the dominant peak also centered around 0.01672, confirming the result. For the irregular oscillatory data, multiple significant peaks are observed in the FFT, spanning a frequency range from approximately 0.01343 to 0.02332, reflecting the fluctuating nature of the oscillations. The FFT-ACF exhibits a similar frequency range, supporting these observations. These observations further demonstrate that while the FFT-ACF effectively denoises the spectrum and enhances peak clarity particularly in regular oscillations, it may attenuate informative dominant peaks in irregular cases. Therefore, leveraging both FFT and FFT-ACF jointly offers complementary advantages for robust and comprehensive frequency analysis.

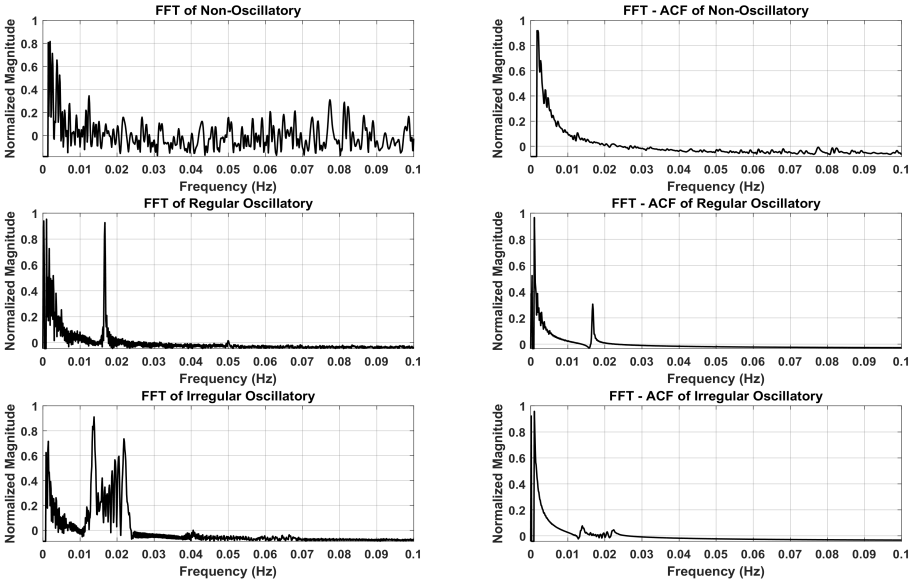


Figure 2.4: Comparison of FFT and FFT-ACF for Non-Oscillatory, Regular Oscillatory, and Irregular Oscillatory Variables.

2.4.2.1 Oscillation Detection for Illustrative Example

Oscillation detection is performed using the methodology outlined in Section 2.3.1. The frequency domain representations are generated by applying the FFT to both the normalized time-series data and the normalized ACF of the original data. The first half of the resulting magnitude spectrum are retained for analysis, as the second half is redundant.

Next, for each frequency spectrum (both FFT and FFT-ACF), the peak with the maximum magnitude is identified. The corresponding frequency is used to calculate the period of the oscillation, and number of cycles. If the calculated number of cycles does not meet the specified cycle count threshold (θ_c), the method searches for the next prominent peak in the spectrum, ensuring that only significant oscillations are considered. This process continues until a valid peak is identified, and for each valid peak, m points on either side of the peak are selected, resulting in $2m + 1$ features from both the FFT spectra and FFT-ACF spectra. These features are then integrated to create a comprehensive feature set resulting in $4m + 2$ features .

The integrated features are fed into the three-layer feedforward neural network,

Table 2.1: Comparison of Predicted Classes for Different Feature Counts and θ_c Values.

No. of feature (4m+2)	**Predicted class (θ_c= 2.5)**			**Predicted class (θ_c= 4)**		
	Non-Oscillatory	**Regular Oscillatory**	**Irregular Oscillatory**	**Non-Oscillatory**	**Regular Oscillatory**	**Irregular Oscillatory**
22 ($m = 5$)	0	0	0	0	1	2
42 ($m = 10$)	0	0	0	0	1	2
62 ($m = 15$)	0	0	0	0	1	2
202 ($m = 50$)	0	0	2	0	1	2
402 ($m = 100$)	0	0	2	0	1	2
802 ($m = 200$)	0	1	2	0	1	2
8194	0	1	2	0	1	2

which classifies the time-series data into three categories: non-oscillatory, regular oscillatory, and irregular oscillatory. The performance of the method is evaluated, and Table 2.1 summarizes the predicted classifications for non-oscillatory, regular oscillatory, and irregular oscillatory data types across various feature counts, as determined by the value of m. At lower feature counts (e.g., 22, 42, 62), the method struggles to accurately classify regular and irregular oscillatory signals, although non-oscillatory data is consistently predicted correctly for $\theta_c = 2.5$. As the number of features increases (e.g., 802, 8194), the classification accuracy for regular and irregular oscillations improves significantly. For $\theta_c = 4$, all predicted classes for non-oscillatory, regular oscillatory, and irregular oscillatory categories are correct across all feature counts. This is primarily due to the nature of Class 1 signals, which contain a slow drift component spanning more than 2.5 cycles but fewer than 4 cycles over the signal duration. When $\theta_c = 2.5$, this slow drift is not filtered out and may dominate the spectrum, leading to misidentification of the true oscillatory frequency. Increasing θ_c to 4 removes this drift, allowing the dominant frequency to be correctly identified and improving classification accuracy. In contrast, Class 2 signals contain a slower drift that spans approximately 1.5 cycles, which is filtered out at both $\theta_c = 2.5$ and $\theta_c = 4$, resulting in more accurate classification at both thresholds. These results highlight the effectiveness of the proposed method in accurately detecting and classifying oscillations based on frequency domain features.

2.4.2.2 Period of Oscillation Estimation for Illustrative Example

The proposed method as described in Section 2.3.2 was tested on two datasets representing regular and irregular oscillations presented in Figure 2.3 for the estimation of oscillation periods. The preprocessing step includes demeaning and detrending, followed by generating FFT-ACF spectrum. The frequency components with magnitudes greater than the magnitude threshold (θ_m) are selected for further analysis. For each selected frequency component, the period of oscillation T_i and the number of oscillation cycles $n_{\text{cycles},i}$ are calculated. The Oscillation Ranking Index (ξ_i) is then computed for each distinct frequency component as defined in the methodology. Table 2.2 and Table 2.3 presents the top four candidates for the period of oscillation and corresponding features extracted from the FFT-ACF of the regular and irregular oscillatory variable, respectively. The results are arranged based on the highest ξ_i values.

Table 2.2: Summary of Detected Period of Oscillation for Regular Oscillatory Data with True Period 59.96.

Period of Oscillation (T_i)	**Number of Oscillation Cycles ($n_{\text{cycles,i}}$)**	**Frequency components (f_i) [Hz]**	**Magnitude ($S(f_i)$)**	**Oscillation Ranking Index (ξ_i)**	**Length of Time Series (L)**
59.809	46.732	0.0167	94.241	2.483	2795
55.741	50.142	0.018	16.557	0.089	2795
1639.3	1.705	0.00061	410.3	0.063	2795
53.533	52.211	0.0187	13.159	0.0604	2795

Table 2.3: Summary of Detected Period of Oscillation for Irregular Oscillatory Data with True Period Around 56.249.

Period of Oscillation (T_i)	**Number of Oscillation Cycles ($n_{\text{cycles,i}}$)**	**Frequency components (f_i) [Hz]**	**Magnitude ($S(f_i)$)**	**Oscillation Ranking Index (ξ_i)**	**Length of Time Series (L)**
44.763	61.279	0.0220	16.135	0.13	2743
71.839	38.183	0.0140	20.818	0.084	2743
48.473	56.588	0.0210	13.032	0.072	2743
50.891	53.9	0.0200	11.476	0.051	2743

Based on the selected frequency components and extracted features from both regular and irregular oscillatory datasets, the proposed method successfully identified significant oscillations for both regular and irregular signals. For the regular

oscillatory dataset, although we have reported four values in Table 2.2, as per the methodology, the detected period of oscillation is 59.809 which closely matches the true period (59.96). This highlights the precision of the method in estimating the period of oscillation for regular oscillations. Note that although the period of $T_i = 1639.3$ has the highest magnitude (410.3), the corresponding ξ_i value is quite small and thus, it was not reported as the top candidate. This confirms that while the method detects multiple T_i values, it correctly prioritizes the time period closest to the true time period, demonstrating the method's ability to identify the most relevant oscillatory behavior in the data. On the other hand, for the irregular dataset, the true reported time period is 56.249 (which is an approximation given that the signal has irregular oscillation), and the detected periods range from 44.7 to 71.8, reflecting the inherent irregularity of the oscillations. Note that the average of the reported periods is 53.99 which matches with the true period. However, if the variability in periods is high, the average will not be an effective measure to assess the performance of the tool. Hence, only the predicted periods are presented for all case studies instead of the average. This demonstrates the method's ability to capture varying oscillation periods in irregular datasets. Furthermore, in practical settings, downstream users may incorporate additional domain knowledge, such as oscillation history, fault records, or loop-specific behavior, to prioritize which oscillatory modes warrant intervention.

2.4.2.3 Amplitude Prediction for Illustrative Example

The amplitude prediction method outlined in Section 2.3.3 was applied to the time-series data from both regular and irregular oscillatory signals in the illustrative example. The denoised data was first segmented based on the estimated or known period of oscillation. For irregular oscillatory signals that exhibit multiple distinct periods, the segmentation is performed individually for each period. Specifically, the signal is divided into segments based on the estimated boundaries of each oscillation cycle. The amplitude is then computed separately for each segment,

resulting in a set of amplitudes corresponding to each identified period. Statistical features, such as the μ_j, $\mathcal{M}_j$, $\mathcal{N}_j$, were computed for each segment j, followed by the calculation of the amplitude for each segment. The final amplitude was determined as the average of the segment amplitudes. Table 2.4 presents the predicted amplitudes for each data type in comparison with true values obtained through visual inspection.

Key results include:

- **Regular Oscillatory Data:** The predicted amplitude was $A_f \approx 0.096$, closely matching the true amplitude.
- **Irregular Oscillatory Data:** The predicted amplitude was

$$A_f \approx [0.1338, 0.1480, 0.1392, 0.1392]$$

again close to the true value although there is higher variance due to the irregular nature of the oscillations.

Table 2.4: Predicted and True Amplitudes for the Illustrative Example.

Data Type	**Predicted Amplitude, A_f**	**True Amplitude (From Visual Inspection)**
Regular Oscillatory	0.0956	0.1
Irregular Oscillatory	[0.1338, 0.1480, 0.1392, 0.1392]	0.15

Table 2.5 shows the overall prediction summary for the illustrative example, integrating oscillation detection, time period prediction, and amplitude estimation.

Table 2.5: Summary of Prediction Results for the Illustrative Example.

Data Type	**Oscillation Detection**		**Period of Oscillation**		**Amplitude**	
	True Class	**Predicted Class ($\theta_c = 4$)**	**True Period**	**Predicted Period (T_i)**	**True Amplitude**	**Predicted Amplitude (A_f)**
Non-Oscillatory	0	0	—	—	—	—
Regular Oscillatory	1	1	59.948	59.8	0.1	0.0956
Irregular Oscillatory	2	2	56.249	[44.8, 71.8, 48.5, 50.9]	0.15	[0.1338, 0.1480, 0.1392, 0.1392]

2.4.2.4 Reporting Multiple Modes in Irregular Oscillations

To further support the decision to report multiple periods and amplitudes for irregular signals, we constructed a synthetic signal composed of three superimposed sinusoidal components with known frequencies, periods, and amplitudes.

As shown in Table 2.6, the proposed method successfully detects all three dominant oscillatory components, closely matching both the true periods and amplitudes. However, computing a simple average of the detected periods (51.59 s) does not meaningfully represent any of the actual oscillatory modes, thereby illustrating the risk of collapsing rich oscillatory behavior into a single value. Similarly, reporting a single average amplitude would fail to reflect the variation in oscillatory strength across the components. These findings confirm that the strategy of reporting the top few dominant modes yields a more informative and reliable characterization of irregular signals.

Table 2.6: Ground Truth vs. Detected Periods and Amplitudes for Synthetic Signal

Component	Frequency (Hz)	True Period (s)	True Amplitude	Detected Period (s)	Detected Amplitude
1	0.03	33.3	0.6	32.77	0.83
2	0.01	100	1.0	102.4	1.3
3	0.05	20	0.4	19.6	0.57

2.4.3 Synthetic Data Results

2.4.3.1 Oscillation Detection

The approach employed for detecting oscillations follows the methodology outlined in Section 2.3.1. A neural network model is trained using feature subsets derived from the FFT and FFT-ACF spectra of the time-series data, with sensitivity analysis determining the optimal window size (m) and θ_c.

The effectiveness of the proposed method was evaluated on the test dataset which includes 3,985 non-oscillatory variables, 3,027 oscillatory variables, and 2,988 irregularly oscillatory variables. Table 2.7 summarizes the precision, recall, and accuracy results for various feature configurations. For clarity on how these met-

Table 2.7: Results of Oscillation Detection and Sensitivity Analysis Using Combinations of FFT and FFT-ACF Features for $\theta_c = 0$.

No. of Features $(4m+2)$	Confusion Matrix	Accuracy	Precision			Recall			Time
			Class 0	Class 1	Class 2	Class 0	Class 1	Class 2	
22 $(m=5)$	$\begin{bmatrix}3709 & 70 & 206\\ 1038 & 1919 & 70\\ 1558 & 38 & 1392\end{bmatrix}$	70.20%	0.588	0.947	0.835	0.931	0.634	0.466	44.8s
42 $(m=10)$	$\begin{bmatrix}3425 & 106 & 454\\ 797 & 2147 & 83\\ 1336 & 94 & 1558\end{bmatrix}$	71.30%	0.616	0.915	0.744	0.859	0.709	0.521	42.3s
62 $(m=15)$	$\begin{bmatrix}3478 & 33 & 474\\ 785 & 2134 & 108\\ 1232 & 46 & 1710\end{bmatrix}$	73.22%	0.633	0.964	0.746	0.873	0.705	0.572	36.2s
202 $(m=50)$	$\begin{bmatrix}3778 & 15 & 192\\ 355 & 2437 & 235\\ 530 & 129 & 2329\end{bmatrix}$	85.44%	0.810	0.944	0.845	0.948	0.805	0.779	56.2s
402 $(m=100)$	$\begin{bmatrix}3860 & 7 & 118\\ 194 & 2685 & 148\\ 325 & 53 & 2610\end{bmatrix}$	91.55%	0.881	0.978	0.908	0.969	0.887	0.873	44.9s
802 $(m=200)$	$\begin{bmatrix}3928 & 13 & 44\\ 100 & 2848 & 79\\ 147 & 43 & 2798\end{bmatrix}$	95.74%	0.941	0.981	0.958	0.986	0.941	0.936	74.4s
8194	$\begin{bmatrix}3951 & 16 & 18\\ 99 & 2867 & 61\\ 136 & 76 & 2776\end{bmatrix}$	95.94%	0.944	0.969	0.972	0.991	0.947	0.929	259s

rics are computed, please refer to Appendix A.

The results highlight a notable reduction in the number of features needed to achieve high accuracy. As the feature count increases from 11 ($m = 5$) to 8194 (full data), the accuracy steadily improves from 70.20% to 95.94% on the test set. However, there is no significant gain in accuracy by increasing the number of features from 802 to 8194. Utilizing all 8194 features results in a slightly higher precision for Class 0 and Class 2, as well as a slightly higher recall for Class 0 and Class 1. However, it leads to a decrease in precision for Class 1 and a decline in recall for Class 2. Overall, there is no significant improvement in class-wise precision or recall by adding more features than 802. By selecting a subset of features around significant peaks in the FFT spectrum, we can achieve comparable accuracy with fewer features.

The configuration with 802 features achieved an accuracy of 95.74%, demonstrating that reducing the feature set by a factor of ten (from 8194 to 802) main-

tains a high level of performance while significantly lowering computational complexity. This reduction is especially advantageous for chip-based implementations, where computational resources are constrained. Additionally, the total time/computational time (including both model training and testing) for the model with 802 features was approximately 74.4 seconds, compared to over four minutes (259 seconds) when using 8194 features. It is important to note that these computational times exclude the duration for hyperparameter tuning, as no optimization was performed. The same network architecture was used for all values of m, but hyperparameter optimization would likely enhance the performance of each model.

We observed that the accuracy tends to be lower when fewer features are used, compared to when a larger set of features is employed for $\theta_c = 0$. This could occur because the peak identified in the FFT and FFT-ACF spectra might correspond to a very low frequency, which may not represent the dominant frequency of the variable. To ensure the selection of the dominant peak, different values of θ_c are evaluated based on the number of features, and sensitivity analysis is then performed for each configuration.

Figure 2.5 illustrates how precision and recall vary across all classes for different values of θ_c and numbers of features when both FFT and FFT-ACF are used as input features. For lower numbers of features (e.g., 22, 42, 62, 202, 402), class-wise precision and recall generally improve with increasing θ_c. Higher thresholds, such as $\theta_c = 4$ or $\theta_c = 5$, result in better precision and recall across most classes, demonstrating their effectiveness in capturing relevant oscillations. For larger feature sets (e.g., 802, 8194), the impact of θ_c on precision and recall is less significant. For moderately large feature sets (e.g., 802), any $\theta_c \geq 2$ provides good precision and recall. However, for very large feature sets (8194), high θ_c values may reduce class-wise precision and recall. A moderate range, such as $\theta_c = 2$ to $\theta_c = 4$, appears to offer a good balance for such configurations.

Table 2.8 presents the accuracy results for different values of θ_c. Lower values of θ_c

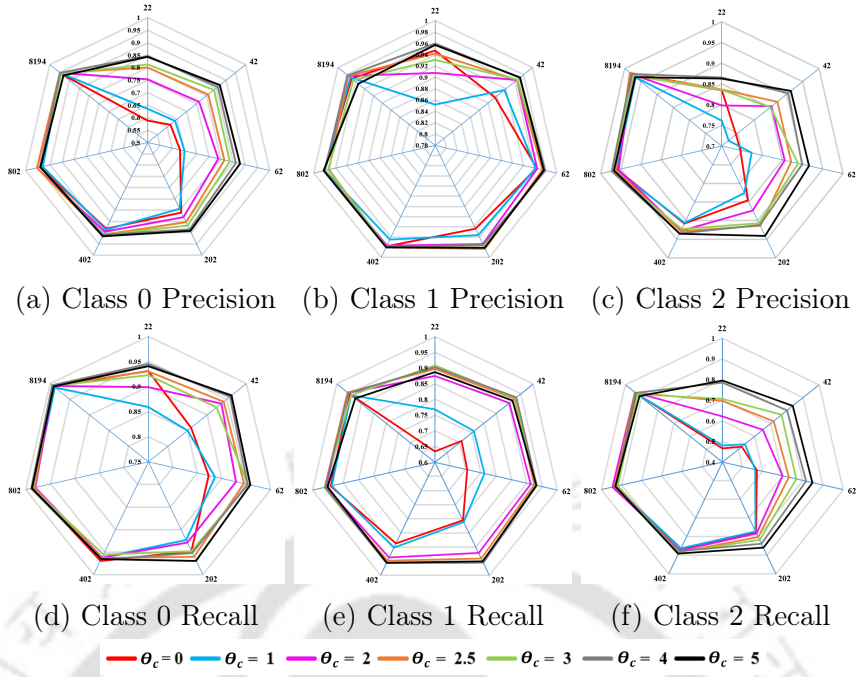


Figure 2.5: Variation in Performance Metrics (Precision and Recall) of the Neural Network Classifier Using FFT and FFT-ACF Features Across Different Classes for θ_c. Subfigures (a), (b), and (c) Show Precision for Classes 0, 1, and 2 Respectively, While (d), (e), and (f) Show Recall for the Same Classes.

Table 2.8: Accuracy Results for Subsets of FFT and FFT-ACF Features Using Different θ_c Values.

No. of Features $(4m+2)$	Accuracy (%)						
	$\theta_c = 0$	$\theta_c = 1$	$\theta_c = 2$	$\theta_c = 2.5$	$\theta_c = 3$	$\theta_c = 4$	$\theta_c = 5$
22 $(m = 5)$	70.20	71.85	80.88	83.77	85.38	88.40	88.08
42 $(m = 10)$	71.30	73.02	84.08	85.45	87.87	90.31	91.02
62 $(m = 15)$	73.22	75.21	85.59	86.88	89.08	90.60	91.53
202 $(m = 50)$	85.44	84.43	88.32	89.51	91.03	91.90	93.09
402 $(m = 100)$	91.55	91.39	92.82	92.74	93.27	93.67	93.99
802 $(m = 200)$	95.74	95.37	96.27	96.63	95.91	96.16	96.00
8194	95.94	95.91	96.52	96.48	95.76	96.57	94.79

(0 and 1) result in generally lower performance across all feature sets. Moderate values of θ_c (2, 2.5, and 3) lead to significant improvements, with $\theta_c = 2.5$ and 802 features yielding the highest accuracy (96.63%), highlighting the importance of filtering out peaks with very short periods. As θ_c increases to higher values (4 and 5), accuracy improves for smaller feature sets (22, 42, 62, 202, 402) but starts to decrease slightly for larger feature sets (802, 8194). This suggests that while filtering out irrelevant peaks is beneficial, excessive filtering can be detrimental for larger feature sets. For smaller feature sets, accuracy increases with higher θ_c values, but for larger feature sets, the improvement is marginal. The marginal

accuracy gain does not justify the additional filtering of peaks with higher thresholds, especially for very large feature sets. For instance, for $m = 200$, the accuracy with $\theta_c = 2.5$ is 96.63%, while for $\theta_c = 5$, it drops slightly to 96.00%.

In summary, the primary objective of the algorithm is to detect oscillations, making the recall for Classes 1 and 2 particularly important. The highest recall values of 0.96 for Class 1 and 0.945 for Class 2 were achieved using 802 features with a threshold of $\theta_c = 2.5$. It is also noteworthy that the accuracy peaked at $\theta_c = 2$ and $\theta_c = 2.5$ for 802 number of features.

In addition to recall, the False Positive Rate (FPR) and False Negative Rate (FNR) were considered as critical evaluation metrics to assess the performance of the model. These metrics provide valuable insights into the model's classification behavior, beyond overall accuracy. Specifically:

- **FPR** represents the proportion of negative instances that were incorrectly classified as positive. This metric is crucial in scenarios where false alarms must be minimized. The formula for FPR is given by:

$$\text{FPR}_k = \frac{\text{FP (False Positives)}}{\text{FP (False Positives)} + \text{TN (True Negatives)}} \tag{2.6}$$

  where k denotes the class ($k = 0, 1, 2$).

- **FNR** represents the proportion of positive instances that were misclassified as negative. In the context of oscillation detection, a high FNR means that important oscillations could be missed. The formula for FNR is:

$$\text{FNR}_k = \frac{\text{FN (False Negatives)}}{\text{FN (False Negatives)} + \text{TP (True Positives)}} \tag{2.7}$$

Both FPR and FNR are important matrices, as a high FPR can lead to false alarms, while a high FNR can result in missing significant oscillations. We consider the following confusion matrix to illustrate the calculation of the FPR and FNR.

$$\begin{bmatrix} 3908 & 16 & 61 \\ 64 & 2894 & 69 \\ 124 & 39 & 2825 \end{bmatrix}$$

For Class 0, the True Positives (TP) are 3908, the False Negatives (FN) are $16 + 61 = 77$, the False Positives (FP) are $64 + 124 = 188$, and the True Negatives (TN) are the sum of $2894 + 69 + 39 + 2825$, yielding 5827. Using these values and applying Equations (2.6) and (2.7), the False Positive Rate (FPR_0) is calculated as 0.0313 (3.13%), and the False Negative Rate (FNR_0) is 0.0193 (1.93%).

For Class 1, the True Positives are 2894, the False Negatives are $64 + 69 = 133$, the False Positives are $16 + 39 = 55$, and the True Negatives are $3908 + 61 + 124 + 2825 = 6918$. Using Equations (2.6) and (2.7), the False Positive Rate (FPR_1) is calculated as 0.0079 (0.79%), and the False Negative Rate (FNR_1) is 0.0439 (4.39%).

Similarly, for Class 2, using the corresponding values for True Positives (2825), False Negatives ($124 + 39 = 163$), False Positives ($61 + 69 = 130$), and True Negatives ($3908 + 16 + 64 + 2894 = 6882$), the FPR_2 is 0.0185 (1.85%), and the FNR_2 is 0.0546 (5.46%).

To compute the weighted average FPR and FNR, we consider the class-wise contribution based on the number of variables in each class. Specifically, for each metric, the weighted average is calculated as:

$$\text{Weighted Average} = \sum_{k=0}^{2} \left(\frac{N_k}{N} \cdot \text{Metric}_k \right) \tag{2.8}$$

where:

- N_k is the number of variables in class k,
- $N = N_0 + N_1 + N_2$ is the total number of variables,
- Metric_k is either FPR or FNR for class k.

Table 2.9: FPR and FNR Percentages Using FFT and FFT-ACF Features for Different θ_c Values.

No. of Features	CCT	Class 0		Class 1		Class 2		Weighted Average	
		FPR (%)	FNR (%)	FPR (%)	FNR (%)	FPR (%)	FNR (%)	Weighted FPR(%)	Weighted FNR(%)
802	$\theta_c = 0$	4.11	1.43	0.80	5.91	1.75	6.36	2.41	4.26
	$\theta_c = 1$	4.92	1.58	0.83	6.34	1.55	6.96	2.68	4.63
	$\theta_c = 2$	3.13	1.93	0.79	4.39	1.85	5.46	2.04	3.73
	$\theta_c = 2.5$	3.23	1.76	1.18	3.96	1.51	6.43	2.1	3.82
	$\theta_c = 3$	4.22	1.43	1.13	3.77	1.08	7.97	2.35	4.09
	$\theta_c = 4$	4.16	1.10	0.83	4.13	1.08	7.20	2.23	3.84
	$\theta_c = 5$	4.17	1.18	0.77	4.89	1.35	6.86	2.3	4
8194	$\theta_c = 0$	3.91	0.85	1.32	5.29	1.13	7.10	2.30	4.06
	$\theta_c = 1$	3.79	1.18	1.18	5.81	1.41	6.22	2.29	4.09
	$\theta_c = 2$	3.62	0.83	0.98	4.59	0.88	5.89	2	3.48
	$\theta_c = 2.5$	3.52	0.48	0.96	4.33	0.67	5.89	1.9	3.26
	$\theta_c = 3$	4.02	0.65	1.49	5.81	1.11	7.43	2.39	4.24
	$\theta_c = 4$	3.61	0.48	0.95	4.72	0.86	6.06	1.98	3.43
	$\theta_c = 5$	4.84	0.90	1.95	7.47	1.34	8.67	2.92	5.21

Table 2.9 presents the FPR and FNR percentages for various θ_c values. For Class 1 using 802 features and $\theta_c = 2$, the FPR was low at 0.79%, indicating that the model rarely misclassified negative instances as positive. In summary, $\theta_c = 2$ with 802 features provides optimal performance, especially for Class 1, with a low FPR and manageable FNR across all classes.

In this study, a fixed neural network architecture was employed across all experiments involving different values of the feature selection parameter m, leading to varying input dimensions ranging from 22 to 802 features. This design choice ensured a consistent basis for evaluating the impact of different feature sets on model performance. While the network demonstrated robust performance across this wide range, we acknowledge that a more tailored approach, where the network size is adapted to the input dimensionality, could potentially reduce the risk of underfitting for high-dimensional inputs or overfitting in lower-dimensional cases. Jointly optimizing the network architecture along with feature selection may further enhance model efficiency and predictive performance, and this represents a promising direction for future work.

Through effective feature engineering and selection, the neural network that incorporates 802 input features, derived from both the FFT spectra and the FFT-ACF spectra, with a θ_c threshold of 2 or 2.5, proved to be the optimal choice, offering

the best balance between accuracy, weighted FPR, weighted FNR and computational efficiency. Additionally, a trade-off between accuracy and computational time was observed; for scenarios where lower computational time is a priority, fewer features can be employed, albeit at the cost of reduced accuracy.

To further support the robustness of the proposed approach, we performed a 10-fold stratified cross-validation on the synthetic dataset. This allowed us to quantify the variability in prediction performance across different data splits. The model with 802 features and θ_c of 2.5 achieved a mean accuracy of 0.9642, with a standard deviation of 0.0019, and a 95% confidence interval of ± 0.0012, demonstrating consistent and reliable performance.

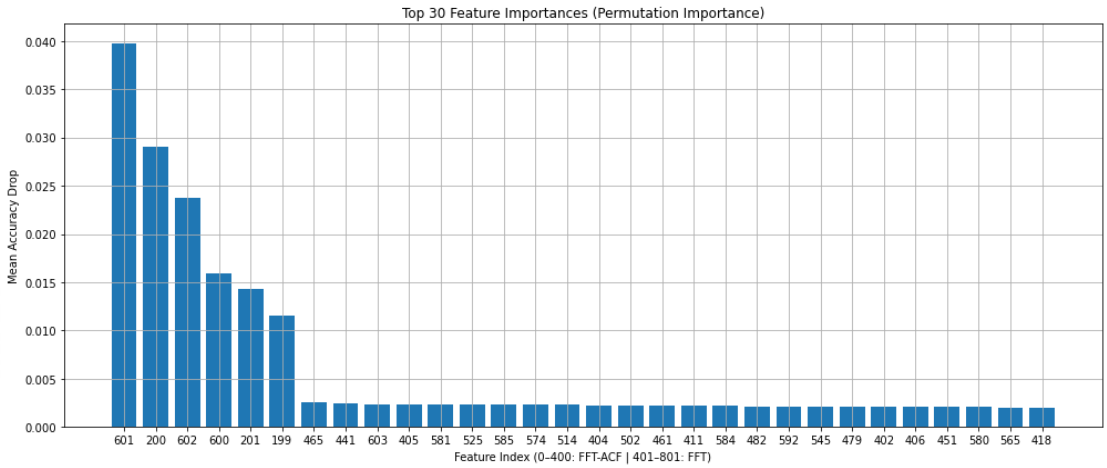


Figure 2.6: Top 30 Features Ranked by Permutation Feature Importance. Feature Indices 0–400 Represent FFT-ACF Features, and 401–801 Represent FFT Features.

To gain further insight into the decision-making process of the neural network, we conducted a Permutation Feature Importance (PFI) analysis. This was aimed at identifying which of the 802 input features contributed most significantly to the classification performance. As described earlier, the input vector comprises 401 FFT-ACF features and 401 FFT features, where index 200 in each block corresponds to the dominant frequency region.

We measured the change in model accuracy on the test set when individual features were randomly shuffled. The results, presented in Figure 2.6, reveal that the most critical features are centered around the dominant spectral peak:

- Feature 601 (center of FFT block) led to the highest accuracy drop of 3.98%.
- Feature 200 (center of FFT-ACF block) caused a 2.90% drop.
- Neighbouring features like 600, 602, and 199 also showed considerable importance.
- The top six features alone accounted for a cumulative accuracy drop of 13.4% when individually permuted.

These findings indicate that the model places strong emphasis on the height, sharpness, and symmetry of the dominant spectral peak, leveraging both magnitude and autocorrelation-based information for robust classification. This enhances the interpretability of the proposed model and reinforces its practical relevance in real-world applications.

Note: For all evaluations on the industrial dataset, we used the pre-trained model obtained from the synthetic dataset. This model was trained using 802 input features—derived from both FFT and FFT-ACF spectra—and employed a threshold value of $\theta_c = 2.5$. No retraining or parameter tuning was performed on the industrial data, as the intention is to demonstrate the model's direct applicability in real-world industrial scenarios.

2.4.3.2 Period of Oscillation Estimation for Synthetic Dataset

The following section evaluates the method outlined in Section 2.3.2 for predicting period of oscillation across a set of 10,000 test datasets, which included 3,027 oscillatory variables and 2,988 irregularly oscillatory variables. Each dataset was labeled with true period of oscillation, and the method's predictions were compared to these true values.

Table 2.10 shows the performance of the method for predicting regular and irregular oscillation time periods across the test dataset. For regular oscillations, the method achieved a correct prediction rate of 99.3%, with 3005 out of 3,027 regular oscillations correctly predicted within a ±5% tolerance. This indicates

that the method is highly effective in accurately predicting the periods of regular oscillations.

Table 2.10: Summary of Correct Predictions for Period of Oscillation.

Oscillation Type	**Total Variables**	**Correct Prediction**			
		±5%	**±10%**	**±12%**	**±15%**
Regular Oscillation	3027	3005 (99.27%)	3020 (99.77%)	3021 (99.80%)	3023 (99.87%)
Irregular Oscillation	2988	1582 (52.9%)	2266 (75.8%)	2428 (81.3%)	2598 (86.9%)

The performance of the method for predicting irregular oscillation periods on the test dataset indicates that prediction accuracy improves as the tolerance for period estimation increases. The results are based on the top four highest ξ_i values, where the corresponding periods of oscillation are compared with the true values to calculate prediction accuracy at different tolerance levels. Specifically, if any one of the 4 predicted periods falls within the specified tolerance range of the true value, it is considered a correct prediction. Specifically, the percentage of correct predictions within ±5% is 52.9%, and this increases to 86.9% within ±15%. This result aligns with the inherently complex characteristics of irregular oscillations, where the period is varying and it is often difficult to mention a single true period of oscillation. This demonstrates that the method is effective in reliably detecting and characterizing irregular oscillations, especially when more flexibility is allowed in estimating the period of oscillation.

Additionally, we conducted a comparative evaluation using the same period estimation method but with an alternative ranking based solely on spectral magnitude ($\xi_i = S(f_i)$). The results, included in Table 2.11, show that the proposed ORI metric consistently outperforms the magnitude-only variant, particularly for irregular signals.

Table 2.11: Summary of Correct Prediction Using Magnitude-Only Ranking Metric ($\xi_i = S(f_i)$).

Oscillation Type	**Total Variables**	**Correct Prediction**			
		±5%	**±10%**	**±12%**	**±15%**
Regular Oscillation	3027	2790 (92.17%)	2805 (92.66%)	2806 (92.70%)	2807 (92.73%)
Irregular Oscillation	2988	1294 (43.31%)	1852 (62.00%)	2020 (67.60%)	2190 (73.30%)

To further highlight the effectiveness of the proposed methodology, a comparison was made with an alternative approach where the oscillation time period was

predicted directly based on the top three peaks from the FFT spectrum. For oscillatory signals, the FFT spectrum typically exhibits distinct peaks corresponding to dominant frequencies, and the associated time periods can be estimated by taking the inverse of these frequencies [121]. In this alternative approach, the first, second, and third most prominent peaks were identified, and their corresponding frequencies were used to estimate the time periods. The predicted periods were compared with the true periods, and accuracy was computed by checking whether any of the predicted values fell within a ±5% tolerance of the true period. For regular oscillations, this method achieved a lower accuracy of 79.39%, and for irregular oscillations, the accuracy dropped drastically to 13.69%. In contrast, the proposed method achieved a significantly higher accuracy of 46.18% for irregular oscillations under the same evaluation criteria. This highlights that relying solely on dominant FFT peaks fails to capture irregular patterns, whereas the proposed method demonstrates greater robustness.

2.4.3.3 Oscillation Amplitude Estimation for Synthetic Dataset

In this section, we evaluate the performance of the proposed amplitude prediction method on a selection of 8 variables from the test dataset, as shown in Figure 2.7. Since the actual amplitude values were unavailable for the test data, the validation was conducted through visual inspection using a segment-wise approach. Specifically, the time-series data for each variable was plotted and randomly segmented. Each segment, typically spanning a few oscillation cycles, was closely examined to manually identify the maximum, minimum, and midpoint (mean) values for each cycle. This process was repeated across multiple segments to assess the consistency and range of amplitude values throughout the time series. The amplitude prediction approach, outlined in Section 2.3.3, was applied to each of the selected variables, and the predicted amplitude values were compared with the observed oscillatory patterns. The test dataset included a range of oscillatory behaviors, providing an opportunity to assess the method's accuracy across different signal

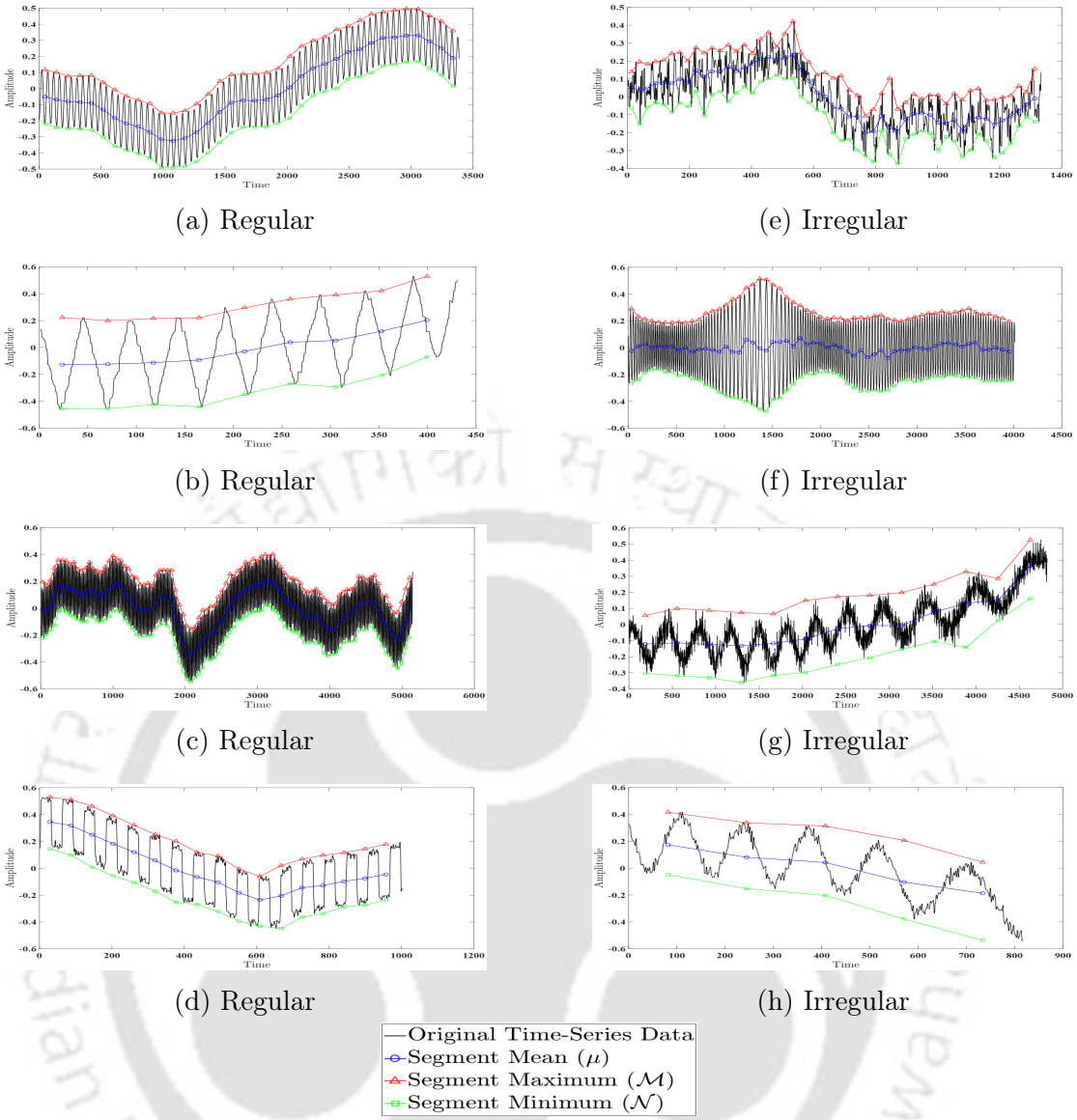


Figure 2.7: Regular and Irregular Time-Series Data Plots for Amplitude Estimation.

characteristics. The amplitude prediction was executed using statistical feature extraction on the denoised, segmented signals, with key metrics such as μ_j, $\mathcal{M}_j$, $\mathcal{N}_j$ being calculated. The results, reported as the overall amplitude (A_f), are summarized in Table 2.12.

In Subfigures (a), (b), (c), and (d), the predicted amplitude is within or very close to the actual range based on visual inspection. In Subfigures (e), (f), (g), and (h), multiple amplitude values are predicted for each test dataset based on the predicted period of oscillation. Despite multiple predictions, all predicted values are found to be within the expected range for the actual amplitude. This indicates

that the model is consistent in predicting across slightly varied inputs of period of oscillation. Overall, the proposed method gives good accuracy, with predicted amplitudes well within the expected ranges across all subfigures. The ability to handle variations in the data, as seen in Subfigures (e) to (h), without significant deviations from the actual range further underscores the robustness of the approach. Additionally, as for irregular oscillations, the method estimates multiple period–amplitude pairs, with each amplitude quantifying the strength of the corresponding oscillatory mode in the signal. This information is valuable in industrial settings, as it enables downstream users such as control engineers or plant operators to assess which oscillatory components may be most dominant or disruptive. A higher-amplitude mode may signal a fault or disturbance with greater influence on control loop performance, whereas lower-amplitude modes may be less critical. Moreover, based on prior process knowledge, historical data, or the dynamic characteristics of the control system, users can interpret and prioritize these oscillations more effectively. Providing multiple amplitudes thus supports informed fault prioritization, root-cause analysis, and targeted troubleshooting in complex process environments.

Table 2.12: Amplitude Prediction Results for Test Variables.

Subfigure	**Actual T_i**	**Predicted T_i**	**A_f (Visual Inspection)**	**Predicted A_f**
(a)	47.2	47.4	0.14 – 0.17	0.164
(b)	48.5	48.1	0.30 – 0.35	0.324
(c)	18.1	18.1	0.17 – 0.23	0.197
(d)	60.3	60.2	0.17 – 0.22	0.21
(e)	24.8	[19 26.3 11.8 14.3]	0.10 – 0.15	[0.12 0.135 0.09 0.10]
(f)	26.4	[24.4 26.6 28.8 20.8]	0.20 – 0.30	[0.268 0.269 0.27 0.267]
(g)	426.6	[357.1 322.6 588.2 500]	0.15 – 0.24	[0.195 0.18 0.22 0.20]
(h)	154.8	[149.3 120.5 102 90.1]	0.20 – 0.35	[0.26 0.24 0.23 0.21]

To quantitatively assess the accuracy of the amplitude predictions, we computed the Root Mean Square Error (RMSE) between the predicted amplitude values and the midpoints of the visually inspected amplitude ranges. RMSE is defined as:

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(A_{f,i} - A_i^{\text{ref}}\right)^2} \tag{2.9}$$

where $A_{f,i}$ is the predicted amplitude for the i-th time series, A_i^{ref} is the corre-

sponding reference amplitude, and N is the number of test instances. The resulting RMSE was found to be 0.0175, indicating a high level of prediction accuracy across the evaluated time series.

2.4.4 Performance Analysis on Industrial Benchmark Data

The industrial benchmark dataset introduced by Jelali and Huang [120] was used to assess the performance of the proposed method. Process outputs (PV) were extracted from the dataset of 93 control loops, primarily from chemical processes, including flow control (FC), level control (LC), temperature control (TC), pressure control (PC), analyser control (AC), and gauge control (GC) loops. The control loops exhibit a broad spectrum of operational behaviors and fault types. While many loops maintain constant SP, others operate under varying SP due to cascade or supervisory control strategies. Nonlinearities such as valve stiction, controller output saturation, and deadzones are common. Some loops operate in manual (open-loop) mode, while others are affected by heavy quantization, non-stationary trends, or measurement noise. For example, loops such as POW 3 and CHEM 33 are significantly influenced by noise or external disturbances, whereas MIN 1 and PAP 10 demonstrate multiple oscillation modes or evolving dynamics over time [120].

Signal lengths vary substantially across the industrial dataset, ranging from 200 to 277,115 time steps, reflecting diverse sampling durations and process dynamics. In comparison, the synthetic dataset was generated under controlled conditions with predefined oscillation periods, waveform types, and noise levels. Unlike the synthetic data, the industrial dataset exhibits unstructured variability, overlapping dynamic behaviors, inconsistent sampling characteristics, and measurement noise arising from multiple sources. These complexities make it a challenging yet essential testbed for evaluating the robustness and generalizability of oscillation detection algorithms [120]. For oscillation detection, we used a model trained solely on the synthetic dataset, utilizing 802 features extracted from both the

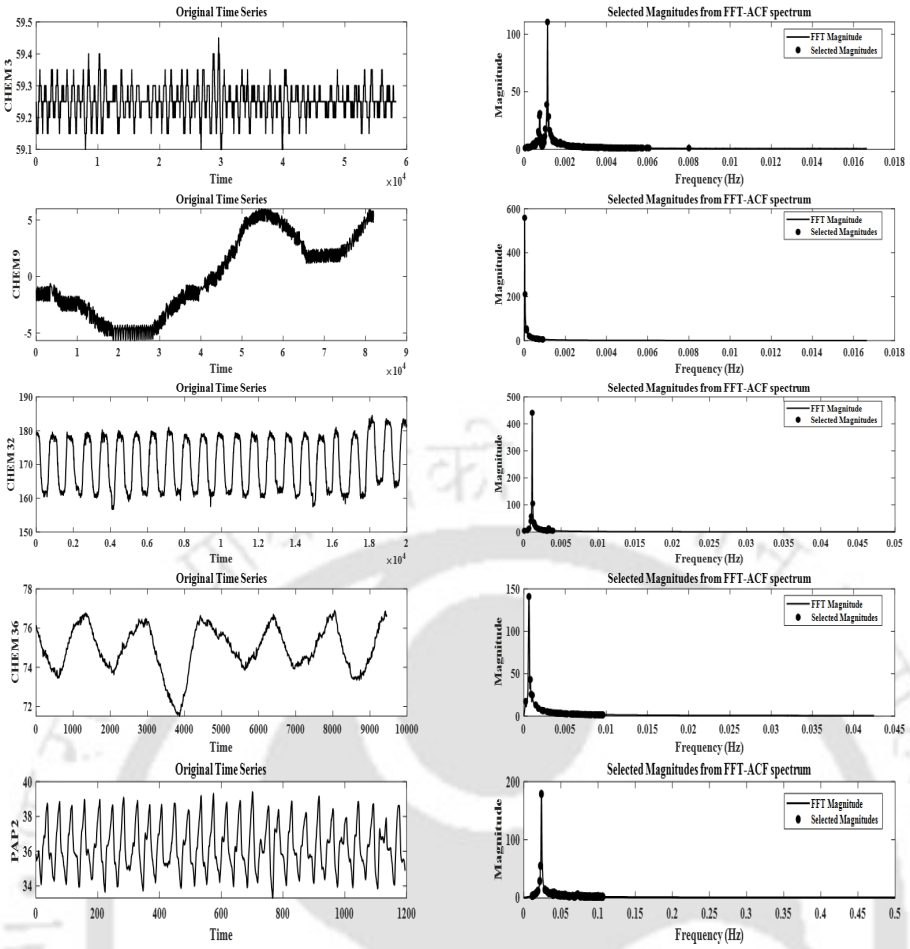


Figure 2.8: (a) PV Time Series Data for Different Industrial Loops; (b) FFT-ACF Plots for Period Evaluation.

FFT and the FFT-ACF spectra, applying a cycle count threshold (θ_c) of 2.5 cycles. A detailed analysis of the method's performance on these 93 control loops used exclusively for testing is provided in Appendix C.

Table 2.13: Accuracy Analysis of PV Time Series Data for Industrial Dataset.

No. of Features	Confusion Matrix	Class-wise Accuracy (%)			Overall Accuracy (%)	Computational time
		Class 0	Class 1	Class 2		
802 Features ($\theta_c = 2.5$)	$\begin{bmatrix} 12 & 0 & 5 \\ 1 & 21 & 1 \\ 4 & 2 & 47 \end{bmatrix}$	70.59	91.30	88.68	86.02	24 s

Table 2.13 presents the overall results of oscillation classification for the 93 PV time series dataset. The true data count classification is based on available comments in literature [120] and manual visual inspection for each PV time-series data. The

proposed approach gives an overall accuracy of 86.02%, demonstrating that the method is capable of accurately detecting and classifying oscillations across a wide variety of time series, even in complex industrial datasets, where oscillations of various types and magnitudes are common. The performance is higher for regular oscillatory data compared to non-oscillatory and irregular cases. The relatively high accuracy for all classes suggests that the trained neural network model, using 802 features derived from subsets of features of both FFT and FFT-ACF, is well-suited for handling the diverse oscillatory behavior present in industrial data. These findings support the method's potential for real-time oscillation detection in industrial applications. Nonetheless, when the model was evaluated under a binary classification setting (i.e., distinguishing Class 0 from either Class 1 or Class 2), the accuracy improved to 89.24%, demonstrating better robustness under simplified conditions. Additionally, the total time required for evaluating all 93 signals was approximately 24 seconds, resulting in an average per-variable inference time of around 0.26 seconds.

For additional analysis, the time series data with distinctive features shown in Figure 2.8(a) were examined. Based on the observations from Jelali and Huang [120] and visual inspection, it can be concluded that CHEM 3 experiences a quantization issue that causes oscillations. CHEM 9 is a non-oscillatory loop, while CHEM 32 demonstrates regular oscillations. Meanwhile, CHEM 36 and PAP 2 exhibit irregular and regular oscillations, respectively. Figure 2.8(b) presents the FFT-ACF spectrum with selected magnitude values to calculate Oscillation Ranking Index (ORI, ξ_i) for the estimation of period of oscillation. Table 2.14 presents the results of oscillation detection and quantification for these time series, where we compare the predicted period of oscillation (T_i), and amplitudes (A_f) with those obtained from manual visual inspection and Dambros et al. [97] for the selected PV data.

- **CHEM 3 (Irregular Oscillations)**: The proposed technique successfully predicted the class as irregular oscillatory (2). The period of oscillations

Table 2.14: Results for Oscillation Detection and Characterization Methods on Industrial Data.

PV Data	Class			Period of Oscillation (T_i)			Oscillation Amplitude (A_f)		
	True* (Visual)	Dambros	Predicted	True* (Visual)	Dambros	Predicted	True* (Visual)	Dambros	Predicted
CHEM 3	2	2	2	≈ 950**	998.68	[884.96, 840.34, 1315.8, 980.39]	≈ 0.08**	0.108	[0.075, 0.085, 0.074, 0.079]
CHEM 9	0	0	0	—	—	—	—	—	—
CHEM 32	1	1	1	≈ 850	658.53	900.9	≈ 10	10.159	10.127
CHEM 36	2	2	2	≈ 1570**	1925.47	[1639.3, 1282.1, 980.39, 1136.4]	≈ 1.8**	2.184	[1.668, 1.749, 1.575, 1.641]
PAP 2	1	1	1	≈ 43	33.28	42.23	≈ 2.4	2.821	2.330

**Values labeled as "True" are based on visual inspection. **Note that the period of oscillation (T_i) and oscillation amplitude (A_f) are not constant but vary over time for Class 2.*

were detected, which align closely with the results from visual inspection and Dambros et al. [97]. The amplitude estimations showed good agreement with the expected values.

- **CHEM 9 (Non-Oscillatory)**: This time series data was correctly classified as non-oscillatory (0) by the proposed method.

- **CHEM 32 (Regular Oscillations)**: The proposed technique successfully identified regular oscillations(1), providing estimates for period of oscillation and amplitude. These results were consistent with those from visual inspection, and the method demonstrated better accuracy in quantifying the period of oscillation when compared to the result from Dambros et al. [97]. The amplitude predictions showed good agreement with the expected value from visual inspection.

- **CHEM 36 (Irregular Oscillations)**: The proposed method successfully detected irregular oscillations. The estimated amplitude values closely matched the expected results from visual inspection, confirming the robustness of the technique. Visually, the period of oscillation detected aligns with the finding.

- **PAP 2 (Regular Oscillations)**: Regular oscillations in PAP 2 were correctly identified. The proposed method estimates the period of oscillation accurately, while the prediction from Dambros et al. [97] shows a large deviation. Moreover, the amplitude estimations closely matched the observed

value from visual inspection.

The results show that the proposed method is able to accurately identify oscillations and estimate their parameters, even when the oscillations present in industrial data vary in nature. The proposed method showed good performance with that of the comparison method by Dambros et al. [97], showcasing its suitability for real-world industrial applications where precise oscillation analysis is essential. Moreover, the proposed technique, with the trained model is capable of processing time series data within seconds, ensuring practicality for quick and seamless implementation.

2.4.5 Computational Efficiency Analysis

The computational efficiency of the proposed method was evaluated by comparing its execution time with existing approaches in the literature. Prior studies have reported detection speeds using Average Detection Time (ADT) per time-series signal, which represents the average time required to detect oscillations per time-series signal, including preprocessing steps. Additionally, the accuracy values of the existing methods were also collected from Wang et al. [101]. Table 2.15 summarizes both ADT and accuracy values for different detection methods.

For comparison, we used the industrial benchmark dataset introduced by Jelali et al. [120] , which contains 108 control loops. To ensure consistency with previous research, the same dataset of 63 control loops as used in Wang et al. [101] was selected for benchmarking. These prior methods were evaluated using an Intel Pentium G4600 (3.60 GHz CPU), 8 GB of RAM, and an NVIDIA GeForce GTX 1050 Ti (4 GB GPU).

In contrast, the proposed method was executed on two CPU-only systems without GPU acceleration:

- **System 1**: Dell Precision 3630 Tower (Intel Core i7-8700, 3.31 GHz, 32 GB RAM)

- **System 2**: Dell OptiPlex 5070 (Intel Core i5-8500, 3.00 GHz, 8 GB RAM)

Table 2.15: Comparison of Detection Speed and Accuracy of Each Method.

Method	**ADT (s/time-series signal)**	**Binary Classification Accuracy (%)**
ACF ratio [35]	0.0124	75.76
ACF zero-crossings regularity [36]	0.0057	78.33
Improved LMD [122]	2.2813	73.44
FACMD [123]	1.6206	81.25
EfficientNet-B0 [101]	0.3883	90.94 ± 3.01
ShuffleNet-V2 [101]	0.3503	92.50 ± 2.79
GhostNet [101]	0.3525	93.75 ± 2.47
MobileNet-V1 [101]	0.3577	94.37 ± 1.40
Proposed Method: 802 features, $\theta_c = 2.5$		
System 1	**0.1712**	**95.24**
System 2	**0.2442**	**95.24**

The results in Table 2.15 demonstrate the superior computational efficiency of the proposed method, with a total execution time of 10.79s on System 1 and 15.38s on System 2 for processing 63 control loops. The method achieved an ADT of 0.1712 s/time-series signal (System 1) and 0.2442 s/time-series signal (System 2), outperforming deep learning-based models such as EfficientNet-B0, ShuffleNet-V2, GhostNet, and MobileNet-V1 by 50–55%, despite running on CPU-only hardware without GPU acceleration.

In terms of accuracy, the proposed method achieved 92.06% overall accuracy in a multi-class setting. However, for a direct comparison with prior studies that classify only between oscillatory and non-oscillatory cases, the binary classification accuracy of the proposed method is 95.24%. This performance is comparable to deep learning-based techniques reported in Wang et al. [101]. The accuracy is higher than classical signal-processing-based methods such as ACF ratio (75.76%) and fast adaptive chirp mode decomposition (FACMD) (81.25%), while being close to the top-performing convolutional neural network (CNN)-based models, including GhostNet (93.75%) and MobileNet-V1 (94.37%).

In addition to demonstrating computational efficiency, the proposed method (802 features, $\theta_c = 2.5$) was compared against a baseline model [97], which normalizes the input time series and utilizes the complete set of FFT features (4097) as input. Our method achieved a slightly higher overall accuracy of 96.63%, compared to

96.10% for the baseline. Precision values for Classes 0, 1, and 2 were 0.958, 0.984, and 0.960, respectively, as opposed to 0.955, 0.979, and 0.948 for the baseline. Similarly, recall scores were 0.985, 0.960, and 0.958, compared to 0.983, 0.959, and 0.956 for the baseline model. These gains were achieved with 80% fewer features, significantly reducing model complexity and also reducing the risk of overfitting, leading to better generalization on unseen data. The reduced parameter space allows the model to adapt more effectively to new or evolving oscillation patterns, which is particularly valuable in industrial applications where signal characteristics may vary over time.

Finally, memory usage was also assessed to evaluate feasibility for real-time deployment on resource-constrained systems. On System 1, the memory usage increased from 362.79 MB to a peak of 558.94 MB during execution, while System 2 showed an increase from 338.95 MB to 472.44 MB. Thus, the total memory used was 196.15 MB and 133.49 MB on System 1 and System 2, respectively. These results indicate that the method operates within modest memory requirements, making it suitable for real-time industrial applications even on systems with limited RAM.

2.5 Sensitivity Analysis for Hyperparameters

In this section, we perform a sensitivity analysis to evaluate the impact of different hyperparameter configurations on the performance of a neural network classifier. The objective is to determine how variations in key parameters affect classification accuracy. The baseline model, as described in Section 2.3.1, consists of three hidden layers with 400, 100, and 20 neurons, respectively, using the Hard Sigmoid activation function, the Adam optimizer, a batch size of 10,000, and categorical cross-entropy loss. Due to the high computational cost and time constraints of an exhaustive analysis, we conducted a sensitivity analysis on a selected subset of hyperparameter variations. The parameters examined include the number of neurons in hidden layers, activation functions, batch size, optimization algorithms,

and the number of training epochs.

It is important to note that this sensitivity analysis was conducted independently of the methodology applied in our main study. The primary goal here is to analyze how different hyperparameter choices influence model performance, without incorporating additional methodological enhancements or domain-specific adaptations. To assess the sensitivity of the model, variations were introduced in these hyperparameters, as shown in Table 2.16, and the model was trained and evaluated under different configurations. A total of 576 model configurations were tested, with accuracy serving as the primary evaluation metric during training. A detailed analysis of the 576 model configurations is provided in Appendix D.

Table 2.16: Hyperparameter Variations and Preselected Baseline Configuration.

Hyperparameter	**Values Considered**	**Preselected Baseline**
Neurons per layer	[200, 50, 10], [400, 100, 20], [600, 200, 50]	[400, 100, 20]
Activation function	ReLU, Sigmoid, Tanh, Hard Sigmoid	Hard Sigmoid
Batch size	1000, 5000, 10000, 20000	10000
Optimizer	Adam, SGD, RMSprop, Adamax	Adam
Epochs	10, 30, 50	50

To evaluate the effectiveness of the preselected hyperparameters, the preselected configuration was evaluated alongside the top-performing configurations identified from the sensitivity analysis. Table 2.17 presents these configurations along with their accuracy, precision, recall, F1-score, and training time. The results indicate that while the preselected configuration is not the absolute best-performing setup, it remains a good and well-balanced choice. The sensitivity analysis demonstrated that different hyperparameter combinations could lead to slight improvements in accuracy (e.g., 0.9776 in the best case vs. 0.9756 for the preselected configuration). Additionally, a broader and more detailed sensitivity analysis covering a wider range of hyperparameter values could potentially uncover even better configurations.

Table 2.17: Comparison of Preselected Hyperparameters with Top Configurations.

Neurons	**Activation**	**Batch Size**	**Optimizer**	**Epochs**	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**Training Time (sec)**
[400, 100, 20]	ReLU	1000	RMSprop	50	0.9776	0.9777	0.9776	0.9776	404.78
[600, 200, 50]	Sigmoid	1000	Adamax	50	0.9772	0.9772	0.9772	0.9772	633.91
[600, 200, 50]	ReLU	10000	Adam	50	0.9759	0.9760	0.9759	0.9758	320.37
[400, 100, 20]	Hard Sigmoid	10000	Adam	50	0.9756	0.9757	0.9756	0.9756	390.02
[600, 200, 50]	ReLU	1000	SGD	30	0.9753	0.9753	0.9753	0.9753	259.61

2.6 Summary

This study introduces neural network-driven algorithms for accurate oscillation detection in process datasets. Methodologies for estimating period of oscillation and amplitudes are also proposed in this work. To improve accuracy and reduce computational demands, the approach integrates advanced feature engineering techniques grounded in domain expertise. Notably, features from the FFT and the FFT-ACF were utilized, with dominant peaks identified using a cycle count threshold.

The proposed technique was evaluated using synthetic and industrial datasets and showed strong performance across both. For the synthetic dataset, a cycle count threshold of 2 to 3 gave the highest accuracy of 96% when using a feature set of 802 features derived from both FFT and FFT-ACF spectra. On a benchmark industrial dataset, the method achieved an accuracy of 86.02%, demonstrating its ability to detect and classify oscillations in complex industrial datasets. Additionally, the trained neural network model performed the complete analysis in just a few seconds, making the method highly efficient for industrial applications. For estimating the period of oscillation, the method achieved a prediction accuracy of 99.3% for regular oscillations within a $\pm 5\%$ tolerance for the synthetic dataset and achieved 86.9% accuracy within a $\pm 15\%$ tolerance for irregular oscillations. The amplitude estimation results showed that the method consistently predicted amplitudes close to the actual values. Period of oscillation and amplitude estimations for the industrial dataset also showed accurate results.

Overall, the proposed method demonstrates high potential for accurate oscillation detection and parameter estimation in both synthetic and real-world industrial datasets. Future work could aim on improving the technique for irregular oscillations and exploring its scalability and adaptability to various industrial processes. In addition, the method could be extended for near real-time analysis using sliding window techniques, allowing continuous monitoring over several hours of control loop data. This would help address challenges such as concept drift and evolv-

ing process dynamics. The current approach uses a fixed cycle count threshold θ_c, but future efforts could explore adaptive threshold selection based on signal characteristics, such as spectral content, noise level, loop type, and available prior knowledge. Finally, integrating the proposed method with plant-wide oscillation diagnosis frameworks would enhance its practical deployment in industrial settings, enabling large-scale and automated monitoring of control loop performance.

Chapter 3

Analysis of Valve Stiction in Control Loops

3.1 Introduction

Modern process plants are dependent on automatic control systems and generally consist of hundreds of feedback control loops that maintain product quality, ensure energy efficiency, and operational safety. The performance of these loops often deteriorates over time due to issues such as poor controller tuning, sensor faults, multivariable interconnections, external disturbances, and actuator-related faults. Control valves are the most widely used actuators in sectors like petroleum refining, chemical manufacturing, and pulp and paper [30]. Based on the controller output, control valves regulate the manipulated variable but are susceptible to mechanical problems including hysteresis, deadband, backlash, saturation, and, most notably, stiction [24]. Stiction is caused by static friction and delays the movement of the valve stem relative to the controller signal, introducing nonlinearities and sustained oscillations in the process variable [25]. Various factors such as tight packing, particulate accumulation, lubricant degradation, or seal wear induce stiction, which is typically quantified as a percentage of the valve's operating range [7]. Stiction negatively impacts control loop performance, reduces product

quality, increases energy and material consumption, and accelerates equipment wear. Hence, in systems with frequent valve actuation, making early stiction detection is vital for maintaining process stability, reducing maintenance costs, extending valve lifespan, and enhancing operational safety [7, 30, 32].

Across different process industries, valve stiction has been identified as the cause of 20–30% of oscillatory behavior in control loops [6, 20, 22, 26, 27, 124]. A large-scale survey of more than 26,000 Proportional–Integral–Derivative (PID) loops showed that only 16% performed well, while 38% were rated fair or poor, with stiction identified as a key factor [28]. Detecting stiction in control valves is often difficult, particularly because valve stem position is rarely measured directly. Although stiction induces noticeable cycling patterns in the PV and OP signals, distinguishing it from other oscillatory disturbances requires robust diagnostic tools. In recent years, research has increasingly moved from rule-based heuristics to data-driven and machine learning-based approaches, following the broader Industry 4.0 trend. As sensor-rich automation becomes common in modern plants [29], the need for reliable and automated stiction detection methods has become more important, particularly in safety-critical operations where valve condition is directly tied to process stability.

In large industrial plants with hundreds of control valves, manual checking for stiction becomes impractical, especially for routine monitoring, highlighting the need for automated detection techniques. Several model-free approaches have been developed to assess valve condition. Notable among these are CURVE [47], AREA [48], CORR [49], HIST [50], and the shape-based techniques proposed by Kano and Yamashita [51–53]. While these methods generally perform well under constant set-point conditions, their accuracy deteriorates when the reference signal varies. For instance, Yamashita's method considered a consistent relationship between the PV and valve position, which is valid in flow loops but not in level loops. To address this, Brásio et al. [54] introduced a finite-difference transformation that accounts for reference changes but relies on valve position data, which

is often unavailable or is rarely measured in practice. To further tackle varying references, Dambros et al. [55] proposed the SLOPE and ZONE methods in which they compared the shapes of OP and PV signals with triangular and sinusoidal patterns. These shape-based methods work well under clean and high-resolution data but are sensitive to noise, sampling rate, and waveform shapes. Subsequent studies [56, 57] mitigate such issues by providing a sampling rate framework. Acknowledging the limitations of standalone methods, Pozo Garcia et al. [58] proposed detection framework that integrates multiple methods to achieve more robust performance across variable operating conditions. Additionally, Thornhill et al. [59] observed that valve stiction introduces odd harmonics in control signals. Building on this, Aftab et al. [60, 61] developed adaptive detection strategies using the HHT and MEMD, which can reliably distinguish nonlinear stiction-induced oscillations from linear disturbances even under noisy conditions and without prior process models.

Model-based approaches, particularly those employing Hammerstein models, have been widely explored for stiction detection. These models represent the system as a static nonlinear block (capturing stiction behavior) followed by a linear dynamic block (capturing process dynamics) [64]. Several variants, such as HAMM1–3 [7], were proposed, but issues like sensitivity to parameter initialization and ambiguity in integrating loops limited their effectiveness [65]. Improvements included robustness metrics [65], hybrid time-frequency analyses [66], and bootstrap-based parameter estimation [67]. Comparative studies by Bacci di Capaci et al. [68, 69] assessed linear models such as ARX, ARMAX, EARX, and EARMAX, and nonlinear models (Kano's, He's), noting that while simpler models suffice in clean conditions, extended models perform better in nonstationary environments. Extensions such as normality-test-driven model selection [125], nonlinear optimization [126], and Preisach-based structures [127] have enhanced generalizability. Building on this, Xiong and Zhu [128] proposed a Hammerstein model with a cubic spline nonlinear block and high-order ARX dynamics for closed-loop data. Aksornsri and

Wongsa [129] applied particle swarm optimization to estimate Kano model parameters within a Hammerstein-like framework. Additionally, Jeremiah et al. [70] introduced an integrated detection-compensation scheme using an ANFIS combined with dual-mode model predictive control. Most recently, El-Ferik et al. [71] developed a robust signal-processing and ML-based framework for conventional valves, validated under diverse industrial conditions. While these model-based strategies provide insight into stiction dynamics, challenges such as loop-specific tuning, excitation requirements, and limited scalability remain.

Classical statistical and regression-based approaches remain popular due to their simplicity and interpretability. Zheng et al. [72] proposed a K-means clustering-based moving window technique for stiction detection and quantification. Damarla et al. [73] introduced a linear regression method using OP–PV dynamics within sliding windows, while Liu and Wang [74] improved this by dynamically adjusting window size based on curve shape indices and signal features. Traditional statistical hypothesis testing, including F-tests, t-tests, and Hotelling's T^2-test, has also been used for anomaly detection [130], though these require manual threshold selection and often lack robustness across diverse datasets.

Unsupervised and supervised learning techniques have also been employed. Daneshwar and Mohd Noh [102] applied fuzzy c-means clustering to distinguish stiction-induced oscillations from external disturbances. Miskin et al. [103] used PCA to identify valve faults, including wear and stiction. Teh et al. [104] further developed a nonlinear PCA method combined with autocovariance analysis (NLPCA-AC), outperforming several existing techniques. Shang et al. [105] introduced a novel stiction index based on dynamic slow feature analysis and the Hurst exponent. Navada et al. [131] assessed the impact of data preprocessing and showed that the D-value method combined with an artificial neural network (ANN) achieved higher accuracy than PCA.

Artificial intelligence and deep learning methods have gained significant traction in recent years. Amiruddin et al. [106] used a multi-layer neural network trained on

transformed OP and PV data to classify stiction conditions. Venceslau et al. [96] applied a multi-layer perceptron (MLP) for estimating Choudhury's stiction parameters. Damarla and Huang [107] developed a Learning Vector Quantization Neural Network (LVQNN) trained on both simulated and industrial datasets for improved generalization. Several studies have used image-based signal transformation to leverage CNNs. Henry et al. [108, 109] transformed OP–PV trajectories into unthresholded recurrence plots (URPs), utilizing AlexNet for classification and severity estimation, later incorporating transfer learning. Memarian et al. [110] integrated Markov Transition Fields (MTF) with CNNs for robust detection. Dambros et al. [97] and Kamaruddin et al. [111] employed pixel-based encodings and butterfly-shaped CNN structures for oscillation and stiction detection. Tian et al. [132] introduced a neural network model based on deviation product features for accurate quantification.

SVMs have also been applied. Yazdi et al. [112] trained an SVM model on OP and PV signals to distinguish stiction from other oscillatory behaviors. Guan et al. [113] proposed a method based on phase space reconstruction and recurrence-based feature analysis, showing strong performance on industrial data. Vazquez et al. [114] trained a CNN using valve position and controller output data, achieving detection probabilities between 68.3% and 96.8%. With the rise of multimodal artificial intelligence, VLMs have also been investigated. StictionGPT [115] leverages a large vision-language model with few-shot adaptation, using shape features and aligned textual prompts to detect stiction with improved generalizability and interpretability. Despite substantial progress in both AI/ML-driven techniques for stiction detection, many existing approaches still face challenges such as sensitivity to parameter tuning, dependence on handcrafted features, high data requirements, and limited generalizability across diverse operating conditions.

In this work, we present a model-free methodology for control valve stiction detection that leverages the dynamic interaction between the OP and PV. The raw signals are first preprocessed through detrending, mean removal, and normaliza-

tion to ensure consistency across diverse operating conditions. Multiple forward and backward time shifts are then applied to generate heatmaps for OP and PV, which highlight stagnation or slow-varying regions symptomatic of stiction. These heatmaps are then projected back onto the original signal timeline to form merged heat traces, where the heat values from the original and all shifted versions are summed. This produces a time-series representation showing how strongly each time point exhibits slow or stuck behavior. From these traces, only PV-dominant regions are retained using mode-based scaling and sticky phase filtering. The merged heat signals are then segmented into oscillation cycles, and a PV/OP heat ratio is computed for each cycle. The average ratio across cycles serves as a feature for a threshold-based classifier, and an optimal threshold is determined via Receiver Operating Characteristic (ROC) analysis on the training set to maximize classification performance. This threshold is then applied to unseen loops for classifying loops as sticky or non-sticky, providing a simple, interpretable, and computationally efficient solution for stiction detection.

3.2 Industrial Data Availability

The proposed method is a model-free approach for valve stiction detection. This work uses the Industrial Stiction Benchmark Dataset (ISDB) [7], a publicly available repository of real industrial time-series data. The ISDB consists of a total of 93 control loops drawn from multiple industrial sectors like metals (MET), mining (MIN), power generation (POW), pulp and paper (PAP), chemicals (CHEM), and building automation systems (BAS). The dataset covers a wide range of control objectives such as flow (FC), level (LC), temperature (TC), pressure (PC), analyzer (AC), and gauge (GC) control. Diverse operating conditions, such as some loops running under constant SP, while others are subject to SP changes arising from cascade or supervisory control strategies, make it a suitable benchmark for validating the proposed method. Most loops in the dataset are affected by nonlin-

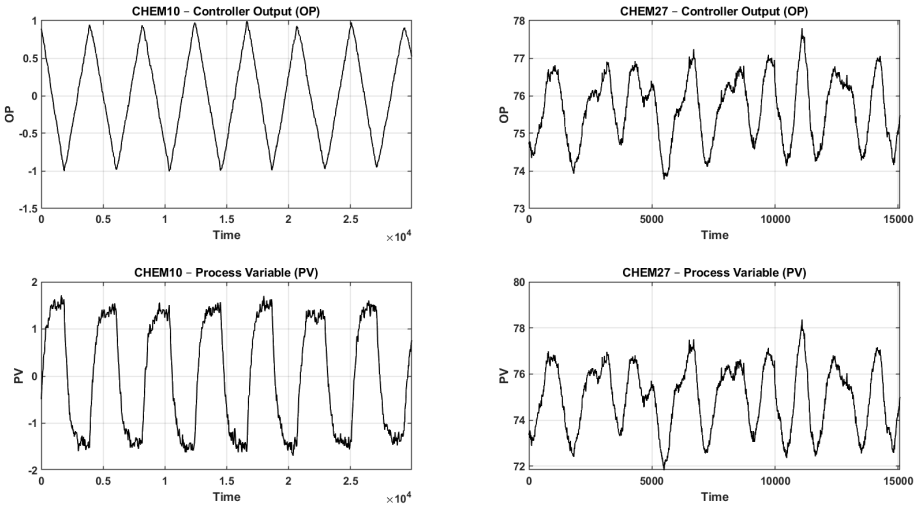
Table 3.1: Summary of the Industrial Stiction Benchmark Dataset (ISDB). Stiction presence is indicated as "1" (sticky) and "0" (non-sticky).

Industry Sector	Total Loops	Training	Testing	Stiction Presence (1/0)
Metals (MET)	3	3	0	0 / 3
Mining (MIN)	1	0	1	1 / 0
Power Generation (POW)	5	5	0	3 / 2
Pulp and Paper (PAP)	12	7	5	7 / 5
Chemicals (CHEM)	52	38	14	22 / 30
Buildings (BAS)	5	5	0	2 / 3
Total	**78**	**58**	**20**	**35 / 43**

earities such as valve stiction, controller saturation, and deadzones, with further complexities including noise, quantization effects, nonstationary trends, and open-loop operation in some cases [7]. In addition, the lengths of signals vary from a few hundred to over 270,000 time steps, showing differences in sampling period and process characteristics. These factors make the ISDB a highly relevant benchmark for developing and validating stiction detection methods. Unlike synthetic datasets, which are generated under controlled conditions, the ISDB represents real-world variability and offers a more rigorous test of algorithm robustness and industrial applicability.

Although the ISDB contains 93 loops, prior research has consistently focused on a benchmark subset of 78 loops [104, 106, 108, 113, 133, 134], and the same subset is adopted here to ensure comparability. Each loop in this subset provides synchronized measurements of the OP and the PV. In this study, 58 loops are used for training and model development, while the remaining 20 loops are reserved for out-of-sample testing. The choice of these 20 loops is consistent with several prior works [7, 25, 47–50, 55, 72, 111, 135], enabling meaningful performance comparison with existing detection methods. The distribution of loops across industrial sectors and their allocation to training and testing is summarized in Table 3.1.

As shown in Table 3.1, the benchmark subset contains 35 stiction-affected loops (denoted as "1") and 43 non-sticky loops (denoted as "0"), with most stiction cases are from chemical (22) and pulp and paper (7) processes. In contrast, no stiction is present in all three metal loops. This distribution reflects the presence of valve nonlinearities in certain industries. As seen, the dataset has slightly more non-



(a) (b)

Figure 3.1: Comparison of control loop behavior. (a): CHEM10 — stiction-affected loop. (b): CHEM27 — healthy loop.

stiction cases, but it still offers a sufficiently large number of stiction-affected loops for robust beachmarking. Also, the test dataset consists of both stiction and non-stiction loops across multiple industrial process plants to better validate the robustness of the proposed approach.

3.3 Methodology

3.3.1 Control Valve Stiction: Behavioral Basis for Classification

Control valves often suffer from stiction due to static friction that prevents movement of the valve stem. The OP and valve position (manipulated variable, MV) generally follow an approximately linear relationship in the absence of stiction. Due to the presence of stiction, the valve remains stationary or moves slowly until OP exceeds a certain threshold, and then slips abruptly, which introduces sustained oscillations. This stick–slip behavior negatively impacts the normal operation, and the induced oscillations may propagate to other interconnected variables in a process plant.

Figure 3.1 illustrates the behavior of a stiction-affected loop (CHEM 10) and a healthy loop (CHEM 27) taken from ISDB dataset. It can be observed that for CHEM 10, during the stick phase, the OP keeps varying while the PV remains nearly constant, indicating that the valve is stagnant or changing slowly. Once the OP exceeds the stiction threshold and overcomes the static friction, the valve shifts abruptly, producing a rapid PV response. In contrast, a healthy loop (CHEM 27) shows smooth co-variation between OP and PV, indicating the absence of stagnation or slow varying regions. These behavioral differences underpin the detection framework.

3.3.2 Proposed Detection Framework

Building on these behavioral characteristics, the proposed detection framework consists of the following steps.

Step 1: Signal Preprocessing and Normalization: For each control loop in the training and testing dataset, the time-series data is first preprocessed before analysis. Each loop consist of the PV, OP, and the corresponding time vector t. The objective of preprocessing is to eliminate offsets and slow linear trends, thereby isolating dynamic variations that are more indicative of stiction.
Let $x(t)$ denote a generic signal (e.g., PV or OP). The procedure consists of two steps:

- **Mean Removal.** The signal is centered by subtracting its temporal mean:

$$x_{\text{centered}}(t) = x(t) - \mu_x \tag{3.1}$$

  where μ_x is the average value of $x(t)$ over the entire time span.

- **Linear Trend Removal.** A best-fit straight line is estimated via least-squares regression and subtracted from $x_{\text{centered}}(t)$, giving the detrended signal $x_{\text{processed}}(t)$.

The detrended signal is then normalized to the range $[0, 1]$ using min–max scaling:

$$x_{\text{norm}}(t) = \frac{x_{\text{processed}}(t) - \min(x_{\text{processed}})}{\max(x_{\text{processed}}) - \min(x_{\text{processed}})} \tag{3.2}$$

The normalized controller output, $\text{OP}_{\text{norm}}(t)$, and process variable, $\text{PV}_{\text{norm}}(t)$, are subsequently used in the heatmap-based analysis. In addition, the average sampling interval is computed as:

$$\Delta t = \frac{1}{N-1} \sum_{i=1}^{N-1} (t_{i+1} - t_i) \tag{3.3}$$

where N is the total number of time points.

Step 2: Time-Shifted Heatmap Construction: Following normalization, both the $\text{OP}_{\text{norm}}(t)$ and $\text{PV}_{\text{norm}}(t)$ undergo time-shifted analysis to construct two-dimensional histograms, referred to as heatmaps. These heatmaps are designed to emphasize regions of temporal stagnation, which are indicative of valve stiction. To enhance the visibility of such behavior, each normalized signal $x_{\text{norm}}(t)$ is replicated across multiple forward and backward time shifts. Let n_s denote the number of shifts in each direction, and δt_{shift} represent the shift interval. The augmented dataset is constructed as

$$\mathcal{D}_x = \{(t,\ x_{\text{norm}}(t))\} \cup \bigcup_{s=1}^{n_s} \{(t \pm s \cdot \delta t_{\text{shift}},\ x_{\text{norm}}(t))\} \tag{3.4}$$

In our implementation, the time-shift interval is defined as

$$\delta t_{\text{shift}} = \alpha_{\text{shift}} \cdot \Delta t, \quad \text{with} \quad n_s = 10,$$

where Δt denotes the sampling interval, and α_{shift} is a dimensionless scaling factor controlling the temporal spacing between consecutive shifts. Unless otherwise stated, $\alpha_{\text{shift}} = 1$ is used as the nominal setting, resulting in a total of $2n_s + 1 = 21$ signal versions (the original plus ten forward and ten backward shifts). Sensitivity

analyses were also performed for $\alpha_{\text{shift}} \in \{0.5, 1.0, 2.0\}$ to examine the impact of different shift intervals, as described in Section 3.4.3.

The combined time–signal pairs from all shifts form the augmented dataset $\mathcal{D}_x$, containing M rows (i.e., time–value pairs). To convert this dataset into a heatmap, we construct a two-dimensional histogram in which the horizontal axis represents time and the vertical axis represents the normalized signal values. Both axes of the histogram are uniformly partitioned into a specified number of bins, and each bin corresponds to a rectangular region in the time–value plane. The heatmap intensity is determined by counting how many augmented data points fall within each bin, thereby capturing the overall density distribution of the shifted samples. The number of histogram bins per axis, defined over the full ranges of time and signal values, is adaptively selected as

$$N_{\text{bins}} = \min\left(400, \max\left(2, \left\lfloor \sqrt{M} \right\rfloor\right)\right) \tag{3.5}$$

Here, M denotes the total number of samples in the augmented dataset $\mathcal{D}_x$, and N_{bins} denotes the number of bins per axis.

The number of bins is selected using a data-adaptive square-root scaling rule based on the total number of samples in the augmented dataset. This approach is consistent with classical histogram binning heuristics introduced by Karl Pearson [136] and discussed in recent studies [137], and provides a practical balance between resolution and statistical reliability. When the number of bins is too low, it produces overly large bin areas that obscure structural patterns. Also, choosing too many bins produces many nearly empty regions and reduces interpretability. The adopted formulation mitigates these effects by scaling the bin count with dataset size, thereby facilitating consistent and interpretable heatmap representations across different datasets.

The resulting heatmap $H_x(i, j)$ encodes the counts of augmented time–value pairs in each 2D bin, where $i = 1, 2 \ldots, N_{\text{bins}}$ indexes the bin location along the time axis, and $j = 1, 2 \ldots, N_{\text{bins}}$ indexes the bin location along the normalized signal-value

axis. Each entry $H_x(i, j)$ therefore, represents the bin count, i.e., the total number of data points that lie within that specific rectangular region of the time–value plane. To ensure comparability between PV and OP heatmaps across datasets, each heatmap is normalized by its mean bin count μ_{H_x}:

$$\tilde{H}_x(i, j) = \frac{H_x(i, j)}{\mu_{H_x}}, \quad \text{where} \quad \mu_{H_x} = \frac{1}{N_{\text{bins}}^2} \sum_{i,j} H_x(i, j) \tag{3.6}$$

The normalized heatmaps, $\tilde{H}_{\text{PV}}$ and $\tilde{H}_{\text{OP}}$, highlight regions in time and signal value that have high point density, corresponding to low-variation, slow-dynamic behavior in the PV and OP signals.

Step 3: Merging Heatmap Values into a Time-Series Trace: Once the heatmaps $\tilde{H}_{\text{PV}}$ and $\tilde{H}_{\text{OP}}$ are constructed, each heatmap is projected back onto the original signal timeline to quantify how strongly each time point reflects slow-varying (i.e., potentially stuck) behavior.

We define a merged heat trace $h_x(t)$ for each signal (PV and OP), which maps each original time index t_i to a scalar value derived from the corresponding heatmap. This value indicates how consistently the point $x_{\text{norm}}(t_i)$ resides in a high-density region across nearby time shifts.

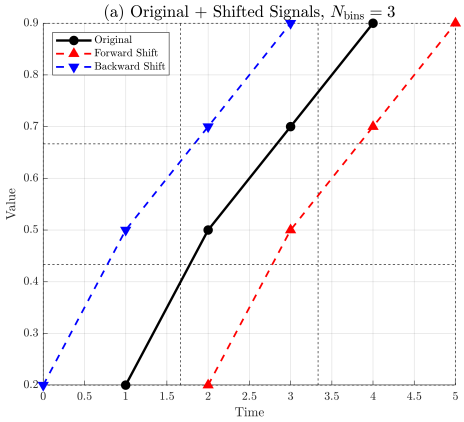
The heatmap $\tilde{H}_x(i, j)$ is defined over a 2D grid of bins, with the horizontal axis representing time and the vertical axis representing signal values. We define two bin-mapping functions:

- $b_t(\cdot)$: maps a timestamp to its corresponding bin index along the time axis,
- $b_x(\cdot)$: maps a normalized signal value to its corresponding bin index along the signal axis.

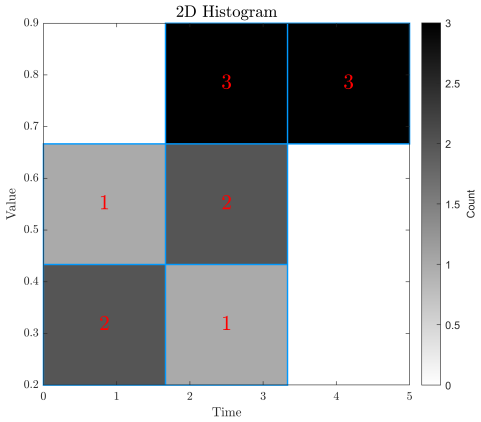
Using these mappings, the merged heat value at each time t_i is computed as:

$$h_x(t_i) = \sum_{s=-n_s}^{n_s} \tilde{H}_x\big(b_t\big(t_i + s \cdot \delta t_{\text{shift}}\big),\ b_x\big(x_{\text{norm}}(t_i)\big)\big) \tag{3.7}$$

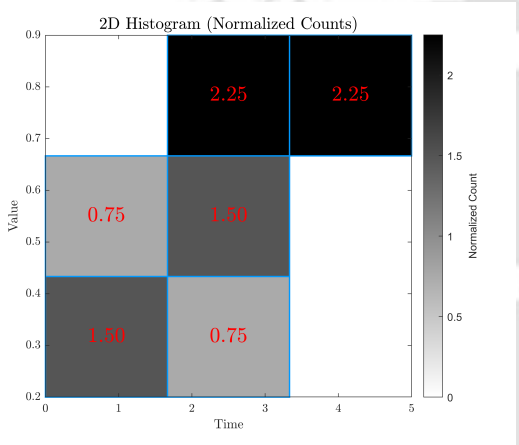
where:



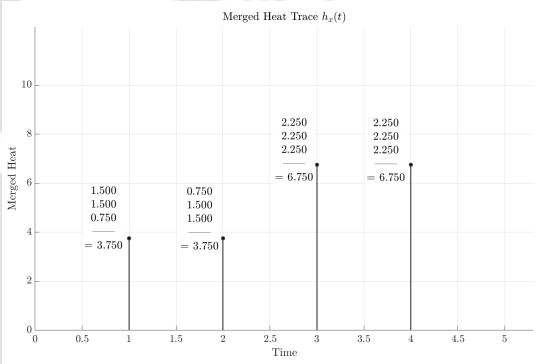
(a) Original + shifted signals.



(b) 2D histogram.



(c) 2D histogram (normalized, $\tilde{H}$), with mean bin count $\mu_{H_x} = 1.3333$.



(d) Merged heat trace.

Figure 3.2: Computation of the merged heat trace $h_x(t)$. (a) Original signal with forward and backward shifts by $\Delta t = 1$ s; dashed lines indicate bin edges. (b) 2D histogram (c) 2D histogram of normalized counts $\tilde{H}$. (d) Merged heat trace $h_x(t)$ obtained by summing the contributions from backward shift, original, and forward shift.

- $\delta t_{\text{shift}} = \alpha_{\text{shift}}\, \Delta t$ is the time–shift interval used in Step 2,
- $s \in [-n_s, n_s]$ spans the same range of shifts used in Step 2,
- Values that map to bin indices outside the heatmap boundaries are ignored (treated as zero).

This process yields two time-series traces, $h_{\text{PV}}(t)$ and $h_{\text{OP}}(t)$, each aligned with the original signal timeline. These traces quantify the temporal persistence of low-variation behavior and help isolate candidate regions of valve stiction.

Figure 3.2 illustrates Steps 2 and 3. For each time point shown in Figure 3.2a,

the original signal and its forward and backward shifts are mapped onto a 2D histogram (Figure 3.2c), where each bin represents a normalized count $\tilde{H}$. The merged heat trace $h_x(t)$ (Figure 3.2d) is obtained by summing the normalized heatmap contributions associated with the backward-shifted, original, and forward-shifted signal locations for each time instant. The annotations above each stem in the merged heat trace indicate the contributions from the backward-shifted, original, and forward-shifted signals, as well as their total sum.

Step 4: Mode-Based Scaling and Stiction Filtering: After generating the merged heat traces $h_{\mathrm{PV}}(t)$ and $h_{\mathrm{OP}}(t)$, we perform a filtering procedure to enhance segments where valve stiction is most prominent. This step includes identifying the most frequent value (mode), applying normalization, and masking out non-relevant regions.

For each trace, the most frequently occurring (mode) value is determined:

$$\hat{h}_{\mathrm{PV}} = \mathrm{mode}\left(h_{\mathrm{PV}}(t)\right), \quad \hat{h}_{\mathrm{OP}} = \mathrm{mode}\left(h_{\mathrm{OP}}(t)\right)$$

These mode values represent baseline heat levels across time and are used to normalize the traces.

Each heat trace is then scaled by its respective mode to highlight regions of unusually high intensity:

$$\tilde{h}_{\mathrm{PV}}(t) = \frac{h_{\mathrm{PV}}(t)}{\hat{h}_{\mathrm{PV}}}, \quad \tilde{h}_{\mathrm{OP}}(t) = \frac{h_{\mathrm{OP}}(t)}{\hat{h}_{\mathrm{OP}}} \tag{3.8}$$

To isolate behavior consistent with valve stiction, a masking operation is applied. At each time point t_i, if the normalized OP trace is greater than or equal to the normalized PV trace, both values are set to zero:

$$\tilde{h}_{\mathrm{PV}}(t_i) = \tilde{h}_{\mathrm{OP}}(t_i) = 0, \quad \text{if } \tilde{h}_{\mathrm{OP}}(t_i) \geq \tilde{h}_{\mathrm{PV}}(t_i).$$

This masking step preserves only those time points where the PV trace exhibits

more persistent (slow-varying) behavior than the OP trace, which is indicative of the "stick" phase in stick–slip dynamics. The filtered signals $\tilde{h}_{\mathrm{PV}}(t)$ and $\tilde{h}_{\mathrm{OP}}(t)$ are then carried forward for subsequent cycle-based ratio analysis.

Step 5: Cycle-Based Ratio Analysis: To quantify the relative dominance of persistent stagnation in the PV compared to the OP, a ratio analysis is performed. If the oscillation period T_c is unknown, the filtered heat traces $\tilde{h}_{\mathrm{PV}}(t)$ and $\tilde{h}_{\mathrm{OP}}(t)$ from Step 4 can be directly used to compute a single overall ratio, which corresponds to μ_R in the following formulation. In this study, the oscillation period T_c of the signal was available, allowing for a cycle-wise analysis to improve the robustness of stiction detection. As detailed in Chapter 2, the oscillation period was estimated using the proposed frequency-domain approach, and this information is now utilized in the present chapter for further analysis. The filtered signals $\tilde{h}_{\mathrm{PV}}(t)$ and $\tilde{h}_{\mathrm{OP}}(t)$ are divided into periodic segments, each corresponding to one oscillation cycle, and the ratio of $\tilde{h}_{\mathrm{PV}}$ to $\tilde{h}_{\mathrm{OP}}$ is computed within each cycle.
The time vector t is partitioned into non-overlapping segments of length T_c, representing the estimated oscillation period of the signal. In this study, T_c was obtained from previously reported values [138] and verified through visual inspection of the signal cycles. Let t_0 and t_N denote the start and end times of the signal, respectively. The i^{th} cycle is defined as

$$\mathrm{Cycle}_i = [\, t_0 + (i-1)T_c,\ t_0 + iT_c \,), \quad i = 1, 2, \ldots, n, \tag{3.9}$$

where

$$n = \left\lfloor \frac{t_N - t_0}{T_c} \right\rfloor$$

Within each cycle, the total merged heat contributions from the filtered heat traces $\tilde{h}_{\mathrm{PV}}(t)$ and $\tilde{h}_{\mathrm{OP}}(t)$ are computed:

$$\mathrm{PV}_i^{\mathrm{sum}} = \sum_{t_j \in \mathrm{Cycle}_i} \tilde{h}_{\mathrm{PV}}(t_j), \quad \mathrm{OP}_i^{\mathrm{sum}} = \sum_{t_j \in \mathrm{Cycle}_i} \tilde{h}_{\mathrm{OP}}(t_j) \tag{3.10}$$

The cycle-wise ratio is then defined as:

$$\text{Ratio}_i = \begin{cases} \dfrac{\text{PV}_i^{\text{sum}}}{\text{OP}_i^{\text{sum}}}, & \text{if } \text{OP}_i^{\text{sum}} > 0, \\ \text{NaN}, & \text{otherwise.} \end{cases}$$

Let

$$R = \{\text{Ratio}_i \mid \text{Ratio}_i \neq \text{NaN}\}$$

be the set of valid ratios. From this set, the following metrics are computed:

$$\begin{aligned} \mu_R &= \text{mean}(R), \\ \sigma_R &= \text{std}(R), \\ |R| &= \text{number of valid cycles}, \\ p_R &= \frac{|R|}{n} \times 100\% \quad \text{(percentage of valid cycles)}. \end{aligned}$$

These metrics provide a comprehensive characterization of stick–slip behavior. Here, μ_R and σ_R quantify the average and variability of the PV/OP heat ratios across valid cycles, providing insight into the typical dominance of persistent PV stagnation relative to OP and the consistency of such behavior. The number of valid cycles $|R|$ and the corresponding percentage p_R indicate the fraction of oscillation cycles that meaningfully contribute to the analysis. A high or low average ratio observed over only a small fraction of cycles (i.e., low p_R) may not provide sufficient evidence to reliably confirm or rule out the presence of stiction, whereas a consistently high ratio across a large proportion of cycles strongly indicates stick–slip dynamics.

Note: If the oscillation period T_c is unknown, the cycle-based computation described above cannot be applied. In that case, the ratio reduces to a single global

measure computed directly from the filtered heat traces:

$$\mu_R = \frac{\sum_t \tilde{h}_{\mathrm{PV}}(t)}{\sum_t \tilde{h}_{\mathrm{OP}}(t)} \tag{3.11}$$

Step 6: Threshold Selection and Loop Classification The loop-level mean ratio μ_R is computed for each loop, and a loop is classified as *Sticky* if its mean ratio exceeds a threshold θ^*. This step is common to both approaches: whether a single overall ratio is computed directly from the filtered heat traces (Step 4) or cycle-wise ratios are used (Step 5), the thresholding procedure remains the same. The threshold selection and evaluation are performed in two phases: a training phase to determine θ^* and assess discriminative ability, and a test phase to evaluate generalization.

Training phase. Let $\{\mu_R^{(k)}, y^{(k)}\}_{k=1}^{K_{\mathrm{train}}}$ denote the training set, where $\mu_R^{(k)}$ is the loop-level mean ratio for the k-th loop and $y^{(k)} \in \{0, 1\}$ is its binary label ($y = 1$ for Sticky, $y = 0$ for Non-Sticky).

To determine the optimal threshold, a ROC curve is constructed by evaluating the true positive rate (TPR) and false positive rate (FPR) across the range of observed ratios:

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}, \qquad \mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}}$$

where TP, TN, FP, FN are the entries of the confusion matrix defined as:

- **True Positives (TP):** sticky loops correctly classified as sticky,
- **True Negatives (TN):** non-sticky loops correctly classified as non-sticky,
- **False Positives (FP):** non-sticky loops incorrectly classified as sticky,
- **False Negatives (FN):** sticky loops incorrectly classified as non-sticky.

The area under the ROC curve (AUC) provides a threshold-independent measure of discriminative performance.

The threshold θ^* is then selected by maximizing the F_1 score, which balances precision and recall:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \qquad \text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}},$$

and the F_1 score is

$$F_1 = 2\,\frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3.12}$$

The optimal threshold is computed numerically by evaluating candidate thresholds in the range of observed ratios:

$$\theta^* = \underset{\theta_{\min} \le \theta \le \theta_{\max}}{\arg\max}\; F_1\Big(y,\ \mu_R \ge \theta\Big) \tag{3.13}$$

To assess robustness, a K-fold cross-validation ($K = 5$) is performed on the training set. For each fold, the ROC curve is computed on the held-out subset, and the mean and standard deviation of the cross-validated AUC provide an estimate of variability in discriminative performance.

Test phase. The threshold θ^* determined from the training set is applied to the independent test set $\{\mu_R^{(k)}, y^{(k)}\}_{k=1}^{K_{\text{test}}}$ to obtain binary predictions. The confusion matrix is constructed from these predictions, and performance metrics (accuracy, precision, recall, F_1 score) are reported. Bootstrap resampling (e.g., 1000 iterations) is used to compute 95% confidence intervals for accuracy.

Figure 3.3 shows the overall workflow of the proposed stiction detection methodology, comprising six steps: preprocessing, heatmap construction, merging into traces, mode-based filtering, cycle-based PV/OP heat ratio analysis, and threshold-based classification.

Note: In addition to the above training–test evaluation, further validation was conducted to assess the robustness of the proposed method and to mitigate potential bias arising from a fixed dataset split. Specifically, K-fold cross-validation (K =

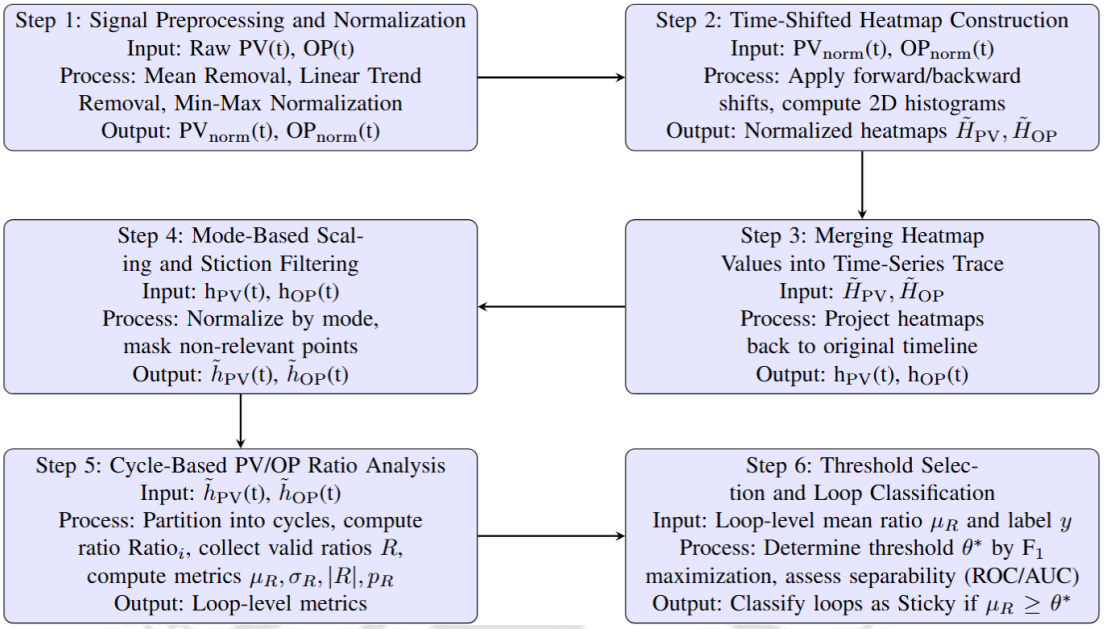


Figure 3.3: Heatmap-based workflow for valve stiction detection. Steps 1–6 show preprocessing, heatmap construction, merging, mode-based filtering, cycle-based ratio analysis, and threshold-based classification.

5) was performed on the entire dataset (78 loops), along with repeated stratified hold-out validation and nested cross-validation, as described in Section 3.5.

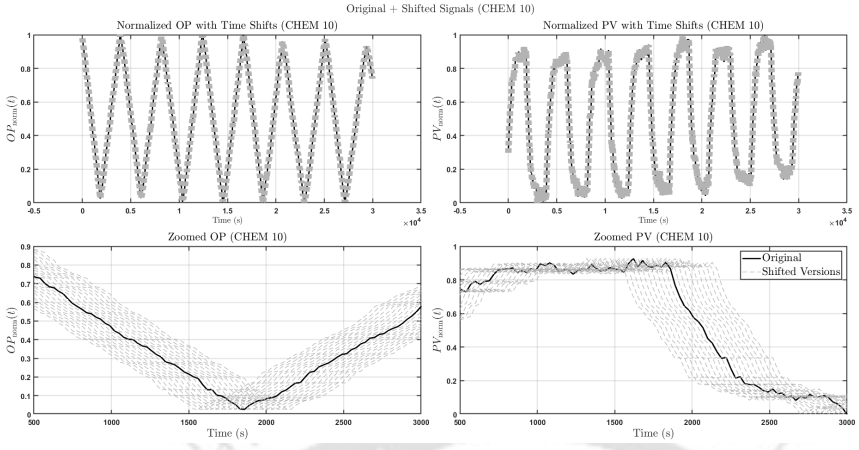
3.4 Results and Discussion

3.4.1 Illustrative Example

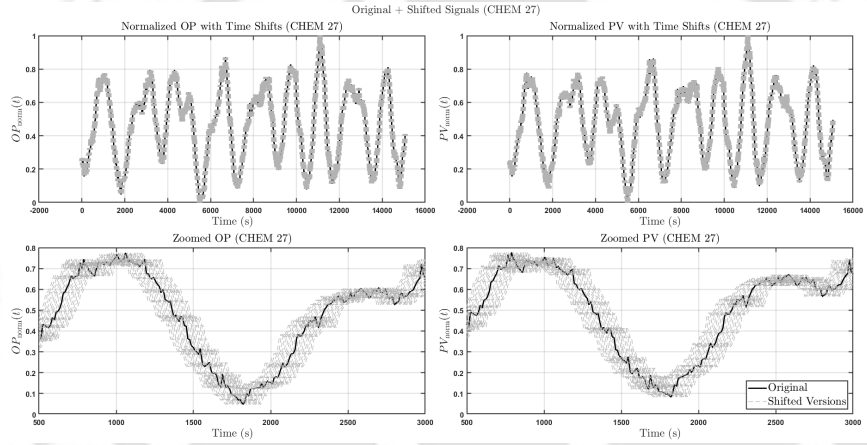
To illustrate the proposed methodology, two control loops, CHEM 10 and CHEM 27, were selected from the ISDB dataset [7]. Both the PV and OP signals were analyzed, with corresponding plots shown in Figure 3.1.

For each loop, the first step involves preprocessing the PV and OP time-series signals to remove offsets and slow trends. After detrending, signals were normalized to the range $[0, 1]$, allowing for direct comparison across loops. The average sampling interval Δt computed for CHEM 10 and CHEM 27 are 30 s and 11.3161 s, respectively. Following preprocessing, the normalized signals $OP_{norm}(t)$ and $PV_{norm}(t)$ were subjected to the time-shifting procedure described in Step 2 of Section 3.3.2 to construct heatmaps and highlight stagnant or slow varying regions in the signals. For each dataset, the signals were replicated across $n_s = 10$

forward and backward shifts, with the shift interval defined as $\delta t_{\text{shift}} = \alpha_{\text{shift}} \cdot \Delta t$, yielding a total of $2n_s + 1 = 21$ signal versions, including the original.



(a) CHEM 10



(b) CHEM 27

Figure 3.4: Time-shifted normalized signals: (a) CHEM 10 and (b) CHEM 27.

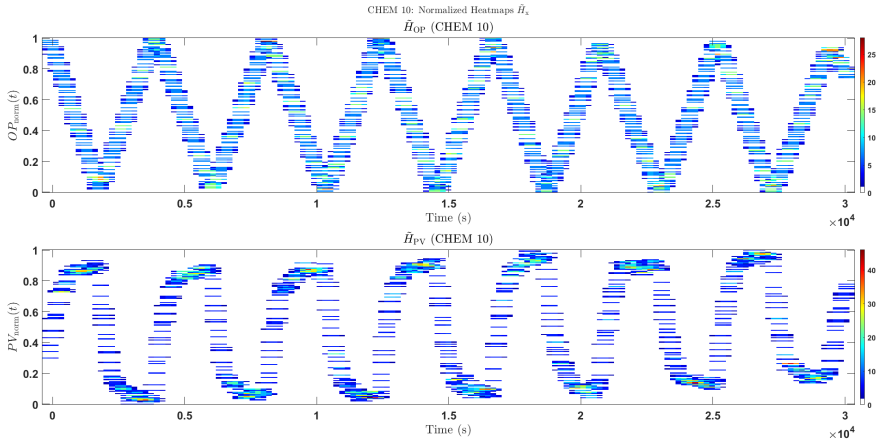
The resulting shifted signals for CHEM 10 and CHEM 27 are shown in Fig. 3.4a and Fig. 3.4b, respectively. For CHEM 10, the OP signal shows a nearly triangular waveform, while the corresponding PV response appears lagged and distorted. In the zoomed view, the PV remains nearly flat despite variations in the OP signal, which is a characteristic signature of valve stiction. The clustering of the shifted trajectories around the original PV signal highlights extended stagnation intervals. In contrast, CHEM 27 shows some clustering of the shifted trajectories in both OP and PV signals, which arises primarily due to measurement noise rather than extended stagnation. Unlike CHEM 10, the PV trajectory for CHEM 27 does not exhibit pronounced flat regions, suggesting the absence of a stiction effect.

These qualitative observations provide an initial indication of differences in valve behavior across the two datasets, which are further examined through the heatmap analysis.

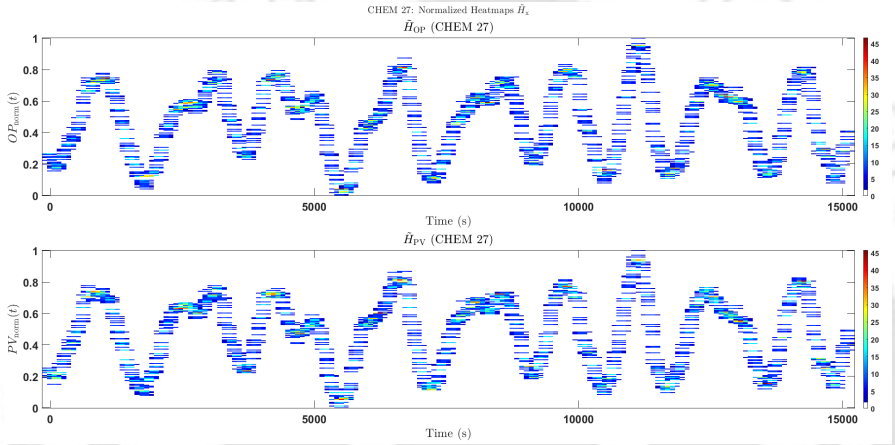
The augmented time–signal pairs formed the datasets $\mathcal{D}_{\mathrm{OP}}$ and $\mathcal{D}_{\mathrm{PV}}$, which were used to construct the normalized heatmaps $\tilde{H}_{\mathrm{OP}}$ and $\tilde{H}_{\mathrm{PV}}$. For CHEM 10, the resulting number of histogram bins along each axis was $N_{\mathrm{bins}} = 145$, whereas for CHEM 27, $N_{\mathrm{bins}} = 167$. Within each loop, the PV and OP heatmaps used the same N_{bins}, but the number differs across loops due to the different lengths of the signals. The heatmaps shown in Figs. 3.5a and 3.5b highlight low-variation regions in the PV and OP signals. For CHEM 10, the OP signal shows relatively uniform heat across time with small clusters appearing near the peaks and troughs, whereas the PV signal shows extended stagnation at these same regions, indicating periods of minimal variation. In contrast, CHEM 27 displays similar heat patterns for both OP and PV signals, with clustering occurring at similar time intervals for both signals and no significant stagnation, suggesting more dynamic behavior and the absence of pronounced stiction effects.

Each normalized heatmap, $\tilde{H}_{\mathrm{PV}}$ and $\tilde{H}_{\mathrm{OP}}$, was subsequently projected back onto the original signal timeline to compute a merged heat trace, $h_x(t)$, for each signal, as described in Step 3 of Section 3.3.2. At each time point t_i, contributions from all nearby time-shifted bins were summed, yielding a scalar value that reflects the persistence of low-variation behavior. This procedure produced two merged heat traces, $h_{\mathrm{PV}}(t)$ and $h_{\mathrm{OP}}(t)$, aligned with the original time axis.

Following the computation of the merged heat traces, each trace was scaled by its most frequently occurring value (mode) to highlight regions of unusually high persistence. For CHEM 10, the mode value was $\hat{h}_{\mathrm{PV}} = \hat{h}_{\mathrm{OP}} = 135.16$, while for CHEM 27, the mode value was $\hat{h}_{\mathrm{PV}} = \hat{h}_{\mathrm{OP}} = 152.43$. A masking operation was then applied: at any time point where the normalized OP trace satisfied $\tilde{h}_{\mathrm{OP}}(t_i) \geq \tilde{h}_{\mathrm{PV}}(t_i)$, both values were set to zero. This procedure effectively preserves only the segments where the PV signal exhibits dominant stagnation, indicative of potential



(a) CHEM 10



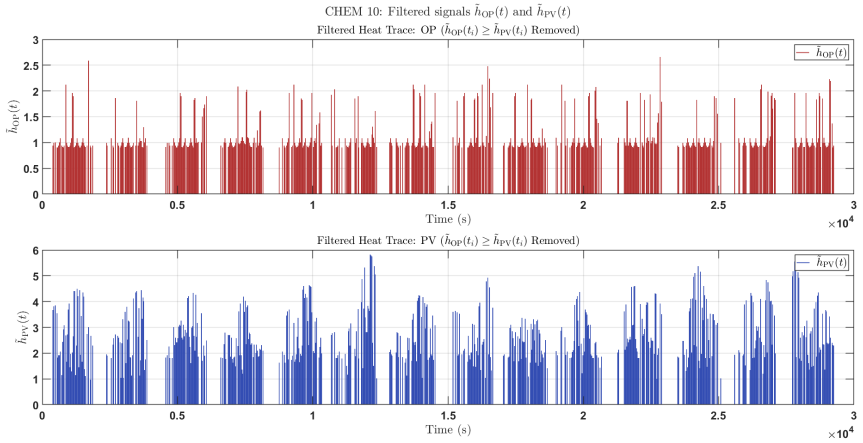
(b) CHEM 27

Figure 3.5: Normalized heatmaps $\tilde{H}_x$ highlighting low-variation regions in PV and OP signals for (a) CHEM 10 and (b) CHEM 27.

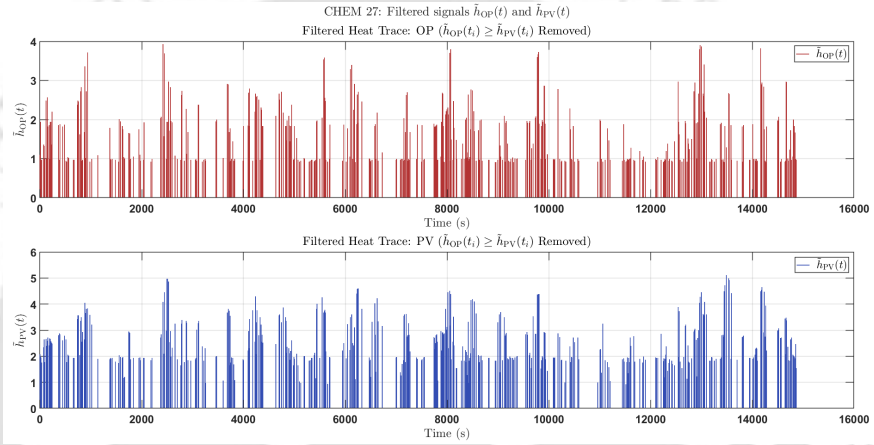
valve stiction.

The resulting filtered traces for CHEM 10 and CHEM 27 are shown in Fig. 3.6. In Fig. 3.6(a), corresponding to CHEM 10, the filtered trace $\tilde{h}_{\mathrm{PV}}$ exhibits distinct segments with high heat values, indicating extended stagnation intervals, while the filtered trace $\tilde{h}_{\mathrm{OP}}$ remains lower or nearly constant during these periods. In contrast, Fig. 3.6(b), corresponding to CHEM 27, shows $\tilde{h}_{\mathrm{PV}}$ and $\tilde{h}_{\mathrm{OP}}$ displaying intermittent, irregular spikes at various time points, reflecting dynamic behavior and the absence of significant valve stiction. These observations demonstrate that the mode-based scaling and masking procedure effectively isolates intervals of valve stiction and enables quantitative comparison across different control loops.

Following the generation of the filtered heat traces $\tilde{h}_{\mathrm{PV}}(t)$ and $\tilde{h}_{\mathrm{OP}}(t)$, a cycle-



(a) CHEM 10: Filtered heat traces $\tilde{h}_{\mathrm{OP}}(t)$ and $\tilde{h}_{\mathrm{PV}}(t)$, with regions where $\tilde{h}_{\mathrm{OP}}(t_i) \geq \tilde{h}_{\mathrm{PV}}(t_i)$ removed.



(b) CHEM 27: Filtered heat traces $\tilde{h}_{\mathrm{OP}}(t)$ and $\tilde{h}_{\mathrm{PV}}(t)$, with regions where $\tilde{h}_{\mathrm{OP}}(t_i) \geq \tilde{h}_{\mathrm{PV}}(t_i)$ removed.

Figure 3.6: Filtered heat traces for CHEM 10 and CHEM 27, showing only regions where PV exhibits dominant stagnation.

wise ratio analysis was conducted to quantify the relative dominance of persistent stagnation in PV compared to OP. The filtered signals were segmented into cycles of length T_c, corresponding to the dominant oscillation period of each loop. Values of T_c were adopted from previously published analyses of the same industrial dataset [138], which reported the characteristic oscillation period for all 78 loops. For instance, $T_c = 4166.67$ s for CHEM 10 and $T_c = 1666.67$ s for CHEM 27. Within each cycle i, the total contributions $\mathrm{PV}_i^{\mathrm{sum}}$ and $\mathrm{OP}_i^{\mathrm{sum}}$ were computed, and the cycle-wise ratio Ratio_i was obtained as described in Step 5 of Section 3.3.2. Only cycles with $\mathrm{OP}_i^{\mathrm{sum}} > 0$ were considered valid, forming the set R. From this set, the mean μ_R, standard deviation σ_R, number of valid cycles $|R|$, and valid

Table 3.2: Cycle-based PV/OP heat ratios for CHEM 10 and CHEM 27

Dataset	Mean Ratio μ_R	Std. Dev. σ_R	Valid Cycles $\lvert R\rvert$	Valid Cycle % p_R
CHEM 10	2.37	0.21	8	100%
CHEM 27	1.60	0.11	9	90%

cycle percentage p_R were computed to quantify the prevalence and consistency of slow-varying behavior in PV relative to OP.

The results for CHEM 10 and CHEM 27, summarized in Table 3.2, indicate the relative dominance of PV over OP across valid cycles. In the illustrative example, CHEM 10, corresponding to a *sticky* loop, shows a higher mean PV/OP heat ratio ($\mu_R = 2.37$), reflecting extended stagnation intervals in the PV signal. In contrast, CHEM 27, representing a *non-sticky* loop, exhibits a lower mean ratio ($\mu_R = 1.60$) and predominantly dynamic behavior with only short or irregular stagnation segments. These examples demonstrate that the cycle-wise PV/OP heat ratios effectively capture valve stiction and provide a quantitative feature for determining the optimal threshold for loop classification in the full dataset.

This illustrative analysis of CHEM 10 and CHEM 27 demonstrates the workflow of the proposed methodology, encompassing time-shifting and merging of heat traces, mode-based scaling and masking, and cycle-wise ratio analysis. While these examples allow qualitative and cycle-level quantitative evaluation, threshold-based classification (Step 6) is not applied here, as it requires the complete dataset to determine a robust threshold θ^* and assess performance statistically. The next section extends the methodology to the full dataset, applying Step 6 for comprehensive loop-level classification and benchmarking of the proposed stiction detection framework.

3.4.2 Performance Analysis on Industrial Benchmark Data

Building on the illustrative examples of CHEM 10 and CHEM 27, the analysis was extended to a comprehensive set of 78 industrial loop datasets [7]. For each loop, cycle-wise PV/OP heat ratio metrics were computed and summarized by the mean ratio (μ_R). Binary stiction labels (0: non-sticky, 1: sticky) were taken

Table 3.3: Confusion matrix for the training set (58 loops).

	Pred 0	Pred 1
True 0	29	7
True 1	6	16

Table 3.4: Confusion matrix for the test set (20 loops).

	Pred 0	Pred 1
True 0	5	2
True 1	0	13

directly from the literature [7]. From this pool, 58 loops were used for classifier training, while the remaining 20 were reserved for independent testing.

Training Phase The training set consists of 36 non-sticky loops (62.1%) and 22 sticky loops (37.9%). Mean ratios μ_R were computed for each training loop, and the optimal threshold for classifying a loop as sticky was determined via ROC analysis. The resulting threshold was $\theta^* = 1.692$, with AUC of 0.683, indicating moderate separability. Five-fold cross-validation gives a mean accuracy of 0.776 ± 0.096, suggesting robustness of the chosen feature across subsets of the training data. The resulting confusion matrix for the full training set is shown in Table 3.3. The classifier correctly identified 29 of 36 non-sticky loops (80.6%) and 16 of 22 sticky loops (72.7%). These results indicate that the obtained threshold $\theta^* = 1.692$ achieves a good balance between precision and recall, even though a few sticky loops were misclassified as non-sticky. Overall, the training results demonstrate that the proposed ratio feature provides meaningful discrimination between the two loop classes.

Test Phase The independent test set consisted of 20 loops, including 7 non-sticky loops (35%) and 13 sticky loops (65%). The loop-level mean ratio threshold θ^* determined from the training phase was applied to classify the test loops. The resulting confusion matrix is shown in Table 3.4.

From the confusion matrix, the classifier achieved perfect recall, correctly identifying all sticky loops, while also classifying 5 of 7 non-sticky loops correctly. The overall test performance metrics are summarized in Table 3.5.

Table 3.5: Performance metrics on the test set (20 loops).

Metric	Value	95% CI
Accuracy	0.900	[0.750, 1.000]
Precision	0.867	–
Recall	1.000	–
F_1-Score	0.929	–

These results indicate that the threshold derived from the training set generalizes well to unseen loops, providing reliable identification of sticky loops while maintaining acceptable classification of non-sticky loops. These results reflect classifier performance using the training-derived threshold $\theta^* = 1.692$ with time-shift parameters $n_s = 10$ and $\alpha_{\text{shift}} = 1$. The effect of varying these parameters on performance and computational efficiency is analyzed in the subsequent sensitivity study (Section 3.4.3).

3.4.3 Sensitivity Analysis of Heatmap Parameters and Computational Cost

(a) Sensitivity to bin selection N_{bins}:

To evaluate the robustness of the bin-selection strategy, the upper bound of N_{bins} was varied over a wide range (200–2500), and the resulting PV/OP heat ratio was computed across multiple datasets, as summarized in Table 3.6. In this analysis, the time-shift parameters were fixed at $n_s = 10$ and $\alpha_{\text{shift}} = 2$.

The results indicate that the computed feature values exhibit only minor variation despite substantial changes in the number of bins. For instance, in the CHEM4 dataset, the heat ratio remains unchanged across all tested configurations. For CHEM29 and BAS1, moderate variations are observed; however, no consistent trend is present as the number of bins increases. This suggests that the extracted heatmap-based features are not highly sensitive to the exact bin resolution.

To further examine the impact of binning strategy, an alternative approach based on Scott's rule [137, 139] was applied to the same augmented datasets, as shown in Table 3.7. This method produces significantly different and axis-dependent

Table 3.6: Sensitivity of PV/OP heat ratio to the upper bound of N_{bins} across datasets.

Dataset	Length (N)	M	$\sqrt{M}$	μ_R obtained by varying Bins (Upper limit)				
				200	400	500	1000	2500
CHEM 4	200	4,200	64.8	1.532	1.532	1.532	1.532	1.532
CHEM 29	7,201	151,221	389.0	2.06	1.93	1.93	1.93	1.93
BAS 1	277,115	5,819,415	2412	1.67	1.43	1.57	1.49	1.73

Table 3.7: Comparison of binning strategies: proposed method versus Scott's rule.

Dataset	Proposed Method (Bins)	μ_R	Scott's Bins (X-axis × Y-axis)	μ_R
CHEM 4	65 × 65	1.532	19 × 20–21	1.60
CHEM 29	389 × 389	1.93	53 × 83–112	1.84
BAS 1	400 × 400 (capped)	1.43	178 × 154–576	1.65

bin configurations compared to the proposed approach. However, despite these differences, the resulting heat ratios remain comparable across datasets, with only small deviations observed.

These findings indicate that the proposed method is robust to variations in both the number of bins and the binning strategy. Consequently, the square-root-based adaptive bin selection provides a stable and reliable representation without requiring fine-tuning across different datasets.

(b) Sensitivity to time-shift parameters n_s ***and*** α_{shift}:

To systematically assess the influence of time-shifted heatmap parameters on both classification performance and computational efficiency, a sensitivity analysis was conducted by varying the number of shifts n_s and the shift scaling factor α_{shift}.

Table 3.8: Sensitivity analysis of stiction detection with respect to number of shifts n_s, shift scaling factor α_{shift}, and computation time.

n_s	α_{shift}	θ^*	Training AUC	Training F_1	Test Accuracy	Computation Time (s)
5	0.5	1.794	0.742	0.692	0.850	29.273
	1.0	1.705	0.857	0.769	0.850	28.511
	2.0	1.692	0.746	0.692	0.900	29.248
10	0.5	1.694	0.793	0.750	0.800	53.012
	1.0	1.692	0.683	0.711	0.900	51.150
	2.0	1.740	0.684	0.696	0.950	52.584
15	0.5	1.656	0.701	0.667	0.750	75.644
	1.0	1.709	0.717	0.739	0.900	80.167
	2.0	1.748	0.681	0.651	0.950	75.391
20	0.5	1.663	0.705	0.735	0.850	102.718
	1.0	1.707	0.702	0.720	0.950	100.726
	2.0	1.772	0.684	0.682	0.900	101.671

The parameter combinations considered were:

$$n_s \in \{5, 10, 15, 20\}, \quad \alpha_{\text{shift}} \in \{0.5, 1, 2\}.$$

For each combination, the following procedure was employed. First, the $\text{PV}_{\text{norm}}(t)$ and $\text{OP}_{\text{norm}}(t)$ signals from the 58 training loops were augmented according to the selected n_s and α_{shift}, and the corresponding heatmaps $\tilde{H}_{\text{PV}}$ and $\tilde{H}_{\text{OP}}$ were generated. The threshold-based classifier was then trained on the mean ratio (μ_R) obtained from these heatmaps to determine the optimal threshold θ^* for stiction detection. Finally, the derived threshold was applied to the independent 20-loop test set to compute classification performance metrics, including F_1 score and accuracy.

This analysis provides a comprehensive view of how variations in n_s and α_{shift} affect both feature extraction and classifier performance. This allows identification of parameter ranges where the proposed method is robust and computationally efficient. Here, computation time is also reported and is the total runtime (in seconds) required to process all 78 industrial loop datasets, including the generation of heatmaps, computation of the heat ratio metrics μ_R, training the 58-loop dataset, and application of the threshold-based classifier to the 20-loop test set. This total runtime represents the end-to-end computational cost of the proposed method for the complete dataset.

Table 3.8 presents the sensitivity analysis showing the influence of n_s and α_{shift} on both classification performance and computational efficiency. For higher values of α_{shift} (e.g., $\alpha_{\text{shift}} = 1$ and 2), an increase in n_s generally improves the test accuracy, whereas beyond $n_s = 10$, no additional gain in accuracy is observed. However, for lower α_{shift} (e.g., $\alpha_{\text{shift}} = 0.5$), the accuracy does not improve with n_s, and in some cases the performance slightly decreases when n_s is increased up to 20. This suggests that longer time shifts improve separability between sticky and non-sticky loops. The highest accuracy of 95% for the lowest computational time is achieved for a combination of $n_s = 10$ and $\alpha_{\text{shift}} = 2$. This indicates that a moderate number

of shifts is sufficient to capture the relevant dynamic patterns for stiction detection with low computational cost. In addition, small variations in training AUC and F_1 scores across heatmap parameter settings indicate consistent performance on the training data, while test accuracy improves for certain configurations. This demonstrates effective generalization of the learned threshold. Computationally, total processing time increases with n_s, from approximately 29 s for $n_s = 5$ to 102 s for $n_s = 20$, reflecting the additional computation required for generating and processing a larger number of shifted signals. Overall, the proposed method generalize well for a wide range of n_s and α_{shift} values, with the configuration $n_s = 10$ and $\alpha_{\text{shift}} = 2$ providing an optimal balance between high accuracy and moderate computational cost, offering practical guidance for parameter selection in real-world industrial loop monitoring.

Table 3.9: Summary of all 78 datasets used in this study: training set (58 datasets) and testing set (20 datasets). The optimal classification threshold is $\theta^* = 1.740$. Cells highlighted in blue indicate datasets where $\mu_R \geq \theta^*$, while cells highlighted in red indicate cases where the predicted label does not match the true label.

Training Set (58 datasets)								
Loop	**μ_R**	**True / Predicted**	**Loop**	**μ_R**	**True / Predicted**	**Loop**	**μ_R**	**True / Predicted**
BAS 1	1.429	0/0	CHEM 28	2.467	1/1	CHEM 59	1.605	0/0
BAS 2	27.691	0/1	CHEM 30	1.74	1/1	CHEM 61	1.724	0/0
BAS 6	1.511	1/0	CHEM 33	1.617	0/0	CHEM 62	1.673	0/0
BAS 7	2.391	1/1	CHEM 34	2.791	0/1	MET 1	1.469	0/0
BAS 8	3.552	0/1	CHEM 35	1.511	1/0	MET 2	1.614	0/0
CHEM 4	1.532	0/0	CHEM 36	1.517	0/0	MET 3	1.557	0/0
CHEM 5	2.181	1/1	CHEM 37	1.587	0/0	POW 1	1.536	1/0
CHEM 7	5.029	1/1	CHEM 38	2.018	0/1	POW 2	1.522	1/0
CHEM 8	6.209	1/1	CHEM 39	1.641	0/0	POW 3	1.651	0/0
CHEM 9	2.586	1/1	CHEM 40	1.529	0/0	POW 4	1.592	0/0
CHEM 15	1.61	0/0	CHEM 44	2.682	0/1	POW 5	2.048	1/1
CHEM 17	1.533	0/0	CHEM 45	1.517	0/0	PAP 1	1.778	1/1
CHEM 18	1.792	1/1	CHEM 46	1.678	0/0	PAP 3	1.701	1/0
CHEM 19	2.344	1/1	CHEM 47	1.794	0/1	PAP 6	3.501	0/1
CHEM 20	1.8	1/1	CHEM 48	1.584	0/0	PAP 8	1.515	0/0
CHEM 21	1.775	0/1	CHEM 52	1.513	0/0	PAP 11	1.909	1/1
CHEM 22	5.096	1/1	CHEM 53	1.56	0/0	PAP 12	1.828	1/1
CHEM 25	1.998	1/1	CHEM 54	1.615	0/0	PAP 13	1.325	1/0
CHEM 26	1.634	0/0	CHEM 56	1.659	0/0			
CHEM 27	1.625	0/0	CHEM 58	1.739	0/0			
Testing Set (20 datasets, True/Predicted)								
Loop	**μ_R**	**True / Predicted**	**Loop**	**μ_R**	**True / Predicted**	**Loop**	**μ_R**	**True / Predicted**
CHEM 1	2.166	1/1	CHEM 13	1.542	0/0	PAP 2	1.950	1/1
CHEM 2	3.162	1/1	CHEM 14	1.668	0/0	PAP 4	1.496	0/0
CHEM 3	1.432	0/0	CHEM 16	2.171	0/1	PAP 5	2.203	1/1
CHEM 6	3.002	1/1	CHEM 23	1.995	1/1	PAP 7	1.45	0/0
CHEM 10	2.703	1/1	CHEM 24	3.209	1/1	PAP 9	1.632	0/0
CHEM 11	1.824	1/1	CHEM 29	1.930	1/1	MIN 1	2.129	1/1
CHEM 12	2.001	1/1	CHEM 32	2.227	1/1			

Table 3.9 presents the full set of 78 industrial loop datasets used in this study, including the 58 training loops and 20 testing loops. For each loop, the mean

PV/OP heat ratio (μ_R) is reported, along with the true stiction label (0: non-sticky, 1: sticky) and, for the test set, the predicted label obtained using the proposed method under the optimal configuration ($n_s = 10$, $\alpha_{\text{shift}} = 2$). For the training dataset, the method correctly classified 44 out of 58 loops, and a single misclassification was observed for CHEM 16 for the test dataset. Overall, the table provides a clear summary of the method's performance on both training and test datasets under the optimal heatmap parameters.

In addition, the mean heat ratio values (μ_R) for sticky and non-sticky loops are often numerically close near the decision boundary defined by the optimal threshold θ^* as seen in Table 3.9. This behavior is expected since μ_R is a normalized feature derived from dynamic interactions between the PV and OP, and even small variations in this ratio can correspond to meaningful differences in loop behavior. The threshold-based classifier distinguishes loops based on the statistical separability of their μ_R distributions across the training dataset instead of relying on large magnitude gaps between the two classes. In conclusion, although the μ_R values for the two classes may appear close, they capture distinct dynamic characteristics in the signals, and the ROC-optimized threshold θ^* effectively delineates these subtle but systematic differences.

3.4.4 Comparison with Literature

To assess the performance of the proposed method, it was compared with several established approaches reported in the literature. Table 3.10 summarizes the number of correctly classified loops out of the 20 widely used benchmark loops for each method, with results for competing methods taken directly from the respective original studies.

From the sensitivity analysis, the optimal configuration of $n_s = 10$, $\alpha_{\text{shift}} = 2$, with the ROC-optimized threshold $\theta^* = 1.740$ was obtained for the test dataset of 20 loops. The proposed method correctly classified 19 loops, achieving 95% accuracy as shown in Table 3.10, and outperforms the statistical test-based approach

Table 3.10: Comparison of stiction detection performance on the 20-loop benchmark dataset.

Method	Correct Diagnosis (out of 20)	Accuracy (%)
BSD [111]	17	85
KMW [72]	16	80
CNN-PCA [108]	16	80
BIC [140]	16	80
SDN [106]	15	75
HAMM2 [7]	14	70
HAMM3 [7]	15	75
CORR [50]	13	65
HIST [49]	13	65
RELAY [135]	13	65
ZONE [55]	13	65
CURVE [47]	12	60
SLOPE [55]	12	60
NLPCA-AC [104]	11	55
AREA [48]	10	50
Practical Linear Regression [73]	17	85
RPs-FI [113]	15	75
MD1 (Statistical Test) [130]	14	70
MD2 (Statistical Test) [130]	18	90
MD3 (Statistical Test) [130]	17	85
MD4 (Statistical Test) [130]	17	85
Proposed Method	**19**	**95**

(MD2), which showed the highest accuracy of 85% among the available literature techniques. Other advanced methods, such as BSD, CNN-PCA, and regression-based approach, achieved 16–17 correct detections (80–85%), while classical approaches such as AREA and NLPCA-AC generally gives accuracies below 60%. Overall, the proposed method, based on the PV/OP heat ratio and a straightforward thresholding procedure, demonstrates high accuracy while maintaining simplicity and interpretability, making it suitable for real-time industrial deployment, offering a reliable yet operationally clear solution for stiction detection.

To further compare performance, the overall classification of 58 training loops, including 20 test loops, based on the optimal threshold θ^* is presented in Table 3.11 along with previously reported methods. It should be noted that for the proposed method, this evaluation includes the loops used to determine the threshold; hence, the reported accuracy of 80.8% shows the overall performance of the method rather than a strictly independent validation. Nevertheless, on the full 78-loop dataset, the proposed method correctly detects 63 out of 78 loops, outperforming

Table 3.11: Comparison of stiction detection methods tested on the full 78-loop dataset.

Method	Correct Predictions	Accuracy (%)
HAMM3 [7]	45	57.7
GLCM-NN [134]	49	62.8
BIC [140]	50	64.1
LBP-NN [134]	53	67.9
NLPCA-AC [104]	54	69.2
CNN-PCA [108]	55	70.5
RPs-FI [113]	60	76.9
SDN [106]	61	78.2
Proposed Method	**63**	**80.8**

prior neural network-based approaches. This overall evaluation indicates that the method generalizes well across the complete set of industrial loops, providing a lightweight and practical approach suitable for real-world deployment.

3.5 Robustness and Generalization Validation

While the above results demonstrate strong performance on both the benchmark 20-loop test set and the full 78-loop dataset, additional validation was conducted to assess robustness and to mitigate potential bias arising from a fixed dataset split.

Specifically, K-fold cross-validation (K = 5), repeated stratified hold-out validation, and nested cross-validation were performed on the entire dataset. In all cases, the threshold θ^* was determined exclusively from the corresponding training subset using the F_1-score maximization criterion (Eq. 3.13), ensuring no information leakage from evaluation data. The optimal configuration of $n_s = 10$, $\alpha_{\text{shift}} = 2$ were kept fixed, as determined from the sensitivity analysis.

The results of these validation strategies yielded average accuracies of 78.1% ± 11.3% for K-fold cross-validation, 79.2% ± 7.1% for repeated validation, and 80.9% ± 10.6% for nested cross-validation. These results demonstrate consistent performance across different data partitions, indicating that the proposed method maintains robust generalization capability and is not overly dependent on a specific train–test split.

These results also indicate that the high accuracy observed on the fixed 20-loop test set is likely influenced by the specific dataset split, and the cross-validation results provide a more reliable estimate of expected real-world performance.

3.6 Summary

This study presents a simple data-driven and computationally efficient method for detecting stiction in industrial control loops. The proposed approach generates heatmaps from multiple time-shifted versions of the normalized time series signal. These generated heatmaps are used to highlight the slow-varying regions, and then cycle-wise PV/OP heat ratios are computed. The average of these heat ratios is then used as a feature for threshold-based classification. The optimal threshold was obtained using ROC analysis, and the proposed method was tested on a set of 20 industrial loops. For optimal heatmap parameters of $n_s = 10$ and $\alpha_{\text{shift}} = 2$, the method correctly classified 19 out of 20 test loops, achieving a high accuracy of 95%. Additional validation using cross-validation strategies yields consistent performance in the range of 78–81%, confirming the robustness and generalization capability of the proposed approach. Sensitivity analysis showed that moderate numbers of shifts are sufficient to reveal the slow dynamics and are computationally efficient. Overall, the proposed method generalizes well across a wide range of industrial loops and offers a good balance between accuracy, simplicity, and computational efficiency, making it suitable for near real-time monitoring and diagnosis of control loop stiction in industrial settings.

To further improve the proposed method, practitioners could apply denoising or signal preprocessing techniques in case of loops with high noise levels. Currently, the proposed approach uses only the mean PV/OP heat ratios from the heatmaps, whereas the heatmaps themselves could be used for advanced analytics, such as image-based or deep learning approaches, enabling real-time, continuous monitoring in evolving industrial environments, and enhancing their applicability in

large-scale process control systems.

Chapter 4

Root Cause Detection in Multivariable Control Networks

4.1 Introduction

Faults in control loops are a major industrial problem. Industrial plants are generally multivariate processes with many feedback control loops aimed at enhancing product quality and maximizing plant efficiency. Over time, plant performance deteriorates due to various factors such as changes in plant dynamics with time, poor controller tuning, stiction in control valves, etc., which would introduce oscillatory or non-oscillatory disturbances into the process. These disturbances (oscillatory/non-oscillatory) in one control-loop may propagate to other control loops and variables, thereby negatively impacting plant's overall performance [4, 31]. This may lead to sub-optimal plant operation, increased energy consumption, poor product quality etc. Hence, to maintain the plant's profitability, faults must be detected, diagnosed, and removed at the earliest. One of the main objectives of fault diagnosis in control loops is to identify and quantify the root causes of the detected faults.

RCA methods are generally classified into knowledge-based and data-driven approaches. Knowledge-based methods rely on qualitative reasoning to define causal

relationships between different units and subsystems in industrial processes. They are widely used for fault detection, fault propagation analysis, and root cause identification by leveraging process connectivity, expert knowledge, and algebraic models. Common techniques include SDG [75], adjacency matrices [76], and fault trees [77]. While these methods are effective in structured systems, their reliance on prior knowledge and static models limits their applicability to complex and dynamic industrial processes. Furthermore, their effectiveness diminishes in scenarios where process knowledge is incomplete or unavailable, making them less adaptable to modern industrial environments with evolving process conditions.

For large multivariate processes, RCA is a very time-consuming process. With advancements in industrial sensing and data acquisition, data-driven RCA methods have gained significant attention. Hence, many automatic root cause detection techniques have been researched over the past two decades[78–81]. Techniques such as CCF [62], Granger causality (GC) [82–84], and transfer entropy (TE) [86] infer causal relationships directly from process data. GC assesses predictive influence using multiple linear regression and is widely used due to its interpretability and simplicity, but it is primarily suited for linear relationships, limiting its effectiveness in complex, nonlinear industrial processes. The TE method [85, 86], which identifies suspicious variables by leveraging reconstruction-based contribution techniques but it requires large volumes of data. As data dimensionality increases, the computational complexity of estimating joint PDFs grows exponentially, making causal topology modeling less efficient. Furthermore, during process abnormalities, true causal relationships may become obscured or distorted, reducing the reliability of TE-based approaches. Among the reviewed methods, the Spectral envelope method [43, 63] has been commonly used for causal analysis in the frequency domain but is limited to only oscillatory faults.

Several Bayesian network-based methods [87–90] are also proposed for using probabilistic properties to determine the interdependencies between process variables. However, these data-based techniques depend on accurate estimation of PDFs and,

thus, would be computationally quite expensive. To address data scarcity and variability, Kumari et al. (2020) introduced a HBM that uses informative priors and key process variables, improving accuracy and efficiency [91], though constructing reliable priors remains a challenge. Later, Kumari et al. (2022) proposed a modified Bayesian Network (mBN) to handle cyclic loops by converting weak causal links into temporal ones [92], but its reliance on transfer entropy increases computational cost. Additionally, a Direct Transfer Entropy (DTE)-based multiblock BN was proposed to effectively discover cyclic loops while considering common source variables, offering improved accuracy and computational efficiency, though segmenting the process into blocks introduces complexity [93].

In addition, deep learning techniques have proven highly effective in capturing temporal dependencies within high-dimensional datasets, making them widely applicable across various domains [116–118]. Among these, RNNs are particularly well-suited for time-series modeling due to their ability to process sequential data. However, as the sequence length increases, traditional RNNs face challenges such as vanishing or exploding gradients, which hinder their ability to retain long-term dependencies. Despite these, most deep learning models remain focused on time-series prediction, while key tasks such as fault propagation analysis and root cause identification continue to be insufficiently addressed.

As seen, some of these methods require complete plant topology information (which may not be readily available) or some user-defined parameters (values of which may be unknown), and hence, difficult to implement. Moreover, most of the methods in the literature try to identify the root cause from a causal map of the process derived from either data or process knowledge. Identification of causal map is a very challenging problem, and the methods available so far cannot accurately identify the causal map. With the advent of data science, data-driven methods are gaining more popularity as they do not rely on plant information but extract the required information from the collected data of process variables. A simple, computationally effective, and purely data-driven approach that does not

rely on causal maps is missing in the literature.

This work proposes a simple, purely data-driven method for ranking likely root cause variables in a multivariate connected control loop system with faults. A metric based on cross-correlation weighted lag is used to provide a heuristic ranking of candidate fault sources rather than formal causal guarantees. To validate the proposed approach, various case studies are generated in MATLAB/Simulink. The synthetic data has been generated by introducing random noise/sinusoidal disturbance to specific variables. The algorithm is then tested on this data following the occurrence of a system fault.

4.2 Cross-correlation-based metric for ranking likely root-cause variables

The basis of the proposed approach lies in leveraging cross-correlation as a fundamental metric to assess the relationships between two-time series variables, namely x_i and x_j, within a multivariate process.The cross-correlation, denoted as, $\rho_{x_i x_j}[l] = E(x_i[k]x_j[k-l])$, quantifies the degree of similarity between these variables at a given time lag l. k indicates the sampling time.

The positive or negative values of cross-correlation at lag l indicate the nature of the relationship between $x_i[k]$ and $x_j[k-l]$. A positive value suggests that an increase in $x_i[k]$ is associated with an increase in $x_j[k-l]$ while a negative value implies an inverse relationship, where an increase in $x_i[k]$ corresponds to a decrease in $x_j[k-l]$. Moreover, the asymmetry of cross-correlation $\rho_{x_i x_j}[l] \neq \rho_{x_i x_j}[-l]$ is the key feature that makes it suitable for correctly identifying which variable lags behind the other. Although correlation does not imply causation, the asymmetry of cross-correlation can serve as a useful heuristic to rank candidate fault sources. In an interconnected process, disturbances originating in one variable propagate through the system and subsequently influence other variables after certain time delays. Therefore, variables closer to the origin of the disturbance are expected

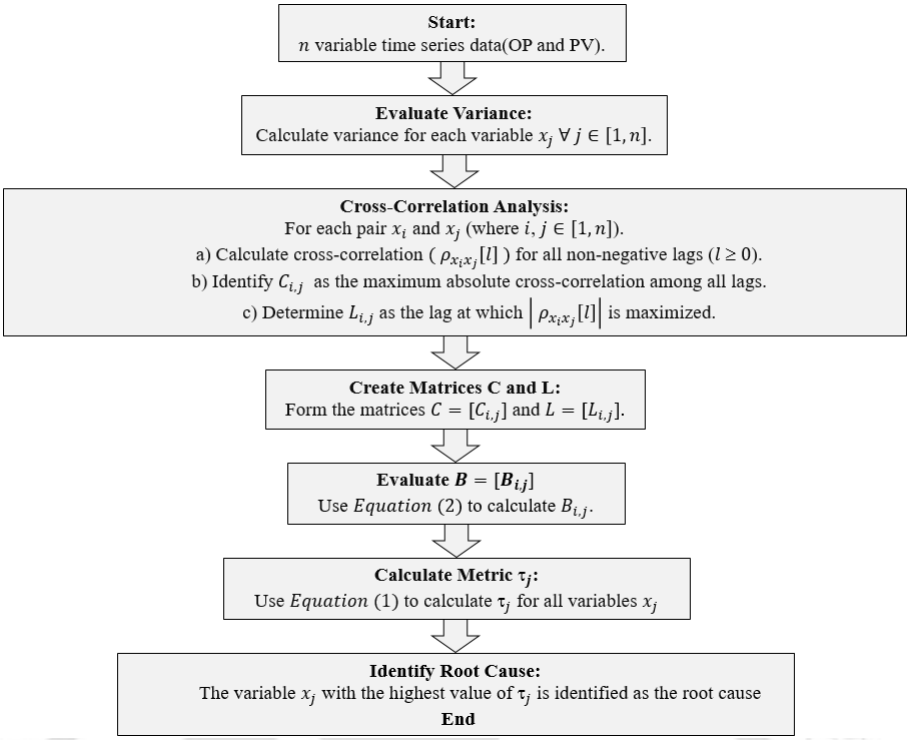


Figure 4.1: Sequential steps for ranking the likely root cause in a multivariate process

to exhibit stronger temporal precedence relative to downstream variables. The proposed approach exploits this propagation behavior by analyzing asymmetry in cross-correlation lags among process variables.

The faulty variable that precedes changes in other faulty variables and has a very high correlation to each of them is therefore considered more likely to be associated with the initiating disturbance. This is the principle behind the proposed approach.

Consider a dataset of n sensor measurements containing m samples ($\mathbf{X} = [x_1, x_2, ..., x_n]'$ where $x_i \in \mathbb{R}^m$). Let the maximum absolute cross-correlation (among non-negative lags) between any 2 measurements x_i and x_j be $C_{i,j}$ ($C_{i,j} = \max_l \quad |\rho_{x_i x_j}[l]| \quad \forall \quad l \geq 0$) and the corresponding lag be $L_{i,j}$. Here, $C_{i,j}$ represents the strength of dynamic association between variables x_i and x_j, while $L_{i,j}$ indicates the lag at which this association becomes maximum. Comparing $L_{i,j}$ and $L_{j,i}$ provides directional information regarding which variable temporally precedes the other.

Based on this notion, we propose a new metric as shown in (4.1) for ranking likely

root-cause variables.

$$\tau_j = \left(\sum_{i=1}^{n} C_{i,j} B_{i,j} \right) \sigma_j^2 \qquad \forall\ j \in [1, n] \tag{4.1}$$

where

$$B_{i,j} = \begin{cases} 1, & if\ L_{i,j} \leq L_{j,i} \ \ \& \ \ L_{i,j} \neq 0 \\ 0, & if\ L_{i,j} \leq L_{j,i} \ \ \& \ \ L_{i,j} = 0 \\ 1, & if\ L_{i,j} > L_{j,i} \ \ \& \ \ L_{j,i} = 0 \\ 0, & if\ L_{i,j} > L_{j,i} \ \ \& \ \ L_{j,i} \neq 0 \end{cases} \tag{4.2}$$

and σ_j^2 is the variance of variable x_j. The variance term σ_j^2 serves as a weighting factor reflecting the overall magnitude of fluctuations. Intuitively, a root cause variable is expected to both lead other variables temporally and exhibit pronounced variability during fault conditions. Although variance-based weighting could favor high-variance variables, we evaluated alternative schemes, including standard deviation (σ_j), normalized variance, entropy-based weighting, and SNR-based weighting, across multiple datasets, as presented in Section 4.4.1.5.

The binary indicator $B_{i,j}$ in (4.2) encodes the directional temporal relationship between variables x_i and x_j based on the estimated cross-correlation lags. Specifically, $B_{i,j} = 1$ indicates that variable x_j is considered to precede or influence x_i. The comparison between $L_{i,j}$ and $L_{j,i}$ is used to determine which variable exhibits an earlier correlation peak. Cases involving zero lag are treated separately since simultaneous changes do not provide sufficient temporal evidence for directional ranking.

If a variable has high τ, it means it is impacting many other variables in future. Thus, the highest ranked root cause variable is identified as the one with the maximum value of τ_j, i.e. (x_r) is the highest ranked root cause if $\tau_r = \max(\tau)$. This proposed metric is referred to as τ-metric hereafter.

$$\text{Root cause index} = \underset{j}{\text{argmax}}\ \tau_j \quad (\text{where } j = [1,\ n]) \tag{4.3}$$

Thus, the proposed approach involves evaluating variances, conducting cross-correlation analysis, and utilizing specific metrics such as $B_{i,j}$ and τ_j to pinpoint the root cause variable. In Fig. 4.1, each step is elucidated to provide a comprehensive understanding of the proposed method.

Interpretation of the τ-metric: The proposed τ-metric operates directly on measured signals and exploits cross-correlation asymmetry. The resulting τ-score provides a heuristic ranking of variables, with the highest-ranked variable representing the most likely candidate associated with the onset of abnormal behavior, rather than a formal causal guarantee. Accordingly, the term *root cause* in this work should be understood in a *diagnostic* and *operational* sense: the variable whose behavior is most strongly and consistently associated with the observed fault. The metric provides a ranked list of candidate variables, which can be used to support operator decision-making.

4.3 Synthetic data generation: Coupled transfer function models

The datasets used for testing the proposed τ-metric for ranking likely root-cause variables are generated using MATLAB Simulink. Multiple connected control loops are defined in Simulink in Laplace domain using the simplified control loop structure as shown in Fig. 4.2.

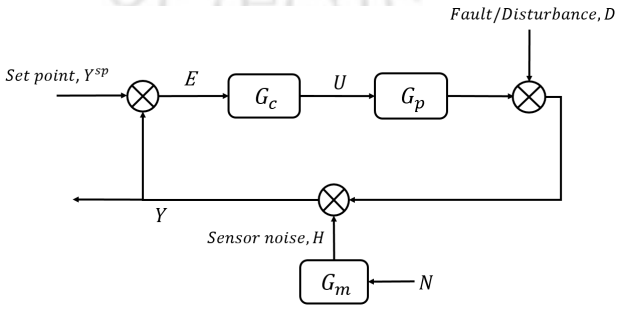


Figure 4.2: Closed-loop feedback control

The relations between various variables are given by:

$$Y(s) = G_p U(s) + H(s) + D(s); \tag{4.4}$$

$$U(s) = G_c E(s); \tag{4.5}$$

$$H(s) = G_m N(s); \tag{4.6}$$

$$E(s) = Y^{sp}(s) - Y(s) \tag{4.7}$$

where $Y(s)$, $U(s)$, $Y^{\text{sp}}(s)$, $D(s)$, $H(s)$ and $E(s)$ are the Laplace transforms of the deviation variables of outputs (y), inputs (u), setpoints (y^{sp}), disturbances (d), sensor noises (h) and errors (e) respectively. G_P and G_C are the process and controller transfer functions. Sensor noise is modeled as independent Gaussian noise $N(s)$ added to each measurement through the sensor model $H(s) = G_m N(s)$, where $G_m = I$. Different noise power values are used for different sensors as specified in the example systems. Faults are modeled as either oscillatory or Gaussian disturbances.

For data generation, we consider a stable plant model at steady state. Let us say, the process has n variables including both controller outputs (u, OP) and process variables (y, PV). For simplicity, we assume that there are no set point changes while simulating the synthetic datasets. For easy validation, datasets are generated such that fault is added only to one of the variables (by keeping only one of the values in D as non-zero for each dataset), i.e., there is only one root cause for each dataset simulated. We generated various datasets by perturbing one of the process variables (y) by adding either oscillatory or non-oscillatory fault/disturbance and simulated the process model to let the disturbance propagate through the process. Disturbance will propagate to other variables as per the interdependency defined through G_p. The variable that we perturbed would be the root cause while all other variables are affected through interactions between variables. The simulated data after the point of introduction of fault is used for root cause analysis.

4.3.1 Coupled transfer function models

We consider four coupled transfer function models with different numbers of variables and different characteristics as example systems to validate the performance of the algorithm in varying real-plant scenarios. The details of these four example systems are provided below. We refer to the dimension of the system as $n \times m$ where n is the number of controlled variables and m is the number of manipulated variables. Each control loop is considered to be SISO (Single Input Single Output).

Closed-loop behavior of these example systems is simulated in MATLAB Simulink using a fixed-step solver with a step size of 1 s. As a result, all simulations are inherently discrete-time, and no explicit continuous-to-discrete conversion is required. The solver performs numerical integration internally and updates the system outputs at each discrete time step.

- **Example 1:** 4×4 **system**[141]

  Process transfer function: Refer (4.11).

  Sensor noise model: Refer (4.12).

  Controller model: PI controllers with the following transfer function is used.

$$G_c = K_c \left[1 + \frac{1}{T_r} \left(\frac{\Delta T}{1 - q^{-1}} \right) \right] \tag{4.8}$$

  where K_c is the proportional gain, T_r is the integral time and $\Delta T = 1$ second. Values of K_c and T_r are as follows:

$$K_c = \begin{bmatrix} 0.816 & 0.625 & 0.184 & 0.137 \end{bmatrix} \tag{4.9}$$

$$T_r = \begin{bmatrix} 20 & 16 & 2.86 & 5 \end{bmatrix} \tag{4.10}$$

  Measurement noise $H(s)$ is introduced using Gaussian white noise $N(s)$ with noise power values $\eta = \begin{bmatrix} 0.0015 & 0.0025 & 0.0013 & 0.0027 \end{bmatrix}$, with each noise

signal generated using a different seed. The noise is applied through the sensor noise model G_m.

- **Example 2:** 6×6 **system**

  Process transfer function: Refer (4.13)

  Controller matrix: Refer (4.14)

  Gaussian white noise $N(s) = \mathcal{N}(0, \sigma^2)$ with
  $\eta = \sigma^2 = \begin{bmatrix} 0.01 & 0.01 & 0.02 & 0.02 & 0.01 & 0.01 \end{bmatrix}$ as noise power values is used as sensor noise $H(s)$ by setting $G_m = I$ (6×6 identity matrix). Each noise signal is generated with a different seed.

- **Example 3:** 10×10 **system**

  Process transfer function: Refer (4.15)

  Controller matrix: Here, G_c is a 10×10 diagonal matrix, where each diagonal element represents an individual transfer function. Refer (4.16)

  Measurement noise $H(s)$ is introduced using zero-mean Gaussian white noise $N(s)$ with noise power values

  $$\eta = \begin{bmatrix} 0.015 & 0.03 & 0.01 & 0.02 & 0.022 & 0.017 & 0.025 & 0.01 & 0.013 & 0.02 \end{bmatrix}$$

  with each noise signal generated using a different seed. $H(s) = N(s)$ is obtained by setting G_m to be a 10×10 identity matrix.

- **Example 4:** 50×50 **system**

  A large multivariable connected control loop system with 50 PV and 50 OP variables is considered. Due to space limitations, the transfer functions aren't displayed for this system and are presented in Appendix E.

$$G_P = \begin{bmatrix} \frac{0.05q^{-3}}{1-0.95q^{-1}} & 0 & 0 & 0 \\ \frac{0.02966q^{-3}}{1-1.627q^{-1}+0.706q^{-2}} & \frac{0.0627q^{-6}}{1-0.937q^{-1}} & 0 & 0 \\ 0 & \frac{0.235q^{-5}}{1-0.765q^{-1}} & \frac{0.5q^{-2}}{1-q^{-1}+0.25q^{-2}} & 0 \\ \frac{0.5q^{-5}-0.487q^{-6}}{1-1.395q^{-1}+0.455q^{-2}} & 0 & 0 & \frac{0.2q^{-1}}{1-0.8q^{-1}} \end{bmatrix} \tag{4.11}$$

$$G_m = \begin{bmatrix} \frac{1-0.1675q^{-1}}{1-0.8875q^{-1}} & 0 & 0 & 0 \\ 0 & \frac{1-0.2275q^{-1}}{1-0.7375q^{-1}} & 0 & 0 \\ 0 & 0 & \frac{1-0.1970q^{-1}}{1-0.8350q^{-1}} & 0 \\ 0 & 0 & 0 & \frac{1-0.1557q^{-1}}{1-0.9678q^{-1}} \end{bmatrix} \tag{4.12}$$

$$G_p = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ \frac{0.95}{2s+1} & 1 & 0 & 0 & 0 & 0 \\ 0 & \frac{0.84}{s+1} & 1 & 0 & 0 & 0 \\ 0 & 0 & \frac{0.77}{2s+1} & 1 & 0 & 0 \\ 0 & 0 & 0 & \frac{0.58}{3s+1} & 1 & 0 \\ 0 & 0 & 0 & 0 & \frac{0.44}{s+1} & 1 \end{bmatrix} \tag{4.13}$$

$$G_c = \begin{bmatrix} \frac{0.487s+0.202}{s} & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{0.426s+0.11}{s} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{0.601s+0.249}{s} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{0.907s+0.213}{s} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{0.904s+0.667}{s} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.14}$$

$$G_p = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{0.222}{6s+1} & 0 & 0 \\ 0 & 1 & \frac{0.7659}{4s+1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{0.2741e^{-4s}}{3s+1} & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{0.6288}{4s+1} & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{0.3276}{s+1} & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{0.02431}{2s+1} & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & \frac{0.001642e^{-7s}}{4s+1} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \frac{0.4698e^{-5s}}{6s+1} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{0.4849}{s+1} & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{0.6318e^{-7s}}{5s+1} & 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.15}$$

$$G_c = diag\left(\begin{bmatrix} \frac{5.101s+0.8026}{s} & \frac{0.094s+0.1881}{s} & \frac{1.894s+0.5249}{s} & \frac{0.6904s+0.134}{s} & \dots \\ \frac{1.263s+0.6763}{s} & \frac{15.74s+3.531}{s} & \frac{264.4s+51.29}{s} & \dots & \\ \frac{1.504s+0.2324}{s} & \frac{0.8204s+0.6138}{s} & \frac{0.8667s+0.1344}{s} & & \end{bmatrix} \right) \tag{4.16}$$

4.3.2 Assumptions of the Proposed Method

The proposed τ-metric is intended as a diagnostic ranking tool for identifying variables most strongly associated with a fault. Its applicability and interpretation rely on the following assumptions, which are consistent with the data generation and evaluation procedures used in this work. The assumptions listed below apply specifically to the synthetic benchmark simulations and are not necessarily satisfied by the industrial case studies.

a) **Stable closed-loop operation:** All synthetic processes and control loops are modeled using linear transfer functions and operate under stable closed-loop conditions.

b) **Steady-state operation:** Measurements are collected after the system reaches steady-state, with no setpoint changes, so that observed variations arise only from injected faults and their propagation.

c) **Local linearity and smoothness:** Although real industrial processes may be nonlinear, the synthetic benchmark systems are modeled using linear transfer functions and operate within a small neighborhood of a steady-state operating point. Over the analysis window, the system dynamics are therefore assumed to be locally linear and smooth.The observed performance of the proposed algorithm for the industrial case studies indicates that this assumption remains reasonable for slightly nonlinear systems.

d) **Sufficient excitation:** Injected faults introduce persistent excitation into the system, generating measurable dynamic responses and cross-correlations among variables. The method assumes that the fault signal contains sufficient energy and duration to be distinguishable from background noise.

e) **Fault scenario assumption:** The primary evaluation of the proposed method is conducted under a single-fault setting, where one dominant fault is active per dataset and modeled as an additive perturbation on an individual process variable. In addition, one experiment considers a two-fault scenario, and the observed results indicate that the methodology remains reasonable under limited multi-fault conditions. The extension of the proposed approach to scenarios involving more than two simultaneous faults is not investigated in this study and is considered beyond the scope of the present work.

f) **Sufficient disturbance propagation:** Fault-induced disturbances propagate through the process interconnections defined by G_p, producing measurable correlations between the faulty variable and affected variables.

g) **Additive measurement noise:** Sensor measurements are corrupted by zero-mean, independent Gaussian noise with finite variance.

h) **Discrete-time simulation:** All datasets are generated in discrete time using fixed-step numerical solvers with uniform sampling intervals.

i) **Loop structure:** In the synthetic benchmark systems, each control loop is single-input single-output (SISO).

4.4 Results: Validation of the proposed method

4.4.1 Coupled transfer function models-based case studies

This section presents the validation and performance evaluation of the proposed τ-metric for root cause analysis using multiple coupled transfer-function-based systems as presented in Section 4.3.1. First, an illustrative example is provided to demonstrate the computation of the proposed metric and its interpretation. Subsequently, extensive simulation studies are carried out on small-scale and large-scale systems to evaluate the accuracy of identifying the top-ranked variable associated with the fault. The robustness of the method is further analyzed under different weighting schemes, disturbance magnitudes, oscillation frequencies, and sampling times. Finally, the proposed approach is compared with the spectral envelope method for oscillatory fault diagnosis, discussed in Section 4.4.1.7.

4.4.1.1 Illustrative Example

To illustrate the proposed approach for heuristic ranking of candidate faulty variables, Example 1 with 8 variables is considered. The dataset generated is represented as X where the first four variables are process variables or PVs (x_1 to x_4) and the last four variables are controller outputs or OPs (x_5 to x_8).

A Gaussian noise disturbance with noise power 1 was added to x_2 as fault. The process is simulated for 5000 samples after introducing the fault. The complete simulated data (PV data) is shown in Fig. 4.3. The variance of the generated data (both PV and OP) is calculated and provided below:

$$\sigma_X^2 = 10^{-2} \times \begin{bmatrix} 0.42 & 101.55 & 5.85 & 2.59 & 0.43 & 44.78 & 1.24 & 1.33 \end{bmatrix}$$

The entire dataset (5000×8) was sent for root cause analysis using the method

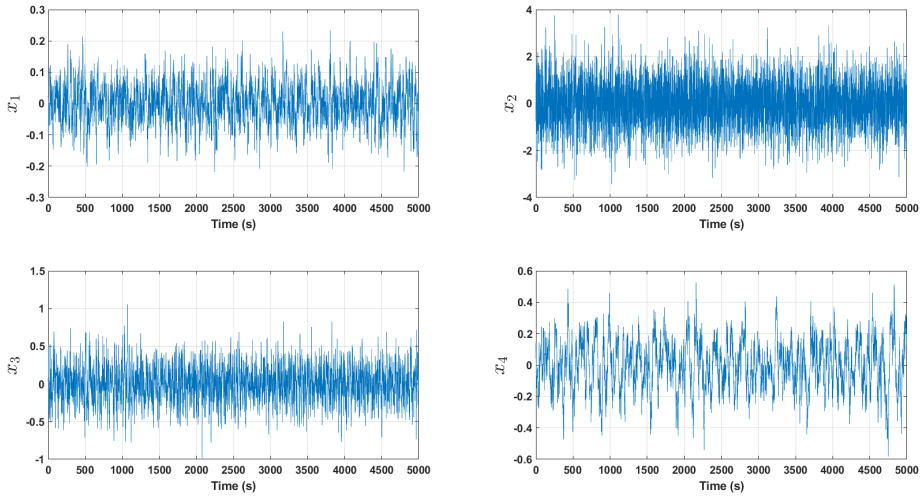
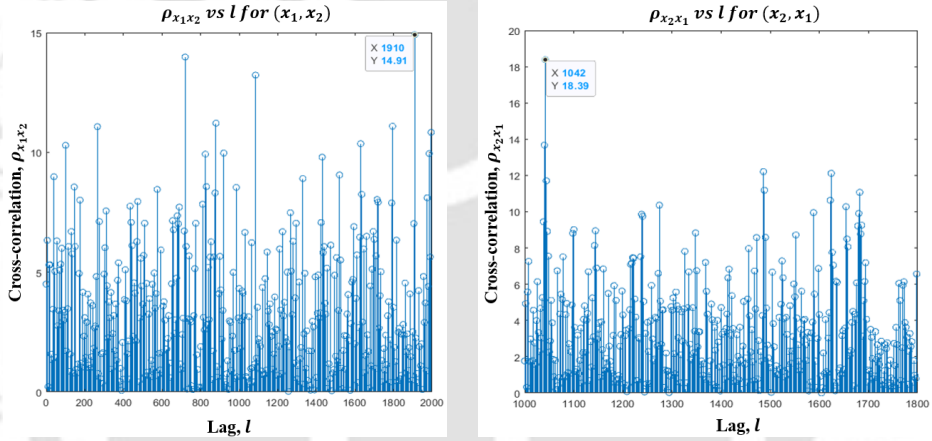


Figure 4.3: Simulated data for Example 1 with white noise fault added to x_2.



(a) (b)

Figure 4.4: Cross-correlation plot for the pairs (x_1, x_2) and (x_2, x_1).

described in section 4.2. For each pair x_i and x_j, cross-correlation (ρ_{x_i,x_j}) for all non-negative lags were found out. Maximum value of cross-correlation and the corresponding lags for each pair are noted down.

For example, Fig. 4.4a shows the plot of ρ_{x_1,x_2} for lags 0 to 2000. Note that the peak cross-correlation is at lag 1910, and the value of cross-correlation at lag 1910 is 14.9. Hence, $C_{1,2} = 14.9$ and $L_{1,2} = 1910$. Similarly, from Fig. 4.4b, we can see that $C_{2,1} = 18.4$ and $L_{2,1} = 1042$. The process is repeated to obtain maximum cross-correlation among all non-negative lags and the corresponding lag value for each pair. The values are presented below in matrix form, C and L.

$$C = \begin{bmatrix} 21.1 & 14.9 & 5.9 & 5.1 & 17.6 & 10.4 & 3.8 & 2.8 \\ 18.4 & 5076.6 & 51.6 & 31.7 & 15.9 & 3272.4 & 76.2 & 13.7 \\ 7.7 & 816.3 & 292.4 & 13.3 & 6.6 & 544.8 & 63.4 & 5.0 \\ 14.1 & 32.3 & 13.7 & 129.2 & 8.8 & 28.2 & 12.3 & 39.6 \\ 17.6 & 13.8 & 5.3 & 6.1 & 21.5 & 12.1 & 5.1 & 6.7 \\ 11.3 & 3272.4 & 49.7 & 28.2 & 11.7 & 2239.3 & 30.7 & 19.9 \\ 3.2 & 163 & 63.4 & 10.5 & 4.3 & 178.6 & 62.3 & 8.9 \\ 4.5 & 14.6 & 5.7 & 39.76 & 6.7 & 22.01 & 10.1 & 66.3 \end{bmatrix}$$

$$L = \begin{bmatrix} 0 & 1910 & 715 & 0 & 0 & 54 & 48 & 570 \\ 1042 & 0 & 360 & 338 & 1042 & 0 & 1 & 1069 \\ 1047 & 5 & 0 & 1043 & 1047 & 5 & 0 & 1074 \\ 5 & 1990 & 1985 & 0 & 5 & 388 & 377 & 33 \\ 0 & 42 & 95 & 2541 & 0 & 101 & 71 & 104 \\ 1042 & 0 & 0 & 1058 & 1042 & 0 & 0 & 370 \\ 671 & 6 & 0 & 1067 & 1634 & 6 & 0 & 1043 \\ 7 & 370 & 365 & 21 & 6 & 1387 & 1377 & 0 \end{bmatrix}$$

Now, using (4.2), B matrix is calculated as follows.

$$B = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \end{bmatrix}$$

$$\tau = [0.156 \quad 857.95 \quad 0.99 \quad 2.193 \quad 0.067 \quad 346.7 \quad 1.21 \quad 0.38] \tag{4.17}$$

It can be clearly seen that the actual fault was introduced in x_2, which serves as the ground-truth root cause in this synthetic example. According to the proposed τ-metric, x_2 also achieves the highest score, correctly ranking it as the most likely candidate associated with the fault.

4.4.1.2 Performance on small systems

For Example 1, four different datasets were generated similar to illustrative example by perturbing each of the PV variables at a time with white noise of power 10. Similarly, four different datasets were generated by perturbing each of the PV variables with sinusoidal disturbance with amplitude 0.5 and frequency 0.628 rad/sec. The τ values obtained for each dataset and the variable ranked highest by the τ-metric are provided in Table 4.1. Note that the τ-metric correctly ranked the perturbed variable as the highest-scoring candidate associated with the fault in seven out of the eight cases generated. For case 5, even though the root cause was x_1, the τ-metric predicted variable x_4 as the top candidate. However, note that x_1 has the second highest τ value and is the second possible candidate associated with the fault based on τ-metric heuristic.

Table 4.1: Results of the proposed τ-method for the example system 1

Cases		Actual root cause	τ-metric	Top ranked variable	First three predictions
White noise		1	$10^{-2}\begin{bmatrix}2562.06 & 2.73 & 0.15 & 91.97 & 1999.76 & 0.51 & 0.015 & 0.69\end{bmatrix}$	1	$\begin{bmatrix}1 & 5 & 4\end{bmatrix}$
		2	$\begin{bmatrix}0.33 & 83312.1 & 382.69 & 4.73 & 0.29 & 32963.5 & 99.02 & 1.28\end{bmatrix}$	2	$\begin{bmatrix}2 & 6 & 3\end{bmatrix}$
		3	$\begin{bmatrix}0.38 & 0.11 & 714.9 & 2.98 & 0.26 & 0.1 & 49.3 & 0.52\end{bmatrix}$	3	$\begin{bmatrix}3 & 7 & 4\end{bmatrix}$
		4	$\begin{bmatrix}0.32 & 0.1 & 0.12 & 1384.9 & 0.08 & 0.06 & 0.0057 & 19.66\end{bmatrix}$	4	$\begin{bmatrix}4 & 8 & 1\end{bmatrix}$
Sine		1	$\begin{bmatrix}15.26 & 0.34 & 0.06 & 85.53 & 9.7 & 0.21 & 0.0045 & 1.16\end{bmatrix}$	4	$\begin{bmatrix}4 & 1 & 5\end{bmatrix}$
		2	$\begin{bmatrix}0.14 & 31.87 & 2.3 & 1.2 & 0.14 & 0.28 & 0.04 & 0.08\end{bmatrix}$	2	$\begin{bmatrix}2 & 3 & 4\end{bmatrix}$
		3	$\begin{bmatrix}0.145 & 0.06 & 51.91 & 1.11 & 0.14 & 0.02 & 0.024 & 0.1\end{bmatrix}$	3	$\begin{bmatrix}3 & 4 & 1\end{bmatrix}$
		4	$\begin{bmatrix}0.14 & 0.07 & 0.018 & 19.22 & 0.14 & 0.03 & 0.002 & 0.093\end{bmatrix}$	4	$\begin{bmatrix}4 & 1 & 5\end{bmatrix}$

Varying noise power for non-oscillatory faults and sine wave parameters (amplitude and frequency) for oscillatory faults, a total of 52 case studies are generated for example system 1 (16 white noise faults and 36 sinusoidal faults). Similarly, various datasets are generated for other sample systems as well by perturbing different PV variables with either oscillatory or non-oscillatory faults. Example 2 is a 6×6 system and a total of 126 case studies are generated for this system (54 white

noise faults and 72 sinusoidal faults). For 10 variable system, a total of 230 case studies are generated (70 white noise faults and 160 sinusoidal faults).

A total of 140 datasets for white noise disturbance and 268 datasets for sine wave disturbances are generated for example systems 1 to 3 to test the performance of the proposed algorithm. Since the goal is to locate the faulty control loop among all loops, a prediction is considered correct if the τ-metric ranks either the perturbed PV variable or the corresponding OP variable as the top candidate associated with the fault. For instance, in an 8-variable system (4 PV and 4 OP variables), if the second PV variable is perturbed (x_2), the identification is deemed correct if the metric ranks either x_2 or x_6 (the corresponding OP variable) as the highest-scoring candidate.

Among the 140 cases generated for white noise disturbances, the τ-metric correctly ranked the perturbed variable as the top candidate in 131 cases, giving an accuracy of 93.57%. For the 268 cases with sinusoidal perturbations, the τ-metric correctly identified the top candidate as the perturbed variable in 236 cases, giving an accuracy of 88.06%.

4.4.1.3 Performance on a large system

In addition to the three example systems discussed in the previous section, the proposed τ-metric was also tested on a larger system. Synthetic datasets were simulated in Simulink similar to smaller example systems. A total of 21 datasets corresponding to white noise faults and 14 datasets for sinusoidal faults were generated. The results obtained using the τ-metric are provided in Table 4.2. Note that the metric correctly ranked the perturbed variable as the top candidate associated with the fault for all datasets.

Table 4.2: Top-ranked variable using τ metric for a large system with 50 PV and 50 OP variables

Cases	Actual root cause	Corresponding OP variable	Top ranked variable	Cases	Actual root cause	Corresponding OP variable	Top ranked variable
Noise Power=0.1	1	51	51	Sine Wave, Amplitude= 0.5, Frequency= 0.628rad/s	1	51	51
	8	58	58		8	58	58
	16	66	66		16	66	66
	25	75	75		25	75	75
	34	84	34		34	84	34
	42	92	42		42	92	42
	50	100	50		50	100	50
Noise Power=5	1	51	51	Sine Wave, Amplitude= 1, Frequency= 0.0628rad/s	1	51	51
	8	58	58		8	58	58
	16	66	66		16	66	66
	25	75	75		25	75	75
	34	84	34		34	84	84
	42	92	42		42	92	92
	50	100	50		50	100	100
Noise Power=10	1	51	51				
	8	58	58				
	16	66	66				
	25	75	75				
	34	84	34				
	42	92	42				
	50	100	50				

4.4.1.4 Overall Detection Performance for Coupled Transfer-Function-Based Systems

The overall accuracy of identifying the top-ranked variable associated with fault using τ-metric including all four example systems is 94.41% for white noise faults (152 cases out of 161) and 88.65% for sinusoidal faults (250 cases out of 282). Thus, the overall accuracy is 90.74% including both white noise and sinusoidal faults (402 cases out of 443). Notably, when the first two predictions are taken into account, the overall accuracy improves significantly to 94.81%, correctly identifying 420 out of 443 cases. It is important to note that the reported accuracy values are computed over a simulated dataset consisting exclusively of single-fault scenarios. The overall accuracy therefore reflects performance on this specific test set rather than a statistically generalizable measure across all possible fault conditions. The dataset is also imbalanced with respect to fault type (161 white-noise cases and 282 sinusoidal cases); for this reason, fault-type-specific accuracies are reported alongside the overall accuracy.

Table 4.3: Comparison of weighting schemes for τ-metric across all example systems

System	Dataset	σ_j^2 (Proposed)	σ_j	Normalized σ_j^2	Entropy	SNR
Example 1	52	43	43	43	40	5
Example 2	126	115	107	115	96	24
Example 3	230	209	193	209	159	42
Large system	35	35	34	35	33	0
Overall	443	402	377	402	328	71

4.4.1.5 Weighting schemes and binary Filter ($B_{i,j}$) analysis

The proposed τ-metric incorporates a variance-based weighting term to reflect the magnitude of fluctuations associated with each variable. While this choice is motivated by the expectation that a root cause variable exhibits both temporal precedence and pronounced variability under faulty conditions, variance-based weighting may potentially bias the metric toward high-variance signals. To examine this effect, we conducted a systematic comparison with several widely used alternative weighting strategies.

Specifically, for each variable x_j, a scalar weight W_j was computed using one of the following schemes: standard deviation ($W_j = \sigma_j$), normalized variance $\left(W_j = \sigma_j^2 / \sum_{i=1}^{n} \sigma_i^2\right)$, Shannon entropy-based weighting

$$W_j = -\sum p_j \log(p_j)$$

and signal-to-noise ratio (SNR) weighting $\left(W_j = \mu_j/\sigma_j\right)$, where μ_j and σ_j denote the mean and standard deviation of x_j, and p_j represents the empirical probability of values of x_j obtained from its observed distribution.

For each weighting scheme, the τ-metric was evaluated as

$$\tau_j = \left(\sum_{i=1}^{n} C_{i,j} B_{i,j}\right) W_j \qquad \forall\ j \in [1, n]$$

and the variable attaining the maximum τ_j was considered the top-ranked variable associated with fault. All weighting strategies were assessed under identical conditions to ensure meaningful comparison.

Table 4.3 reports the total number of correct top-ranked variable associated with fault obtained across all 443 datasets for each weighting strategy. The proposed variance-based weighting (σ_j^2) achieves the highest or joint-highest identification accuracy in every example system, including the large-scale system. The normalized variance produces identical overall performance, while the standard deviation weighting exhibits a modest reduction in accuracy. In contrast, both the Shannon entropy-based and SNR-based weighting schemes result in substantial performance degradation across all cases. These results indicate that variance-based weighting provides the most reliable and stable behavior for the proposed τ-metric.

In addition, to investigate whether explicitly using the magnitude of the lag values improves identification of top top-ranked variable associated with fault, actual lag values $L_{i,j}$ were directly incorporated whenever $B_{i,j} = 1$, instead of using a binary indicator. The total number of correct identifications across all 443 datasets for different weighting schemes is summarized in Table 4.4.

Table 4.4: Performance of the τ-metric using actual lag values for different weighting schemes

System	Dataset	σ_j^2	σ_j	Normalized σ_j^2	Entropy	SNR
Example 1	52	43	35	43	23	3
Example 2	126	107	102	107	78	25
Example 3	230	205	183	205	141	41
Large system	35	35	32	35	27	0
Overall	443	390	352	390	269	69

As seen in Table 4.4, incorporating the actual lag magnitudes slightly reduces performance for all weighting schemes compared to the proposed binary filter formulation of $B_{i,j}$ (see Table 4.3). This confirms that the binary filter effectively captures temporal precedence while maintaining stable and reliable identification of the top-ranked variable associated with the fault across different weighting schemes.

Table 4.5: Detection Performance Across Noise Levels and Sinusoidal Disturbances

Condition	Dataset	Correct Detections	Incorrect Detections
Noise Power			
0.1	20	12	8
0.5	16	15	1
1	20	20	0
5	16	16	0
10	20	20	0
25	16	16	0
50	20	20	0
100	6	6	0
150	6	6	0
Total (Noise Power)	140	131	9
Sinusoidal Disturbances (Frequency, rad/s)			
0.0628	76	58	18
0.628	76	70	6
1	76	69	7
5	40	39	1
Total (Sinusoidal Disturbances)	268	236	32

4.4.1.6 Limitations Related to Noise, Sinusoidal Disturbances and Sampling Time

The proposed method demonstrates strong performance on the evaluated datasets in identifying the primary root cause as the top-ranked variable; however, certain limitations arise when dealing with noise and sinusoidal disturbances.

Table 4.5 summarizes detection performance across different noise powers, highlighting occasional inconsistencies at minimal noise powers. While the method remains robust under high noise powers, maintaining accurate detection, its performance degrades at lower noise powers (e.g., 0.1, 0.5). The increased number of incorrect detections suggests difficulty in distinguishing minor signal fluctuations from normal process variations.

Similarly, Table 4.5 presents the detection performance for various sinusoidal disturbance frequencies, emphasizing potential challenges in detecting lower-frequency oscillations. The method shows reduced responsiveness to low-frequency disturbances (e.g., 0.0628 rad/sec), indicating that longer-period oscillations are more challenging to identify accurately.

For all example systems, root cause analysis was performed using a sampling time of 1 s. To evaluate the impact of sampling time, additional tests were conducted on Example 1 using three different sampling times: 0.1 s, 1 s, and 10 s. Across 52 datasets, the number of correct predictions in the first position remained relatively stable, with 42, 43, and 39 correct identifications for sampling times of 0.1 s, 1

s, and 10 s, respectively. When considering the first two predictions, the correct identifications were 48, 50, and 48, respectively. These results indicate that variations in sampling time have minimal effect on detection accuracy, demonstrating the robustness of the proposed method across different sampling intervals.

Despite these limitations, the method remains a valuable tool for root cause analysis in process industries. Future enhancements should focus on refining sensitivity to weak disturbances at low noise powers and improving detection accuracy for slow-varying oscillations.

4.4.1.7 Comparison with Spectral Envelope Method

The generated datasets are also tested using the spectral envelope method [43]. The spectral envelope method is a frequency-domain technique for detecting and diagnosing plant-wide oscillations. Consider a multivariate time series of standardized process variables, denoted by $Z(t)$. For each frequency f, the corresponding spectral density matrix is defined as $S(f)$. The spectral envelope is then given by

$$\lambda(f) = \max \text{ eig}\left(V^{-\frac{1}{2}} S(f) V^{-\frac{1}{2}}\right),$$

where V is the covariance matrix of the standardized variables. The spectral envelope curve $\lambda(f)$ represents the proportion of signal power concentrated at frequency f. The dominant oscillatory frequency of the system is identified as

$$f^* = \arg\max_f \lambda(f).$$

At this dominant frequency, the eigenvector associated with $\lambda(f^*)$ provides the relative contribution of each process variable to the oscillation. Variables with the largest contributions are identified as the most likely sources of the oscillatory behavior [43].

For comparison, the spectral envelope method [43] performs well for datasets with sinusoidal disturbances, as it is limited to oscillatory faults. Among the 268 cases

with sinusoidal faults, the method correctly identifies the root cause in 259 cases for the smaller example systems and in 13 out of 14 cases for the larger system, yielding an overall accuracy of 96.45%. Table 4.6 presents the identification accuracy for the proposed τ-metric and spectral envelope method.

Table 4.6: Comparison of identification accuracy for the proposed τ-metric and spectral envelope method

System	Disturbance	Dataset	Accuracy (%)	
			Proposed Method	Spectral Envelope Method [43]
Example 1 Example 2 Example 3 Large System	White noise Sine wave	161 282	94.41 88.65	— 96.45
Overall	–	443	90.74	–

4.4.2 Industrial case studies

4.4.2.1 CSTR case study

In Fig. 4.5, a Continuous Stirred Tank Reactor (CSTR) is depicted, involved in an exothermic reaction $A(l) \rightarrow B(l) + C(g)$. The CSTR case study is a well-known example in diagnostic studies[3, 142, 143] and is used to assess the performance of the proposed τ-metric. A comprehensive CSTR model description is available in Vachhani, 2005[3]. Temperature (T) within the reactor is regulated by adjusting the flow rate of the coolant (F_c) circulating through the jacket. Modulation of the reactor's level (V) is accomplished by varying the outlet flow rate (F), while control over the reactor pressure (P) entails adjustments to the vent gas flow rate (F_{vg}). In addition to these three controlled variables, measurements are taken for concentrations within the reactor(C_A), number of moles(n) in gas phase of reactor and the temperature at the coolant inlet(T_{ci}) and outlet(T_c).

The CSTR model is simulated using MATLAB Simulink by varying noise power for non-oscillatory faults and sine wave parameters (amplitude and frequency) for oscillatory faults. A total of 12 datasets are generated by perturbing different controlled variables (PV) with both oscillatory and non-oscillatory faults. Each dataset has 10 measurements including controlled and manipulated variables. The

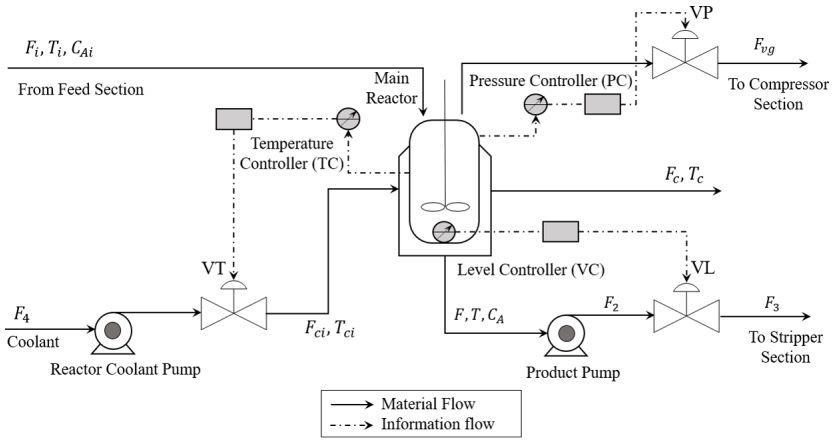


Figure 4.5: CSTR model [3]

outcomes obtained by applying the τ-metric for the CSTR case study are presented in Table 4.7.

Table 4.7: Results of the proposed τ-method for the CSTR model

Cases	Actual root cause (PV variable)	Corresponding OP variable	Top-ranked variable identified
White noise (Noise power = 0.5)	V	F	F
	T	F_c	F_c
	P	F_{vg}	P
White noise (Noise power = 2)	V	F	V
	T	F_c	F_c
	P	F_{vg}	P
Sine Wave(Amplitude = 15, Frequency = 6.28 rad/hr)	V	F	V
	T	F_c	F_c
	P	F_{vg}	P
Sine Wave(Amplitude = 25, Frequency = 15.7 rad/hr)	V	F	V
	T	F_c	P
	P	F_{vg}	P

The τ-metric has successfully identified the top-ranked variable associated with fault in 11 out of the 12 datasets created for the CSTR case study. In the case of a sinusoidal disturbance with parameters amplitude=25 and frequency=15.7 rad/hr, despite the actual root cause being the reactor temperature (T), the τ-metric incorrectly predicted pressure (P) as the top-ranked variable. It's noteworthy that the outlet coolant temperature, T_c, had the second-highest value and is a plausible alternative highly-ranked variable according to the τ-metric, suggesting changes in either coolant flow rate (F_c) or inlet coolant temperature (T_{ci}), both of which are manipulating variables for reactor temperature (T). Additionally, the τ-metric identified reactor temperature, T, as the third highest-ranked variable associated with fault in this scenario.

4.4.2.2 Tennesssee Eastman Process

To exemplify large-scale process industries, we shift our focus to the extensively examined Tennessee Eastman benchmark process. This complex operation entails the synthesis of substances G and H from the initial components A, C, D, and E. Alongside these six components, there is an inert substance B, and a by-product F associated with the overall process.

The reactions are as follows:

$$\mathrm{A(g) + C(g) + D(g) \rightarrow G(l)} \quad \text{(Product 1)}$$

$$\mathrm{A(g) + C(g) + E(g) \rightarrow H(l)} \quad \text{(Product 2)}$$

$$\mathrm{A(g) + E(g) \rightarrow F(l)} \quad \text{(Byroduct)}$$

$$\mathrm{3D(g) \rightarrow 2F(l)} \quad \text{(Byroduct)}$$

The operation comprises five key components: a reactor, a condenser, a product separator, a compressor, and a product stripper. A comprehensive list of 41 measured variables($xmeas$) and 12 manipulated variables(xmv), and additional detailed information about this process is accessible in other references [144–148]. MATLAB codes and Simulink models for simulating this process are accessible at [149].

Faults are introduced in manipulated variables with different sinusoidal signals and white noise. Table 4.8 shows the perturbed variable and the corresponding control loop variables. Since the intent of the algorithm is to localize the fault to the correct control loop, the effectiveness of the algorithm is evaluated based on whether the τ-metric ranks one of the variables in the faulty control loop as the top-ranked variable associated with the fault. A total of 40 datasets are generated each for sinewave and white noise. For generating datasets, the amplitude and frequency for sinusoidal signals are varied in the range of [5 15] and [5 15], respectively. For datasets with white noise disturbances, the range of noise power

Table 4.8: Perturbed variable and the corresponding control loop variables for the Tennessee Eastman problem. Algorithm prediction is considered correct if it identifies the root cause as one of the variables in the same control loop, i.e., the algorithm is judged based on its ability to localize the root cause to the faulty control loop.

Perturbed Variable (Root cause)	**Corresponding control loop variables**
$xmv1$	$xmv1$, $xmeas2$
$xmv2$	$xmv2$, $xmeas3$
$xmv3$	$xmv3$, $xmeas1$
$xmv4$	$xmv4$, $xmeas4$
$xmv5$	$xmv5$, $xmeas5$
$xmv6$	$xmv6$, $xmeas10$
$xmv7$	$xmv7$, $xmeas14$
$xmv8$	$xmv8$, $xmeas17$
$xmv9$	$xmv9$, $xmeas19$
$xmv10$	$xmv10$, $xmeas9$
$xmv11$	$xmv11$, $xmeas11$
$xmv12$	$xmv12$

Table 4.9: Top-ranked variable identification using τ-metric for Tennessee Eastman problem

Cases	**Dataset**	**Predicted as First**		**Predicted in First 2**		**Predicted in First 3**		**Predicted in First 4**	
		Count	**%**	**Count**	**%**	**Count**	**%**	**Count**	**%**
White Noise	40	32	80	34	85	36	90	38	95
Sine wave	40	23	57.5	28	70	31	77.5	34	85

taken is between [0.5 10]. For each dataset, data from all 53 variables is considered for analysis. We identify and report the first three and first four top-ranked variables obtained using the τ-metric for the Tennessee Eastman case study, and the corresponding percentages of correct predictions are presented in Table 4.9.

From Table 4.9, note that τ-metric gives an accuracy of 90% and 95% in correctly ranking the faulty variable within the first three and first four positions, respectively, for white noise disturbances. Also, the proposed τ-metric correctly ranks the faulty variable within the first three and first four positions in 77.5% and 85% of the sinusoidal disturbances cases, respectively. Additionally, the τ-metric ranks the faulty variable as the top-ranked variable in 32 out of 40 cases for white noise faults, translating to an accuracy of 80%. For sinusoidal disturbances, it ranks the faulty variable first in 23 out of 40 cases, or 57.5%. When considering the first two ranked variables, the τ-metric places the faulty variable within the top two positions in 34 out of 40 cases for white noise (85%) and in 28 out of 40 cases for sinusoidal disturbances (70%).

The spectral envelope method which works only for oscillatory variables when

tested on white noise faults gives an accuracy of less than 40%, showing a poor performance as expected. For sinusoidal faults, spectral envelope method correctly identifies the root cause (predicted in first 4) in 62.5% cases (without standardization) and 80% cases (with standardization). The proposed algorithm performs better than the Spectral envelope method, even for sinusoidal faults. This shows that the proposed algorithm has the potential to help operators prioritize the investigation to a smaller subset of variables; 4 out of 53 variables, giving an accuracy of 95% for white noise faults and 85% oscillatory faults. The detailed results for the CSTR and Tennessee Eastman industrial processes are summarized in Table 4.10.

Table 4.10: Summary of Top-ranked variable Identification Accuracy for Industrial Case Studies

System	Disturbance	Dataset	Accuracy (%)	
			Proposed Method	Spectral Envelope Method [43]
CSTR	White noise	6	100	—
	Sine wave	6	83.33	—
Tennessee Eastman	White noise	40	95	—
	Sine wave	40	85 (predicted in first 4)	80 (predicted in first 4)

4.4.2.3 Multi-Fault Scenario in the Tennessee Eastman Process

In real-world process industries, multiple faults may occur simultaneously, either due to independent failures or fault propagation across interconnected subsystems. To evaluate the robustness of the proposed τ-metric under such conditions, we conducted a controlled multi-fault simulation using the Tennessee Eastman (TE) Process [145].

Two faults were introduced simultaneously to represent distinct types of anomalies. A sinusoidal disturbance was applied to $xmv2$ (amplitude = 15, frequency = 20 rad/hr), while white noise was injected into $xmv7$ (noise power = 0.5). The simulation was conducted using the MATLAB implementation of the Tennessee Eastman process, with all 53 variables monitored, including 41 measurement variables ($xmeas$) and 12 manipulated variables (xmv). The τ-metric was computed for each variable to identify fault sources, and Table 4.11 presents the top 5 vari-

ables ranked by their τ scores.

Table 4.11: Top-ranked variables based on τ-metric in a multi-fault TE process scenario with perturbations on $xmv2$ (sine) and $xmv7$ (white noise).

Perturbed Variables	**Control Loop Variables**	**Top τ-Ranked Variables with Values**
$xmv2$ (sine) $xmv7$ (white noise)	$xmv2$, $xmeas3$ $xmv7$, $xmeas14$	1. $xmeas3$: 2.01×10^{15} 2. $xmv2$: 3.44×10^{9} 3. $xmv7$: 1.23×10^{9} 4. $xmeas22$: 1.09×10^{9} 5. $xmeas14$: 1.85×10^{8}

The directly perturbed manipulated variables, $xmv2$ and $xmv7$, are correctly ranked among the top τ-scoring variables, confirming effective fault detection. Their corresponding measurement variables, $xmeas3$ and $xmeas14$, also appear among the top-ranked variables, demonstrating accurate localization to the relevant control loops. Overall, this case study illustrates the potential of the τ-metric under multi-fault conditions. The metric successfully identifies both directly affected variables and dependent variables within associated control loops, supporting real-time fault localization in industrial settings where multiple anomalies may occur simultaneously. This evaluation complements the single-fault analyses and highlights the practical applicability of the proposed approach.

4.5 Computational Efficiency Analysis

To evaluate the computational efficiency of the proposed method, we compared its average detection time against the spectral envelope approach [43] across four different example systems. Average detection time is the mean time required to analyze a single dataset using a given method. Table 4.12 presents the number of datasets, dataset dimensions (simulation length $\times$ number of variables), and the corresponding average detection times. The datasets used vary significantly in dimensions, providing a comprehensive evaluation of the proposed method's efficiency and scalability. All detection time measurements were obtained on a Dell Precision 3630 Tower (Intel Core i7-8700, 3.31 GHz, 32 GB RAM).

The proposed method consistently demonstrates a significant reduction in average detection time across all examples. For instance, in Example 1, the proposed

Table 4.12: Comparison of Average Detection Time Between Spectral Envelope and Proposed Method

Example System	Fault	Dataset	Dataset Dimensions	Average Detection Time (sec)	
				Spectral Envelope[43]	Proposed Method
Example 1	Sine wave	36	5000×8	4.402	0.077
Example 2	Sine wave	72	901×12	0.264	0.061
Example 3	Sine wave	160	901×20	0.326	0.133
Large System	Sine wave	14	1000×100	3.066	1.664

method achieves an average detection time of 0.077 seconds, which is substantially lower than the 4.402 seconds recorded by the Spectral Envelope method. Even for the large dataset (100 variables), the proposed method achieves a notable reduction, completing detection in 1.664 seconds compared to the 3.066 seconds required by the Spectral Envelope method, showcasing its efficiency in handling high-dimensional data.

These results demonstrate that the proposed method is computationally efficient for the tested datasets and system sizes, indicating its potential suitability for real-time or near-real-time applications in process monitoring and control. The reduction in detection time without compromising accuracy highlights the practical benefits of the method for large-scale industrial applications.

4.6 Comparison with Granger Causality Analysis

In addition to evaluating the computational efficiency of the proposed method against the spectral envelope approach, we also consider Granger causality, a widely used statistical method for causal inference in time-series data. Although Granger causality was originally developed for applications in economics and finance, it has also been employed in process systems for root cause analysis [84, 150]. Our purpose here is not to present it as a direct performance baseline, but rather to highlight some of the practical challenges that arise when applying such methods to high-dimensional industrial data.

Granger causality assesses whether the past values of one time series can improve the prediction of another, thereby identifying potential causal relationships. For two stationary time series $X(t)$ and $Y(t)$, we can compare two autoregressive

models. The first model predicts $X(t)$ using only its own past values:

$$X(t) = \sum_{i=1}^{k} a_i X(t-i) + \epsilon_{X|X}, \tag{4.18}$$

and the second model includes the past values of $Y(t)$:

$$X(t) = \sum_{i=1}^{k} a_i X(t-i) + \sum_{i=1}^{k} b_i Y(t-i) + \epsilon_{X|X,Y}, \tag{4.19}$$

where k is the model order, a_i and b_i are coefficients, and $\epsilon_{X|X}$ and $\epsilon_{X|X,Y}$ are the residuals of the respective models. The Granger causality measure from Y to X is then defined as

$$F_{Y\to X} = \ln \frac{\mathrm{var}(\epsilon_{X|X})}{\mathrm{var}(\epsilon_{X|X,Y})} \tag{4.20}$$

A statistically significant reduction in the prediction error variance indicates that $Y(t)$ Granger-causes $X(t)$. For a detailed description and further theoretical background, the reader is referred to the work by Yuan and Qin [84].

Fig. 4.6 illustrates the Granger causality network obtained for a large system with 50 process variables (PV). Each directed edge in the graph represents a statistically significant causal influence, determined using a significance level of $\alpha = 0.05$. The resulting network exhibits a highly dense and interconnected structure, making it difficult to interpret and extract meaningful insights.

Apart from interpretability, Granger causality also suffers from high computational complexity. The method requires fitting multiple autoregressive models for each variable, followed by statistical hypothesis testing. As the number of variables and lag orders increases, the computational burden grows significantly. To obtain the causal network, Granger causality required 98.83 seconds. Furthermore, while Granger causality identifies causal connections, it does not directly pinpoint the root cause of an anomaly. Additional post-processing is necessary to infer the most significant contributors, further increasing computational overhead. Other causality methods such as Transfer Entropy (TE) [85, 86] and Convergent Cross

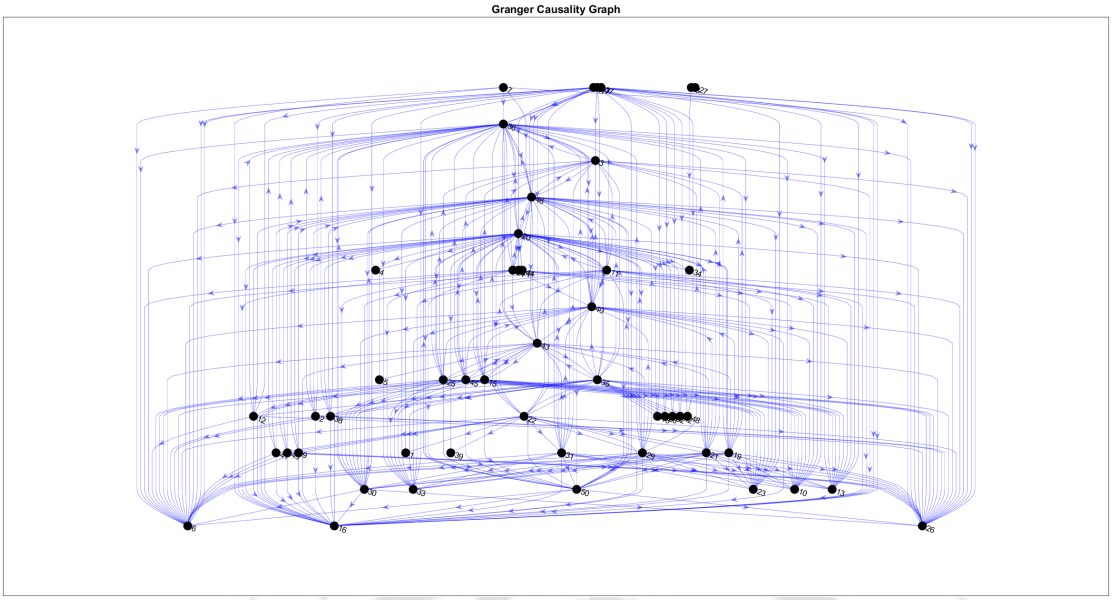


Figure 4.6: Graph representation of Granger causality relationships for a 50-variable system. Each directed edge represents a significant causal influence.

Mapping (CCM) [151] would likely produce similarly dense networks, requiring extensive tuning and interpretation.

In contrast, the proposed method efficiently ranks the most prominent root causes, eliminating the need for manual inference from complex causal graphs, and completes the same analysis in just 1.65 seconds. The proposed method not only significantly reduces computational time but also provides direct and interpretable results, making it a more practical and scalable solution for root cause analysis.

4.7 Discussion

Among the state-of-the-art techniques for root cause analysis, most techniques require process data or process information to generate a causal map of the process and use the causal map to identify the root cause. For example, the adjacency matrix or any causal map-based method requires a list of faulty variables and an adjacency matrix/causal map relating to the concerned variables. The root cause identification based on the adjacency matrix or causal maps is accurate only when there are no errors in the adjacency matrix and the list of faulty variables. This adjacency matrix or causal map must be derived from a process model or

process data. Metrics like Granger causality or Transfer Entropy help identify causal relations or, in essence, the adjacency matrix from process data. However, these methods fail to capture accurate connections due to erroneous information or lack of information about the plant. The causal maps or adjacency matrix generated could be entirely different for slight variations in data. Further, some fault detection techniques are used to identify the faulty variables, which may not correctly identify all the faulty variables. These limitations make it challenging to use these techniques when there is a large number of variables, which is usually the case in practical applications.

A few other techniques, like the spectral envelope method, identify root causes without building a causal map. The spectral envelope method performs well in the case of datasets with sinusoidal disturbances. For example, among the 268 case studies generated with sinusoidal disturbances for smaller example systems, the spectral envelope method correctly identified the root cause for 259 cases. Similarly, for the 14 datasets generated for the larger example system, the correct root cause was identified for 13 cases. Thus, the overall accuracy of the spectral envelope method for predicting the root cause in the case of sinusoidal faults is 96.45%. The method provides an accuracy of 72.5% for the Tennessee Eastman case study with sinusoidal faults. However, this method can only detect root cause in case of oscillatory faults and cannot be used for non-oscillatory faults. Since oscillatory and non-oscillatory faults are common in any process plant, a simple method that can identify the root cause for both types of faults is essential.

As seen from the study, the proposed τ-metric can identify top-ranked variable associated with the fault with an accuracy of 90.74% (402 cases out of 443) for synthetic datasets from coupled transfer function models and 91.6% for the CSTR case study. For the Tennessee Eastman case study, the algorithm shows an overall accuracy of 83.75% (considering first 3 predictions) and 90% (considering first 4 predictions). Moreover, the metric's simplicity in computation, relying solely on variance calculation and cross-correlation, adds to its practical appeal. In

instances where the proposed approach failed to correctly rank the faulty variable as the first option (variable with a maximum value of τ), the actual faulty variable has been ranked as the second or third dominant candidate in most instances. Hence, to reduce the impact of wrong identifications, we can identify and report the first three dominant variables based on the value of τ. This approach allows plant operators to prioritize their focus on the most probable fault sources for further analysis, thereby aiding in effective troubleshooting and decision-making processes. Although the proposed approach is demonstrated using complete datasets, it is not limited to post-hoc analysis. In practical settings, the τ-metric can be computed over a short, recent window of data following fault detection, sufficient to capture relevant cross-correlation patterns. This enables timely root cause analysis and supports the method's applicability in real-time industrial environments. The tool can assist the operators in making decisions faster since it provides a prioritized list. As the final decision is still made by the operators, we believe that they will be able to trust and use the tool, unlike any other fully data-driven algorithms. While the proposed τ-metric demonstrates strong performance across a variety of synthetic and industrial case studies, its validity is subject to the modeling and experimental assumptions stated in Section 4.3.2. In particular, the method is primarily evaluated under single-fault conditions and locally linear operating regimes, with one additional multi-fault case study,and its diagnostic ranking performance is not theoretically guaranteed in the presence of multiple concurrent faults, strong nonlinearities, or severe non-stationary transients. Furthermore, the τ-metric is designed as a correlation-based diagnostic ranking tool and does not provide formal causal guarantees. Under these conditions, the method should be interpreted as prioritizing variables most strongly associated with observed faults rather than identifying true causal origins. Extending the framework to explicitly handle strongly nonlinear dynamics, persistent non-stationarity, and complex multi-fault scenarios remains an important direction for future work.

4.8 Summary

The work introduces a robust approach based on cross-correlation weighted lag for ranking the most likely variables associated with faults in multivariate processes with interconnected control loops. One of the key advantages of the proposed τ-metric framework is its computational efficiency. Unlike methods that rely on causal graph construction or entropy-based metrics, the proposed approach is grounded in simple statistical calculations, namely, cross-correlation and variance, which are computationally inexpensive. Furthermore, these operations can be performed independently for each pair of variables, allowing for straightforward parallelization. This makes the method highly scalable and suitable for large-scale plant-wide applications. The method's data-driven nature eliminates the need for prior knowledge or complex modeling, while maintaining robust diagnostic performance. Rigorous testing was conducted using 443 synthetic datasets, including 161 cases of white noise disturbances and 282 cases of sinusoidal disturbances, generated for various multivariate processes with connected control loops. The proposed τ-metric accurately ranked the faulty variable as the top-ranked variable associated with the fault in 94.41% of cases with white noise disturbances and 88.65% of cases with sinusoidal disturbances. Furthermore, when applied to industrial case studies such as the CSTR and the Tennessee Eastman Process, the proposed τ-metric exhibited good accuracy. While the present study focused on fault scenarios generated synthetically, future research directions may involve testing the algorithm under multiple concurrent faults, on larger datasets, and across different industrial plants to further assess robustness and practical applicability.

Chapter 5

Conclusions & Suggestions for Future Work Direction

5.1 Conclusions

From the present work on fault diagnosis in control loops, the following conclusions are drawn.

1. The neural network-driven algorithms for accurate oscillation detection in process datasets effectively integrate advanced feature engineering techniques grounded in domain expertise. Notably, features from the FFT and the FFT-ACF were utilized, with dominant peaks identified using a cycle count threshold. Methodologies for estimating the period of oscillation and amplitudes also performed well for both synthetic and industrial datasets.

2. For the synthetic dataset, a cycle count threshold of 2 to 3 gave the highest accuracy of 96% when using a feature set of 802 features derived from both FFT and FFT-ACF spectra. On a benchmark industrial dataset, the method achieved an accuracy of 86.02%, demonstrating its ability to detect and classify oscillations in complex industrial datasets. Additionally, the trained neural network model performed the complete analysis in just a few seconds, making the method highly efficient for industrial applications.

3. For estimating the period of oscillation, the method achieved a prediction accuracy of 99.3% for regular oscillations within a ±5% tolerance for the synthetic dataset and achieved 86.9% accuracy within a ±15% tolerance for irregular oscillations. The amplitude estimation results showed that the method consistently predicted amplitudes close to the actual values. Period of oscillation and amplitude estimations for the industrial dataset also showed accurate results.

4. A simple data-driven and computationally efficient method for detecting stiction in industrial control loops was presented. The proposed approach generates heatmaps from multiple time-shifted versions of the normalized time series signal. These generated heatmaps are used to highlight the slow-varying regions, and then cycle-wise PV/OP heat ratios are computed. The average of these heat ratios is then used as a feature for threshold-based classification. The optimal threshold was obtained using ROC analysis, and the proposed method was tested a set of 20 industrial loops. For optimal heatmap parameters of $n_s = 10$ and $\alpha_{\text{shift}} = 2$, the method correctly classified 19 out of 20 test loops, achieving a high accuracy of 95%.

5. Sensitivity analysis showed that moderate numbers of shifts are sufficient to reveal the slow dynamics and are computationally efficient. Overall, the proposed method generalizes well across a wide range of industrial loops and offers a good balance between accuracy, simplicity, and computational efficiency, making it suitable for near real-time monitoring and diagnosis of control loop stiction in industrial settings.

6. Chapter 4 introduces a robust approach based on cross-correlation weighted lag for root cause identification in multivariate processes with interconnected control loops. One of the key advantages of the proposed τ-metric framework is its computational efficiency. Unlike methods that rely on causal graph construction or entropy-based metrics, the proposed approach is grounded in

simple statistical calculations, namely, cross-correlation and variance, which are computationally inexpensive. Furthermore, these operations can be performed independently for each pair of variables, allowing for straightforward parallelization. This makes the method highly scalable and suitable for large-scale plant-wide applications. The method's data-driven nature eliminates the need for prior knowledge or complex modeling, while maintaining robust diagnostic performance.

7. Rigorous testing was conducted using 443 synthetic datasets, including 161 cases of white noise disturbances and 282 cases of sinusoidal disturbances, generated for various multivariate processes with connected control loops. The proposed τ-metric accurately identified the root cause in 94.41% of cases with white noise disturbances and 88.65% of cases with sinusoidal disturbances. Furthermore, when applied to industrial case studies such as the CSTR and the Tennessee Eastman Process, the proposed τ-metric exhibited good accuracy.

8. Overall, the studies presented in this thesis demonstrate the potential of simple, data-driven, and computationally efficient methods for improving fault diagnosis and control loop performance in industrial processes. Together, they form a cohesive framework for oscillation detection, stiction diagnosis, and root cause analysis that can support intelligent performance monitoring in modern process industries.

5.2 Future Scope

There are still some unsolved issues in connection with the content of this thesis, which are briefly described below.

1. **Oscillation Detection:** Future work could aim on improving the technique for irregular oscillations and exploring its scalability and adaptability to various industrial processes.

2. **Oscillation Detection:** The current approach uses a fixed cycle count threshold θ_c, but future efforts could explore adaptive threshold selection based on signal characteristics, such as spectral content, noise level, loop type, and available prior knowledge.

3. **Oscillation Detection:** Integrating the proposed method with plant-wide oscillation diagnosis frameworks would enhance its practical deployment in industrial settings, enabling large-scale and automated monitoring of control loop performance.

4. **Stiction Detection:** Practitioners could apply denoising or signal preprocessing techniques in case of loops with high noise levels.

5. **Stiction Detection:** The heatmaps themselves could be used for advanced analytics, such as image-based or deep learning approaches, enabling real-time, continuous monitoring in evolving industrial environments, and enhancing their applicability in large-scale process control systems.

6. **Root Cause Analysis:** Future research directions may involve testing the algorithm under multiple concurrent faults, on larger datasets, and across different industrial plants to further assess robustness and practical applicability.

7. **Integrated Framework:** Integrating all the proposed method for oscillation detection, stiction diagnosis, and root cause analysis into a single, plant-wide intelligent fault diagnosis framework would strengthen their industrial utility and pave the way for fully automated process monitoring and diagnosis.

8. **Web-Based Tool Development:** As part of the future work, a web-based tool is being developed to enable users to upload process data and perform individual analyses such as oscillation detection, stiction diagnosis, and root cause analysis, or execute a combined, all-in-one diagnostic analysis. Once

completed, this tool can be deployed online as an interactive platform to demonstrate the practical applicability of the proposed methodologies and facilitate their use by industrial practitioners.

Appendices

भारतीय प्रौद्योगिकी संस्थान गुवाहाटी
Indian Institute of Technology Guwahati

Appendix A

Confusion Matrix and Evaluation Metric Computation

This appendix provides a brief explanation of the confusion matrix and how key evaluation metrics accuracy, precision, and recall are computed for the classification model.

A.1 Confusion Matrix

The confusion matrix is a summary of prediction results on a classification task. For a multi-class classification with three classes (Class 0, Class 1, and Class 2), it is structured as follows:

- **Rows** represent the actual (true) class labels.
- **Columns** represent the predicted class labels.
- Each cell $[i, j]$ indicates the number of samples from actual class i predicted as class j.

For example, the confusion matrix:

$$\begin{bmatrix} 3928 & 13 & 44 \\ 100 & 2848 & 79 \\ 147 & 43 & 2798 \end{bmatrix}$$

indicates that 3928 instances of Class 0 were correctly classified as Class 0, while 13 and 44 instances of Class 0 were misclassified as Class 1 and Class 2, respectively and so on.

A.2 Evaluation Metrics

From the confusion matrix, the following metrics are computed for each class $k \in \{0, 1, 2\}$, where k represents the class label:

- **True Positives (TP):** Correct predictions for the class (diagonal entry).
- **False Positives (FP):** Instances from other classes incorrectly predicted as this class.
- **False Negatives (FN):** Instances of this class incorrectly predicted as another class.

Using these values, the evaluation metrics are computed as:

- **Accuracy**

$$\text{Accuracy} = \frac{\text{Total correct predictions}}{\text{Total number of instances}}$$

- **Precision (per class)**

$$\text{Precision}_k = \frac{\text{TP}_k}{\text{TP}_k + \text{FP}_k}$$

- **Recall (per class)**

$$\text{Recall}_k = \frac{\text{TP}_k}{\text{TP}_k + \text{FN}_k}$$

Appendix B

Preliminary Results for the Synthetic Dataset

B.1 Using only the FFT of the time-series data

This section presents the results for the test dataset using features derived solely from the FFT of the time series data. A subset of $2m + 1$ features, including the maximum FFT value, was selected, and a neural network model with the same structure was developed for each value of m. A sensitivity analysis was conducted, and the results for precision, recall, and accuracy for different feature configurations are summarized in Table B.1.

The accuracy progressively increases from 62.47% to 96.1% on the test set as the number of features expands from 11 (m = 5) to 4097 (full data). The use of 4097 features aligns with the work of Dambros et al. (2019c) [97], who achieved an accuracy of 97% in their study. Employing all 4097 features results in the highest precision for both class 0 and class 1, as well as the highest recall for class 0 and class 2. However, this approach leads to a reduction in precision for class 2 and a decrease in recall for class 1. The configuration using 401 features achieved an accuracy of 95.48%, results indicate that reducing the number of features(from 4097 to 401) still maintains a high level of accuracy while significantly reducing

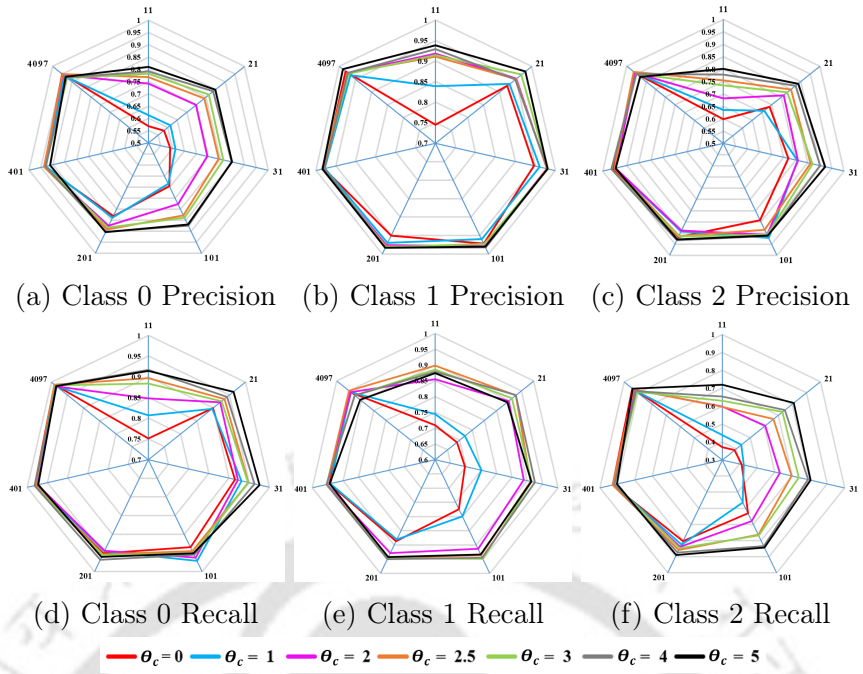
Table B.1: Results of Oscillation Detection and Sensitivity Analysis Using only FFT Features from Data for $\theta_c = 0$.

No. of Features	Confusion Matrix	Accuracy	Precision			Recall			Computational time
			Class 0	Class 1	Class 2	Class 0	Class 1	Class 2	
$(2m+1)$ 11 $(m=5)$	$\begin{bmatrix}2992 & 434 & 559\\ 698 & 2145 & 184\\ 1577 & 301 & 1110\end{bmatrix}$	62.47%	0.568	0.745	0.599	0.751	0.709	0.371	28.5s
21 ($m=10$)	$\begin{bmatrix}3587 & 100 & 298\\ 833 & 2082 & 112\\ 1762 & 71 & 1155\end{bmatrix}$	68.24%	0.580	0.924	0.738	0.900	0.688	0.387	33.6s
31 ($m=15$)	$\begin{bmatrix}3645 & 66 & 274\\ 823 & 2110 & 94\\ 1708 & 51 & 1229\end{bmatrix}$	69.84%	0.590	0.947	0.770	0.915	0.697	0.411	42.3s
101 ($m=50$)	$\begin{bmatrix}3725 & 15 & 245\\ 578 & 2344 & 105\\ 1051 & 52 & 1885\end{bmatrix}$	79.54%	0.696	0.972	0.843	0.935	0.774	0.631	49.5s
201 ($m=100$)	$\begin{bmatrix}3791 & 43 & 151\\ 281 & 2685 & 61\\ 489 & 99 & 2400\end{bmatrix}$	88.76%	0.831	0.950	0.919	0.951	0.887	0.803	54.6s
401 ($m=200$)	$\begin{bmatrix}3914 & 16 & 55\\ 100 & 2861 & 66\\ 171 & 44 & 2773\end{bmatrix}$	95.48%	0.935	0.979	0.958	0.982	0.945	0.928	67.1s
4097	$\begin{bmatrix}3946 & 5 & 34\\ 80 & 2850 & 97\\ 123 & 51 & 2814\end{bmatrix}$	96.10%	0.951	0.981	0.956	0.990	0.942	0.942	195.2s

computational complexity. Additionally, the model training time for 401 features was approximately 67.1 seconds, whereas training the model with 4097 features took over three minutes, specifically around 195.2 seconds, for the training set. Figure B.1 shows the variation of precision and recall for all classes using subset of features from only FFT spectra of the time series data and incorporating the threshold θ_c. Addition of θ_c improves precision and recall across all classes particularly with higher θ_c values. Accuracy obtained for different θ_c value is presented in Table B.2. Accuracy of more than 80% was achieved with just 11 features for

Table B.2: Accuracy Results for Subsets of FFT Data using different θ_c values

No. of Features $(2m+1)$	Accuracy (%)						
	$\theta_c = 0$	$\theta_c = 1$	$\theta_c = 2$	$\theta_c = 2.5$	$\theta_c = 3$	$\theta_c = 4$	$\theta_c = 5$
11 ($m=5$)	62.47	67.83	77.52	80.74	80.89	82.71	84.37
21 ($m=10$)	68.24	70.60	82.07	85.17	86.29	87.96	89.57
31 ($m=15$)	69.84	72.52	82.48	86.22	87.48	90.00	90.53
101 ($m=50$)	79.54	79.76	86.36	88.78	89.57	91.63	91.32
201 ($m=100$)	88.76	89.00	90.82	92.27	92.10	93.42	93.44
401 ($m=200$)	95.48	94.91	95.31	95.55	95.60	95.43	94.25
4097	96.10	95.20	95.57	96.10	94.50	94.78	94.73



(a) Class 0 Precision (b) Class 1 Precision (c) Class 2 Precision

(d) Class 0 Recall (e) Class 1 Recall (f) Class 2 Recall

Figure B.1: Variation in the performance metrics (precision and recall) of the neural network classifier using only FFT features across different classes with θ_c. Subfigures (a), (b), and (c) represent precision for Classes 0, 1, and 2 respectively, while subfigures (d), (e), and (f) represent recall for Classes 0, 1, and 2.

$\theta_c = 2.5$ compared to 62.67% $\theta_c = 0$. For θ_c values beyond 4, the increase in accuracy is marginal, and in some cases, it even decreases (e.g., for 401 features, the accuracy of 95.43% with $\theta_c = 4$ and 94.25% with $\theta_c = 5$).

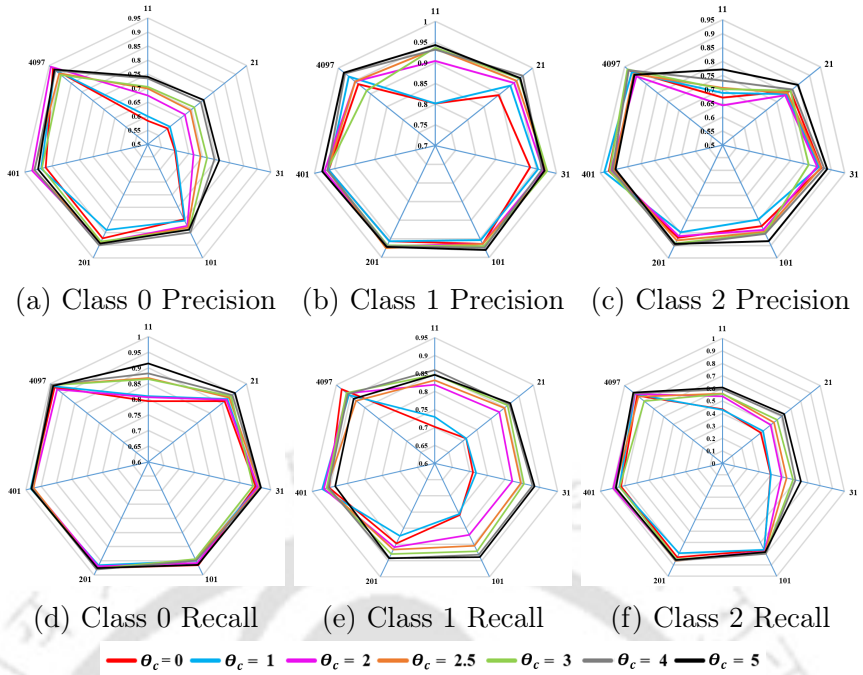
B.2 Using features from the FFT of ACF

The process for selecting a subset of features from the FFT-ACF spectra follows a similar approach. First, the maxima of the FFT-ACF spectra are identified for each variable, and then $2m + 1$ amplitude values around these maxima are considered as features. For each value of m, the neural network architecture described in Section 3.1 is applied. The resulting performance, presented in Table B.3, varies depending on the chosen value of m. While an accuracy greater than 90% is achieved with 401 features, this performance is slightly lower than the case where only a subset of features from the FFT of the data was used. Nevertheless, the predictions indicate that the model successfully predicted the class in some instances where the FFT-only feature set failed.

Table B.3: Results of Oscillation Detection and Sensitivity Analysis Using only FFT of ACF Features from Data for $\theta_c = 0$.

No. of Features	Confusion Matrix	Accuracy	Precision			Recall			Computational time
			Class 0	Class 1	Class 2	Class 0	Class 1	Class 2	
$(2m+1)$ $(m=5)$	$\begin{bmatrix}3163 & 328 & 494\\ 760 & 2124 & 143\\ 1501 & 196 & 1291\end{bmatrix}$	65.78%	0.583	0.802	0.670	0.794	0.702	0.432	35.8s
21 ($m=10$)	$\begin{bmatrix}3631 & 144 & 210\\ 814 & 2151 & 62\\ 1710 & 107 & 1171\end{bmatrix}$	69.53%	0.590	0.896	0.812	0.911	0.711	0.392	44.4s
31 ($m=15$)	$\begin{bmatrix}3786 & 71 & 128\\ 823 & 2146 & 58\\ 1726 & 79 & 1183\end{bmatrix}$	71.15%	0.598	0.935	0.864	0.950	0.709	0.396	50.5s
101 ($m=50$)	$\begin{bmatrix}3851 & 11 & 123\\ 358 & 2299 & 370\\ 617 & 74 & 2297\end{bmatrix}$	84.47%	0.798	0.964	0.823	0.966	0.759	0.769	53.2s
201 ($m=100$)	$\begin{bmatrix}3891 & 9 & 85\\ 174 & 2563 & 290\\ 391 & 108 & 2489\end{bmatrix}$	89.43%	0.873	0.956	0.869	0.976	0.847	0.833	56.4s
401 ($m=200$)	$\begin{bmatrix}3912 & 11 & 62\\ 142 & 2830 & 155\\ 421 & 89 & 2478\end{bmatrix}$	91.20%	0.874	0.965	0.919	0.982	0.902	0.829	65.3s
4097	$\begin{bmatrix}3918 & 8 & 59\\ 75 & 2822 & 130\\ 212 & 177 & 2599\end{bmatrix}$	93.39%	0.932	0.938	0.932	0.983	0.932	0.870	195s

The approach with the FFT of ACF shows a consistent increase in accuracy with the number of features. Accuracy ranges from 65.78% with 11 features to 93.39% with 4097 features. This upward trend indicates that including more features leads to better model performance, suggesting that the model benefits from a richer feature set provided by the FFT of ACF. Precision values for Class 0, Class 1, and Class 2 improve as the number of features increases. For instance, with 11 features, precision values are 0.583 for Class 0, 0.802 for Class 1, and 0.670 for Class 2. With 4097 features, precision increases to 0.932 for Class 0, 0.938 for Class 1, and 0.932 for Class 2. This improvement demonstrates that having more features allows the model to better distinguish between classes. Recall also improves with the number of features. For example, with 11 features, recall values are 0.794 for Class 0, 0.702 for Class 1, and 0.432 for Class 2. With 4097 features, recall improves to 0.973 for Class 0, 0.793 for Class 1, and 0.489 for Class 2. This indicates that more features help the model to capture more true positives.



(a) Class 0 Precision (b) Class 1 Precision (c) Class 2 Precision

(d) Class 0 Recall (e) Class 1 Recall (f) Class 2 Recall

Figure B.2: Variation in the performance metrics (precision and recall) of the neural network classifier using FFT of ACF features across different classes with θ_c. Subfigures (a), (b), and (c) represent precision for Classes 0, 1, and 2 respectively, while subfigures (d), (e), and (f) represent recall for Classes 0, 1, and 2.

Table B.4: Accuracy Results for Subsets of FFT of ACF Data using different θ_c values

No. of Features $(2m+1)$	Accuracy (%)						
	$\theta_c = 0$	$\theta_c = 1$	$\theta_c = 2$	$\theta_c = 2.5$	$\theta_c = 3$	$\theta_c = 4$	$\theta_c = 5$
11 $(m = 5)$	65.78	67.19	72.93	76.38	76.73	78.87	80.27
21 $(m = 10)$	69.53	70.68	76.41	78.43	80.02	82.06	83.13
31 $(m = 15)$	71.15	71.72	77.51	79.78	80.72	83.05	84.60
101 $(m = 50)$	84.47	83.68	86.35	87.01	87.77	88.45	88.89
201 $(m = 100)$	89.43	87.27	90.32	91.00	91.04	91.96	91.49
401 $(m = 200)$	91.20	92.41	93.42	92.60	91.83	92.64	92.06
4097	93.39	93.49	93.02	91.83	91.11	94.06	93.53

The FFT of ACF was combined with θ_c thresholds and sensitivity analysis was performed. As shown in Figure B.2, the precision and recall values for all classes are presented when a subset of FFT of ACF features is used, along with θ_c thresholding. Both precision and recall improve as the θ_c threshold is increased.

Table B.4 presents the accuracy achieved when a subset of the FFT of ACF features is used with cycle count threshold. For $\theta_c = 4$, the accuracy ranges from 78.87% to 94.06%. While the FFT of ACF results in higher accuracy for fewer features, the overall performance is still lower compared to the case where the FFT of the data is used as input features.

Appendix C

Detailed Results for the Industrial Dataset

The complete results for oscillation detection and parameter estimation obtained by applying the proposed technique to the dataset provided by Jelali and Huang[120] are summarized in the table below. The actual class of each data entry is determined based on the available comments and visual inspection of the data.

Table C.1: Detailed Results for the Industrial Dataset

PV Time Series	Oscillation Detection		Quantification	
	True Class	Predicted Class	Time Period	Amplitude
BAS01	0	0	–	–
BAS02	0	0	–	–
BAS03	2	0	–	–
BAS04	2	2	[304.88, 714.29, 331.13, 247.52]	[0.038, 0.050, 0.039, 0.035]
BAS05	1	0	–	–
BAS06	2	2	[12.66, 11.91, 13.52, 10.44]	[0.579 0.564 0.583 0.554]
BAS07	2	2	[62.07, 52.85, 44.29, 75.13]	[0.952, 0.906, 0.854, 1.010]
BAS08	2	0	–	–
CHEM01	2	2	[160.51, 128.04, 146.20, 186.22]	[0.792, 0.733, 0.770, 0.787]
CHEM02	2	2	[215.52, 46.55, 32.38, 28.34]	[6.222, 2.487, 1.714, 1.522]
CHEM03	2	2	[884.96, 840.34, 1315.79, 980.39]	[0.075, 0.075, 0.085, 0.079]
CHEM04	1	1	[16.75, 14.89, 19.32, 12.15]	[7.685, 7.210, 7.648, 7.059]
CHEM05	2	2	[11.11, 10.04, 9.34, 8.84]	[18.342, 18.0899, 17.245 16.083]
CHEM06	2	2	[55.34, 60.24, 47.35, 44.52]	[28.861, 29.059, 26.500, 25.227]
CHEM07	0	0	–	–
CHEM08	2	2	[10000.00, 4166.67, 3125.00, 1639.34]	[2.378, 2.117, 1.448, 0.866]
CHEM09	0	0	–	–
CHEM10	1	1	[4166.67, 3125.00, 1388.89, 2777.78]	[1.584, 1.557, 0.973, 1.530]
CHEM11	1	1	[3703.70, 2857.14, 2564.10, 2380.95]	[2.587, 2.405, 2.334, 2.329]
CHEM12	2	2	[14285.71, 6666.67, 9090.91, 5263.16]	[4.266, 2.807, 3.666, 2.681]
CHEM13	2	2	[480.77, 454.55, 392.16, 416.67]	[1.338, 1.327, 1.297, 1.321]
CHEM14	2	2	[469.48, 395.26, 418.41, 353.36]	[1.115, 1.101, 1.103, 1.056]
CHEM15	2	0	–	–
CHEM16	2	2	[378.79, 272.48, 350.88, 318.47]	[1.098, 0.960, 1.086, 1.036]
CHEM17	2	2	[469.48, 416.67, 392.16, 500.00]	[0.744, 0.738, 0.721, 0.749]
CHEM18	2	2	[342.47, 288.18, 2272.73, 406.50]	[1.916, 1.667, 4.161, 2.066]
CHEM19	1	1	[1587.30, 1234.57, 757.58, 934.58]	[3.938, 3.807, 3.471, 3.576]
CHEM20	2	2	[2380.95, 1162.79, 88.73, 1538.46]	[1.471, 1.058, 0.293, 1.396]
CHEM21	2	2	[104.38, 99.01, 110.99, 2325.58]	[0.286, 0.271, 0.286, 0.833]
CHEM22	2	2	[1612.90, 1098.90, 555.56, 740.74]	[0.172, 0.148, 0.092, 0.111]
CHEM23	1	1	[1282.05, 1123.60, 1041.67, 408.16]	[0.387, 0.384, 0.383, 0.249]
CHEM24	2	2	[543.48, 135.87, 438.60, 363.64]	[48.632, 22.150, 47.631, 44.172]
CHEM25	2	2	[188.68, 172.71, 207.90, 221.24]	[0.086, 0.085, 0.087, 0.088]
CHEM26	1	1	[1492.54, 1282.05, 1136.36, 735.29]	[2.841, 2.776, 2.618, 1.851]
CHEM27	2	2	[1666.67, 1428.57, 1098.90, 2000.00]	[2.269, 2.262, 2.003, 2.302]
CHEM28	1	1	[1515.15, 1176.47, 1010.10, 892.86]	[1.817, 1.712, 1.655, 1.583]
CHEM29	2	2	[1190.48, 1298.70, 1020.41, 1408.45]	[71.39, 72.9, 68.2, 75.08]
CHEM30	2	2	[724.64, 100000.00, 20000.00, 1851.85]	[159.9, 475.2, 324.8, 185.5]
CHEM31	2	2	[109.53, 102.67, 115.47, 97.28]	[653.04, 626.57, 651.91, 626.86]
CHEM32	1	1	[900.90, 847.46, 1020.41, 751.88]	[10.127, 10.214, 10.110, 9.892]
CHEM33	2	2	[109.17, 262.47, 2857.14, 116.96]	[0.261, 0.368, 0.752,0.273]
CHEM34	2	2	[62.19, 75.70, 71.12, 100.30]	[4.799, 5.144, 5.390, 5.643]
CHEM35	2	2	[165.56, 184.84, 198.02, 173.91]	[5.563, 5.751, 5.774, 5.581]
CHEM36	2	2	[1639.34, 1282.05, 980.39, 1136.36]	[1.668, 1.749, 1.575, 1.641]
CHEM37	2	2	[2941.18, 1754.39, 1351.35, 1470.59]	[3.350, 2.690, 2.477, 2.643]
CHEM38	2	2	[1851.85, 1388.89, 2702.70, 826.45]	[0.067, 0.065, 0.072, 0.042]
CHEM39	2	2	[304.88, 518.13, 392.16, 266.67]	[0.009, 0.012, 0.011, 0.010]
CHEM40	0	0	–	–
CHEM41	1	1	[1000.00, 1063.83, 943.40, 1123.60]	[1.287, 1.306, 1.279, 1.317]
CHEM42	1	1	[1000.00, 1052.63, 943.40, 1123.60]	[1.320, 1.333, 1.293, 1.350]
CHEM43	2	2	[1000.00, 1052.63, 943.40, 892.86]	[0.936, 0.973, 0.923, 0.892]
CHEM44	1	1	[20000.00, 14285.71, 10000.00, 9090.91]	[1.503, 1.277, 1.054, 1.030]
CHEM45	0	0	–	–
CHEM46	0	0	–	–
CHEM47	0	2	[990.10, 1052.63, 934.58, 396.83]	[0.512, 0.517, 0.513, 0.301]
CHEM48	0	2	[25000.00, 990.10, 2040.82, 50000.00]	[0.979, 0.185, 0.233, 1.357]
CHEM49	1	1	[1000.00, 1063.83, 925.93, 877.19]	[1.331, 1.336, 1.312, 1.290]
CHEM50	1	1	[1000.00, 1052.63, 497.51, 934.58]	[1.697, 1.703, 1.224, 1.651]

Table C.2: Detailed Results for the Industrial Dataset (Continued)

PV Time Series	Oscillation Detection		Quantification	
	True Class	Predicted Class	Time Period	Amplitude
CHEM51	2	2	[3030.30, 2702.70, 3225.81, 2857.14]	[1.572, 1.483, 1.582, 1.497]
CHEM52	2	2	[990.10, 2380.95, 2222.22, 5000.00]	[1.281, 1.790, 1.781, 2.149]
CHEM53	0	0	–	–
CHEM54	0	0	–	–
CHEM55	2	2	2272.73, 1538.46, 990.10, 1219.51]	[1.590, 1.269, 0.967, 1.141]
CHEM56	0	2	[5000.00, 1000.00, 709.22, 653.59]	[1.154, 0.646, 0.555, 0.512]
CHEM57	2	2	[1000.00, 1052.63, 460.83, 537.63]	[1.608, 1.644, 1.043, 1.157]
CHEM58	0	0	–	–
CHEM59	0	2	[5000.00, 4545.45, 709.22, 990.10]	[0.846, 0.871, 0.410, 0.476]
CHEM60	2	0	–	–
CHEM61	0	0	–	–
CHEM62	0	2	[1000.00, 1052.63, 943.40, 520.83]	[0.544, 0.543, 0.527, 0.366]
CHEM63	2	2	[1000.00, 578.03, 520.83, 666.67]	[1.561, 1.411, 1.359, 1.472]
MET01	1	1	[5.06, 0.33, 0.89, 0.39]	[0.020, 0.005, 0.009, 0.006]
MET02	2	1	[1.34, 0.67, 0.55, 0.37]	[0.005, 0.004, 0.003, 0.002]
MET03	2	1	[0.66, 0.41, 1.74, 1.04]	[0.003, 0.003, 0.005, 0.004]
MIN01	2	2	[6666.67, 3333.33, 4000.00, 7692.31]	[10.339, 7.121, 7.984, 10.945]
PAP01	2	2	[327.87, 194.93, 115.34, 101.11]	[1.758, 1.648, 1.340, 1.090]
PAP02	1	1	[42.23, 44.52, 13.79, 35.16]	[2.330, 2.327, 1.187, 2.196]
PAP03	2	2	[64.52, 73.80, 87.18, 78.74]	[2.674, 2.783, 2.924, 2.837]
PAP04	1	1	[42.23, 44.52, 37.58, 35.16]	[17.908, 18.028, 17.87, 17.652]
PAP05	2	2	[154.56, 143.68, 134.23, 182.15]	[0.086, 0.084, 0.084, 0.089]
PAP06	1	2	[303.03, 0.00, 0.00, 0.00]	[2.698, 0.000, 0.000, 0.000]
PAP07	2	2	[45.33, 210.97, 77.88, 54.73]	[0.043, 0.076, 0.055, 0.047]
PAP08	2	2	[17.47, 18.37, 16.52, 15.28]	[1.528, 1.568, 1.518, 1.489]
PAP09	1	1	[134.23, 120.48, 112.23, 81.10]	[18.386, 18.042, 17.860, 16.450]
PAP10	1	1	[33.85, 31.03, 35.61, 29.46]	[0.685, 0.701, 0.691, 0.681]
PAP11	0	0	–	–
PAP12	2	2	[854.70, 689.66, 775.19, 641.03]	[5.732, 5.452, 5.481, 5.075]
PAP13	1	1	[854.70, 909.09, 746.27, 980.39]	[0.957, 0.959, 0.942, 0.964]
POW01	2	2	[89.61, 111.36, 98.23, 119.05]	[0.337, 0.348, 0.344, 0.351]
POW02	1	1	[292.40, 221.24, 146.20, 207.04]	[0.805, 0.762, 0.627, 0.746]
POW03	2	2	[526.32, 602.41, 671.14, 561.80]	[0.279, 0.285, 0.291, 0.282]
POW04	1	1	[242.13, 255.75, 217.86, 273.22]	[0.503, 0.506, 0.497, 0.510]
POW05	2	2	[242.13, 255.75, 229.89, 121.21]	[0.776, 0.782, 0.764, 0.549]

Appendix D

Detailed Results for Sensitivity Analysis of Hyperparameters

D.1 Ranking Methodology and Performance Evaluation

A comprehensive sensitivity analysis was conducted to assess the impact of key hyperparameters on the neural network's performance in oscillation detection. The hyperparameters analyzed include the optimizer, activation function, batch size, number of epochs, and neuron configuration. The evaluation was based on two key metrics:

- Accuracy (%): A higher value indicates better performance in classification.
- Training Time (sec): A lower value is preferred to ensure computational efficiency.

For each hyperparameter, models were trained with different settings, and the results were ranked. The ranking prioritizes higher accuracy while considering training time efficiency. Rankings were determined based on the median values obtained from multiple runs to ensure robustness against outliers.

Boxplots were used to visualize the impact of hyperparameter variations. A boxplot is a statistical tool that provides a summary of data distribution, highlighting the following elements:

- Median (central line in the box): Represents the middle value of the dataset.
- Interquartile Range (IQR, the box): Shows the spread of the middle 50% of the data.
- Whiskers: Indicate the range of data within 1.5 times the IQR from the quartiles.
- Outliers (points beyond whiskers): Represent values significantly different from the rest of the dataset.

Figure D.1 illustrates the boxplot representation of the data distribution, clearly showing the median, interquartile range (IQR), whiskers, and outliers. This visualization aids in understanding the variability and distribution characteristics of the hyperparameter effects.

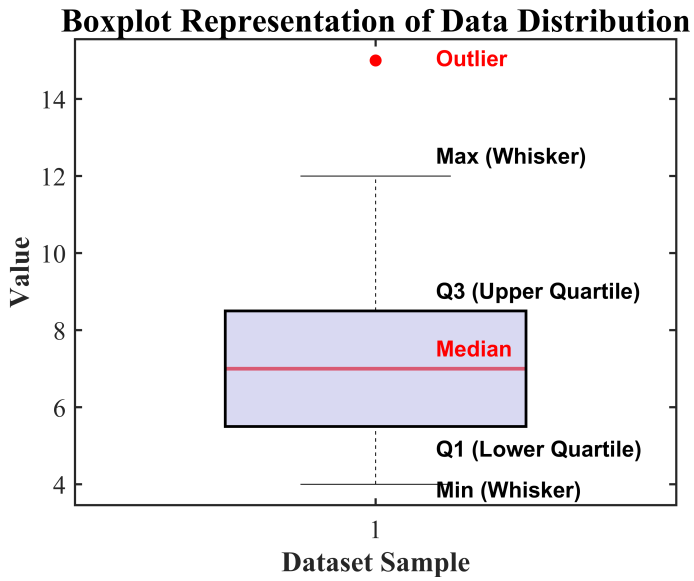
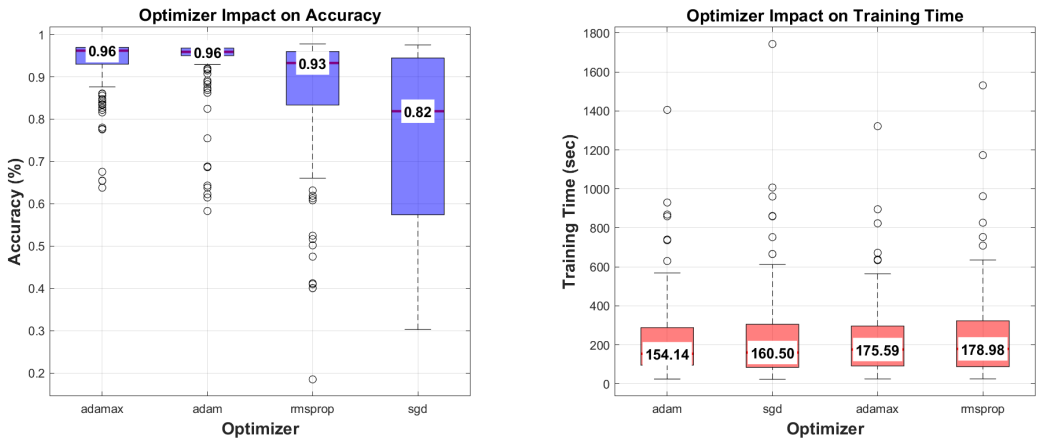


Figure D.1: Boxplot representation of data distribution, illustrating the median, interquartile range (IQR), whiskers, and outliers.

By examining the boxplots, one can observe how different hyperparameter choices impact accuracy and training time. A narrower interquartile range with a high



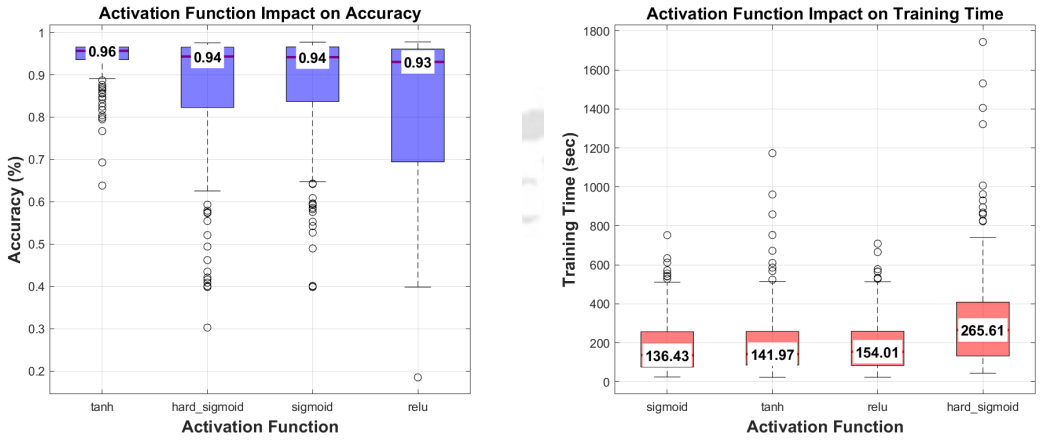
(a) Effect of Optimizer on Accuracy (b) Effect of Optimizer on Training Time

Figure D.2: Impact of Optimizer on Model Performance

median indicates stable and high performance, whereas wider boxes or the presence of extreme outliers suggest variability in performance.

D.2 Impact of Hyperparameters

The results for each hyperparameter are presented in Figure D.2 through Figure D.6. Each figure consists of two subplots: one for accuracy and the other for training time. The evaluation was based on two performance metrics: accuracy (%) and training time (sec). Ranked results for each hyperparameter were obtained, prioritizing higher accuracy and lower training time.



(a) Effect of Activation Function on Accuracy (b) Effect of Activation Function on Training Time

Figure D.3: Impact of Activation Function on Model Performance

D.2.1 Optimizer Impact

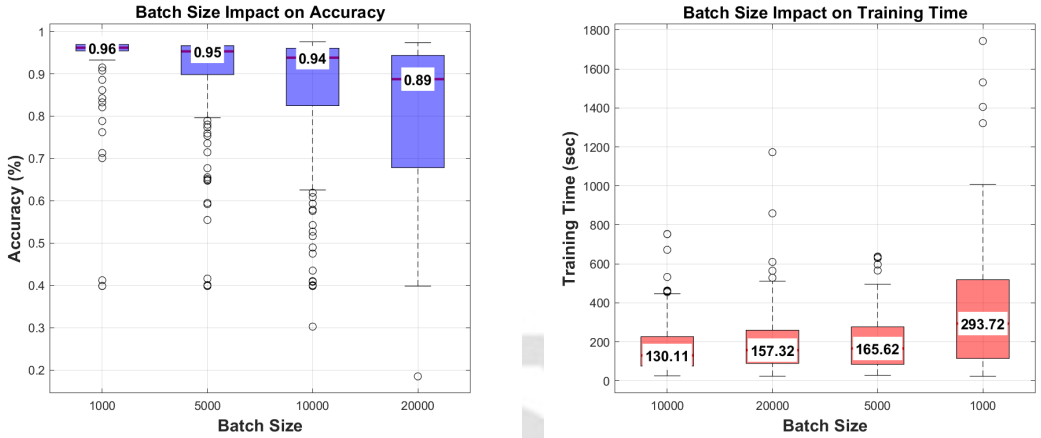
The choice of optimizer significantly affected both accuracy and training time. The Adamax and Adam optimizers achieved the highest median accuracy (96%), while SGD had the lowest accuracy (82%). However, in terms of computational efficiency, Adam resulted in the lowest training time (154.14 sec), closely followed by SGD (160.50 sec). In contrast, RMSprop had the highest training time (178.98 sec). The results indicate that while Adamax and Adam both provide optimal accuracy, Adam offers the best balance between accuracy and computational cost.

D.2.2 Activation Function Impact

The analysis of activation functions showed that Tanh provided the highest accuracy (96%), followed by Hard Sigmoid and Sigmoid (94%). ReLU had slightly lower accuracy (93%). Regarding training time, Sigmoid was the fastest (136.43 sec), whereas Hard Sigmoid was the slowest (265.61 sec). Given its high accuracy and reasonable training time (141.97 sec), Tanh emerges as the best activation function for this model.

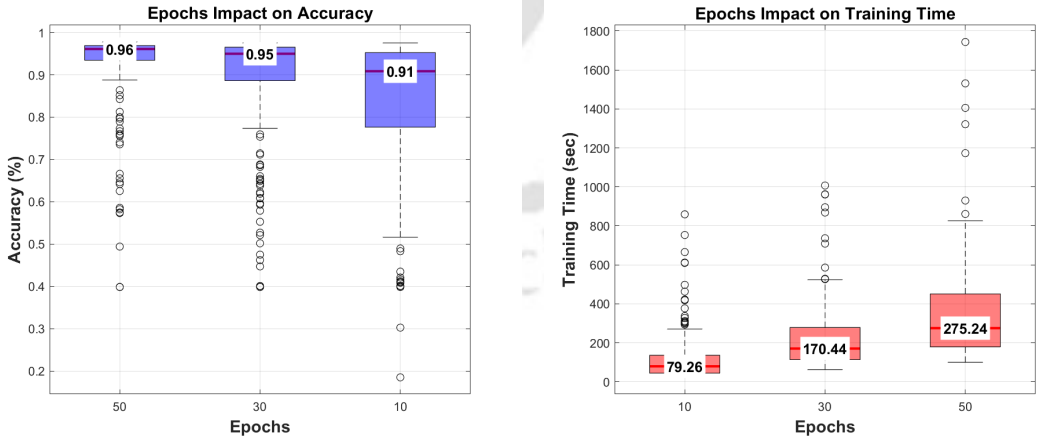
D.2.3 Batch Size Impact

Batch size influences both model convergence and computational efficiency. The highest accuracy (96%) was obtained with batch size 1000, while larger batch sizes (5000 and 10000) resulted in slightly lower accuracy (95% and 94%, respectively). The largest batch size (20000) had the lowest accuracy (89%). However, training time was inversely proportional to batch size, with batch size 10000 providing the fastest training time (130.11 sec). Although batch size 1000 achieved the highest accuracy, its training time was significantly longer (293.72 sec). A batch size of 5000 offers a reasonable trade-off between accuracy (95%) and training efficiency (165.62 sec).



(a) Effect of Batch Size on Accuracy (b) Effect of Batch Size on Training Time

Figure D.4: Impact of Batch Size on Model Performance



(a) Effect of Epochs on Accuracy (b) Effect of Epochs on Training Time

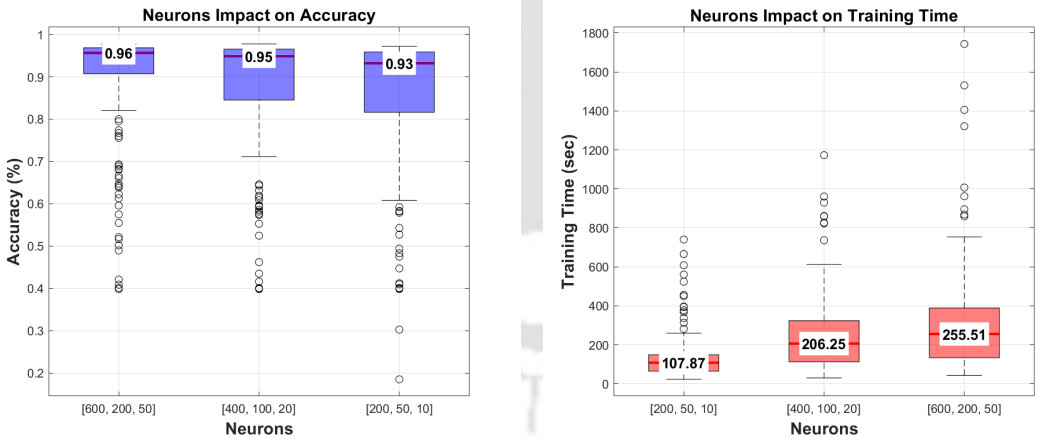
Figure D.5: Impact of Epochs on Model Performance

D.2.4 Epochs Impact

A higher number of epochs generally improved accuracy but increased training time. The best accuracy (96%) was observed at 50 epochs, followed by 30 epochs (95%). Reducing the epochs to 10 resulted in a lower accuracy (91%) but significantly reduced training time (79.26 sec). While 50 epochs yielded the best accuracy, 30 epochs appears to be the optimal choice, balancing accuracy and computational efficiency (170.44 sec training time).

D.2.5 Neuron Configuration Impact

The complexity of the neural network was analyzed by varying the number of neurons in the hidden layers. The configuration [600, 200, 50] achieved the highest accuracy (96%), but required the longest training time (255.51 sec). The smallest configuration, [200, 50, 10], had the lowest accuracy (93%) but trained significantly faster (107.87 sec). The [400, 100, 20] configuration provided a balance with 95% accuracy and a reasonable training time of 206.25 sec, making it a practical choice.



(a) Effect of Neurons on Accuracy (b) Effect of Neurons on Training Time

Figure D.6: Impact of Neurons on Model Performance

D.2.6 Summary of Optimal Hyperparameter Selection

Based on the sensitivity analysis, the following hyperparameter combination is recommended for achieving an optimal balance between accuracy and training

efficiency:

- **Optimizer**: Adam (96% accuracy, 154.14 sec training time)
- **Activation Function**: Tanh (96% accuracy, 141.97 sec training time)
- **Batch Size**: 5000 (95% accuracy, 165.62 sec training time)
- **Epochs**: 30 (95% accuracy, 170.44 sec training time)
- **Neuron Configuration**: [400, 100, 20] (95% accuracy, 206.25 sec training time)

This configuration provides high accuracy while maintaining a reasonable computational cost, ensuring the model is both effective and efficient for oscillation detection.

Appendix E

Detailed Structure of Transfer Functions for Example: 4 (50×50) System

A large multivariable connected control loop system with 50 PV and 50 OP variables is considered.

E.1 Process Transfer Function (G_P)

The process transfer function G_P is represented by the following matrix:

$$G_P = \begin{bmatrix} G_{1,1} & G_{1,2} & \cdots & G_{1,50} \\ G_{2,1} & G_{2,2} & \cdots & G_{2,50} \\ \vdots & \vdots & \ddots & \vdots \\ G_{50,1} & G_{50,2} & \cdots & G_{50,50} \end{bmatrix}$$

All diagonal elements are equal to 1. Except the transfer functions given below, all other transfer functions are kept as 0.

$$G_{2,15} = \frac{-0.24359s + 0.081198}{s^2 + 1.3333s + 0.33333}$$

$$G_{2,23} = \frac{-0.85439s + 0.24411}{3s^2 + 1.8571s + 0.28571}$$

$$G_{2,27} = \frac{-0.73793s + 0.16398}{3s^2 + 1.6667s + 0.22222}$$

$$G_{3,40} = \frac{-0.86666s + 0.24762}{3s^2 + 1.8571s + 0.28571}$$

$$G_{5,19} = \frac{-0.26584s + 0.53169}{3s^2 + 7s + 2}$$

$$G_{5,34} = \frac{-0.5653s + 0.11306}{5s^2 + 2s + 0.2}$$

$$G_{5,35} = \frac{-0.17807s + 0.050876}{2s^2 + 1.5714s + 0.28571}$$

$$G_{7,6} = \frac{-0.31245s + 0.62489}{3s^2 + 7s + 2}$$

$$G_{7,24} = \frac{-0.78767s + 1.5753}{2s^2 + 5s + 2}$$

$$G_{7,39} = \frac{-0.40908s + 0.40908}{4s^2 + 5s + 1}$$

$$G_{8,48} = \frac{-0.10002s + 0.20004}{3s^2 + 7s + 2}$$

$$G_{9,3} = \frac{-0.80809s + 0.26936}{4s^2 + 2.3333s + 0.33333}$$

$$G_{9,16} = \frac{-0.43851s + 0.10963}{5s^2 + 2.25s + 0.25}$$

$$G_{9,30} = \frac{-0.5206s + 0.20824}{2s^2 + 1.8s + 0.4}$$

$$G_{9,36} = \frac{-0.7953s + 0.17673}{3s^2 + 1.6667s + 0.22222}$$

$$G_{9,38} = \frac{-0.50678s + 1.0136}{s^2 + 3s + 2}$$

$$G_{9,45} = \frac{-0.52478s + 0.13119}{5s^2 + 2.25s + 0.25}$$

$$G_{11,10} = \frac{-0.10354s + 0.025885}{2s^2 + 1.5s + 0.25}$$

$$G_{11,30} = \frac{-0.35953s + 0.071906}{3s^2 + 1.6s + 0.2}$$

$$G_{12,9} = \frac{-0.32812s + 0.10937}{s^2 + 1.3333s + 0.33333}$$

$$G_{12,18} = \frac{-0.41423s + 0.092051}{4s^2 + 1.8889s + 0.22222}$$

$$G_{12,45} = \frac{-0.72535s + 0.16119}{2s^2 + 1.4444s + 0.22222}$$

$$G_{13,11} = \frac{-0.42982s + 0.095515}{5s^2 + 2.1111s + 0.22222}$$

$$G_{13,39} = \frac{-0.56608s + 0.16174}{5s^2 + 2.4286s + 0.28571}$$

$$G_{13,42} = \frac{-0.7729s + 1.5458}{3s^2 + 7s + 2}$$

$$G_{14,35} = \frac{-0.44976s + 0.099947}{3s^2 + 1.6667s + 0.22222}$$

$$G_{15,7} = \frac{-0.45705s + 0.13059}{2s^2 + 1.5714s + 0.28571}$$

$$G_{15,18} = \frac{-0.13843s + 0.034608}{2s^2 + 1.5s + 0.25}$$

$$G_{15,42} = \frac{-0.89913s + 0.22478}{3s^2 + 1.75s + 0.25}$$

$$G_{15,46} = \frac{-0.62305s + 1.2461}{3s^2 + 7s + 2}$$

$$G_{17,16} = \frac{-0.53338s + 0.35559}{4s^2 + 3.6667s + 0.66667}$$

$$G_{17,23} = \frac{-0.65402s + 1.308}{4s^2 + 9s + 2}$$

$$G_{17,32} = \frac{-0.87952s + 0.21988}{4s^2 + 2s + 0.25}$$

$$G_{18,34} = \frac{-0.58302s + 0.12956}{4s^2 + 1.8889s + 0.22222}$$

$$G_{19,4} = \frac{-0.16819s + 0.056064}{2s^2 + 1.6667s + 0.33333}$$

$$G_{19,17} = \frac{-0.82654s + 0.33062}{3s^2 + 2.2s + 0.4}$$

$$G_{19,28} = \frac{-0.20508s + 0.058596}{5s^2 + 2.4286s + 0.28571}$$

$$G_{20,46} = \frac{-0.48599s + 0.108}{4s^2 + 1.8889s + 0.22222}$$

$$G_{21,2} = \frac{-0.1441s + 0.032022}{5s^2 + 2.1111s + 0.22222}$$

$$G_{21,28} = \frac{-0.62228s + 0.13828}{2s^2 + 1.4444s + 0.22222}$$

$$G_{22,17} = \frac{-0.60531s + 0.60531}{5s^2 + 6s + 1}$$

$$G_{22,38} = \frac{-0.35042s + 0.35042}{2s^2 + 3s + 1}$$

$$G_{24,26} = \frac{-0.11623s + 0.023247}{s^2 + 1.2s + 0.2}$$

$$G_{24,45} = \frac{-0.21776s + 0.043552}{2s^2 + 1.4s + 0.2}$$

$$G_{25,11} = \frac{-0.35929s + 0.11976}{5s^2 + 2.6667s + 0.33333}$$

$$G_{25,31} = \frac{-0.12612s + 0.036035}{s^2 + 1.2857s + 0.28571}$$

$$G_{25,42} = \frac{-0.47304s + 0.23652}{5s^2 + 3.5s + 0.5}$$

$$G_{26,14} = \frac{-0.21043s + 0.042086}{2s^2 + 1.4s + 0.2}$$

$$G_{26,42} = \frac{-0.63012s + 0.42008}{3s^2 + 3s + 0.66667}$$

$$G_{27,1} = \frac{-0.86661s + 0.19258}{3s^2 + 1.6667s + 0.22222}$$

$$G_{29,34} = \frac{-0.3107s + 0.062141}{4s^2 + 1.8s + 0.2}$$

$$G_{29,41} = \frac{-0.27954s + 0.069884}{2s^2 + 1.5s + 0.25}$$

$$G_{30,1} = \frac{-0.40516s + 0.81031}{3s^2 + 7s + 2}$$

$$G_{30,9} = \frac{-0.11031s + 0.031518}{4s^2 + 2.1429s + 0.28571}$$

$$G_{31,5} = \frac{-0.6605s + 0.33025}{4s^2 + 3s + 0.5}$$

$$G_{31,23} = \frac{-0.34389s + 0.22926}{s^2 + 1.6667s + 0.66667}$$

$$G_{33,4} = \frac{-0.16256s + 0.046447}{2s^2 + 1.5714s + 0.28571}$$

$$G_{34,5} = \frac{-0.7519s + 0.30076}{s^2 + 1.4s + 0.4}$$

$$G_{34,8} = \frac{-0.2101s + 0.070032}{2s^2 + 1.6667s + 0.33333}$$

$$G_{34,38} = \frac{-0.55678s + 0.37119}{2s^2 + 2.3333s + 0.66667}$$

$$G_{35,6} = \frac{-0.18017s + 0.12011}{2s^2 + 2.3333s + 0.66667}$$

$$G_{35,26} = \frac{-0.18628s + 0.04657}{2s^2 + 1.5s + 0.25}$$

$$G_{35,40} = \frac{-0.13429s + 0.067143}{3s^2 + 2.5s + 0.5}$$

$$G_{36,12} = \frac{-0.35537s + 0.14215}{5s^2 + 3s + 0.4}$$

$$G_{36,13} = \frac{-0.76895s + 0.2197}{4s^2 + 2.1429s + 0.28571}$$

$$G_{36,32} = \frac{-0.12713s + 0.05085}{4s^2 + 2.6s + 0.4}$$

$$G_{36,43} = \frac{-0.27648s + 0.06912}{s^2 + 1.25s + 0.25}$$

$$G_{37,19} = \frac{-0.53306s + 0.10661}{5s^2 + 2s + 0.2}$$

$$G_{37,21} = \frac{-0.2542s + 0.2542}{5s^2 + 6s + 1}$$

$$G_{37,29} = \frac{-0.56951s + 0.12656}{5s^2 + 2.1111s + 0.22222}$$

$$G_{37,44} = \frac{-0.59908s + 0.17117}{3s^2 + 1.8571s + 0.28571}$$

$$G_{37,50} = \frac{-0.33058s + 0.094452}{3s^2 + 1.8571s + 0.28571}$$

$$G_{38,25} = \frac{-0.61839s + 0.12368}{s^2 + 1.2s + 0.2}$$

$$G_{38,42} = \frac{-0.87879s + 0.2197}{4s^2 + 2s + 0.25}$$

$$G_{39,34} = \frac{-0.89244s + 0.59496}{3s^2 + 3s + 0.66667}$$

$$G_{40,7} = \frac{-0.60234s + 0.12047}{4s^2 + 1.8s + 0.2}$$

$$G_{40,12} = \frac{-0.50687s + 0.33791}{2s^2 + 2.3333s + 0.66667}$$

$$G_{40,24} = \frac{-0.41054s + 0.082107}{4s^2 + 1.8s + 0.2}$$

$$G_{40,30} = \frac{-0.20998s + 0.083992}{s^2 + 1.4s + 0.4}$$

$$G_{40,49} = \frac{-0.10913s + 0.072752}{4s^2 + 3.6667s + 0.66667}$$

$$G_{41,5} = \frac{-0.54307s + 0.36205}{3s^2 + 3s + 0.66667}$$

$$G_{41,28} = \frac{-0.66977s + 0.33489}{3s^2 + 2.5s + 0.5}$$

$$G_{43,9} = \frac{-0.12381s + 0.027513}{2s^2 + 1.4444s + 0.22222}$$

$$G_{43,19} = \frac{-0.69516s + 0.17379}{s^2 + 1.25s + 0.25}$$

$$G_{43,42} = \frac{-0.80427s + 0.20107}{s^2 + 1.25s + 0.25}$$

$$G_{44,42} = \frac{-0.82848s + 0.33139}{4s^2 + 2.6s + 0.4}$$

$$G_{45,15} = \frac{-0.6745s + 0.22483}{3s^2 + 2s + 0.33333}$$

$$G_{46,11} = \frac{-0.34257s + 0.11419}{2s^2 + 1.6667s + 0.33333}$$

$$G_{46,12} = \frac{-0.29935s + 0.29935}{4s^2 + 5s + 1}$$

$$G_{46,33} = \frac{-0.73455s + 0.14691}{4s^2 + 1.8s + 0.2}$$

$$G_{46,44} = \frac{-0.73146s + 0.36573}{3s^2 + 2.5s + 0.5}$$

$$G_{47,18} = \frac{-0.11772s + 0.026159}{3s^2 + 1.6667s + 0.22222}$$

$$G_{47,38} = \frac{-0.77705s + 0.38853}{5s^2 + 3.5s + 0.5}$$

$$G_{49,24} = \frac{-0.42699s + 0.2135}{s^2 + 1.5s + 0.5}$$

$$G_{49,26} = \frac{-0.40604s + 0.40604}{4s^2 + 5s + 1}$$

$$G_{49,29} = \frac{-0.68261s + 0.22754}{3s^2 + 2s + 0.33333}$$

$$G_{49,33} = \frac{-0.87596s + 0.25028}{4s^2 + 2.1429s + 0.28571}$$

$$G_{49,39} = \frac{-0.35116s + 0.14046}{5s^2 + 3s + 0.4}$$

$$G_{50,14} = \frac{-0.63786s + 0.63786}{4s^2 + 5s + 1}$$

E.2 Controller Transfer Function (G_C)

$G_C = I$ (50 × 50 identity matrix), i.e., all elements except diagonal are kept as 0.

$$
\begin{aligned}
G_{1,1} &= \frac{5.101s+0.3026}{s} & G_{18,18} &= \frac{1.504s+0.2324}{s} & G_{35,35} &= \frac{1.263s+0.6763}{s} \\
G_{2,2} &= \frac{0.094s+0.1881}{s} & G_{19,19} &= \frac{0.8204s+0.6138}{s} & G_{36,36} &= \frac{15.74s+3.531}{s} \\
G_{3,3} &= \frac{1.894s+0.5249}{s} & G_{20,20} &= \frac{0.8667s+0.1344}{s} & G_{37,37} &= \frac{22.4s+41.29}{s} \\
G_{4,4} &= \frac{0.6904s+0.134}{s} & G_{21,21} &= \frac{5.101s+0.8026}{s} & G_{38,38} &= \frac{1.504s+0.2324}{s} \\
G_{5,5} &= \frac{1.263s+0.6763}{s} & G_{22,22} &= \frac{0.094s+0.1881}{s} & G_{39,39} &= \frac{0.8204s+0.6138}{s} \\
G_{6,6} &= \frac{15.74s+3.531}{s} & G_{23,23} &= \frac{1.894s+0.5249}{s} & G_{40,40} &= \frac{0.8667s+0.1344}{s} \\
G_{7,7} &= \frac{264.4s+29.29}{s} & G_{24,24} &= \frac{0.6904s+0.134}{s} & G_{41,41} &= \frac{5.101s+0.8026}{s} \\
G_{8,8} &= \frac{1.504s+0.2324}{s} & G_{25,25} &= \frac{1.263s+0.6763}{s} & G_{42,42} &= \frac{0.094s+0.1881}{s} \\
G_{9,9} &= \frac{0.8204s+0.6138}{s} & G_{26,26} &= \frac{15.74s+3.531}{s} & G_{43,43} &= \frac{1.894s+0.5249}{s} \\
G_{10,10} &= \frac{0.8667s+0.1344}{s} & G_{27,27} &= \frac{14.4s+33.29}{s} & G_{44,44} &= \frac{0.6904s+0.134}{s} \\
G_{11,11} &= \frac{5.101s+0.8026}{s} & G_{28,28} &= \frac{1.504s+0.2324}{s} & G_{45,45} &= \frac{1.263s+0.6763}{s} \\
G_{12,12} &= \frac{0.094s+0.1881}{s} & G_{29,29} &= \frac{0.8204s+0.6138}{s} & G_{46,46} &= \frac{15.74s+3.531}{s} \\
G_{13,13} &= \frac{1.894s+0.5249}{s} & G_{30,30} &= \frac{0.8667s+0.1344}{s} & G_{47,47} &= \frac{18.94s+31.29}{s} \\
G_{14,14} &= \frac{0.6904s+0.134}{s} & G_{31,31} &= \frac{5.101s+0.8026}{s} & G_{48,48} &= \frac{1.504s+0.2324}{s} \\
G_{15,15} &= \frac{1.263s+0.6763}{s} & G_{32,32} &= \frac{0.094s+0.1881}{s} & G_{49,49} &= \frac{0.8204s+0.6138}{s} \\
G_{16,16} &= \frac{15.74s+3.531}{s} & G_{33,33} &= \frac{1.894s+0.5249}{s} & G_{50,50} &= \frac{0.8667s+0.1344}{s} \\
G_{17,17} &= \frac{26.4s+51.29}{s} & G_{34,34} &= \frac{0.6904s+0.134}{s}
\end{aligned}
$$

E.3 Sensor Noise (G_m) and Noise Power Values (η)

Measurement noise $H(s)$ is introduced using zero-mean Gaussian white noise $N(s)$ with noise power values η, with each noise signal generated using a different seed. $H(s) = N(s)$ is obtained by setting G_m to be a 50×50 identity matrix.

The noise power values η and corresponding seeds are given below:

Variable	η	**Seed**	**Variable**	η	**Seed**
1	0.019	23342	26	0.013	23441
2	0.011	23343	27	0.013	23441
3	0.015	23341	28	0.02	23444
4	0.023	23443	29	0.014	23343
5	0.018	23442	30	0.016	23342
6	0.013	23441	31	0.01	23341
7	0.02	23344	32	0.019	23342
8	0.014	23343	33	0.011	23343
9	0.016	23442	34	0.023	23443
10	0.01	23441	35	0.018	23442
11	0.019	23442	36	0.013	23441
12	0.011	23443	37	0.02	23344
13	0.015	23441	38	0.014	23343
14	0.023	23443	39	0.016	23342
15	0.018	23442	40	0.01	23341
16	0.013	23441	41	0.019	23342
17	0.02	23344	42	0.011	23343
18	0.014	23343	43	0.015	23341
19	0.016	23342	44	0.023	23443
20	0.01	23341	45	0.018	23442
21	0.019	23342	46	0.013	23441
22	0.011	23343	47	0.02	23344
23	0.015	23341	48	0.014	23343
24	0.023	23443	49	0.016	23342
25	0.018	23442	50	0.01	23341

Bibliography

[1] M. Jelali, "Control performance management in industrial automation: assessment, diagnosis and improvement of control loop performance," 2012.

[2] L. H. Chiang, E. L. Russell, and R. D. Braatz, *Fault detection and diagnosis in industrial systems*. Springer Science & Business Media, 2012.

[3] P. Vachhani, R. Rengaswamy, V. Gangwal, and S. Narasimhan, "Recursive estimation in constrained nonlinear dynamical systems," *AIChE Journal*, vol. 51, no. 3, pp. 946–959, 2005.

[4] V. Venkatasubramanian, R. Rengaswamy, K. Yin, and S. N. Kavuri, "A review of process fault detection and diagnosis: Part i: Quantitative model-based methods," *Computers & Chemical Engineering*, vol. 27, no. 3, pp. 293–311, 2003.

[5] A. Horch, "Benchmarking control loops with oscillations and stiction," in *Process control performance assessment: From theory to implementation*, pp. 227–257, Springer, 2007.

[6] L. Desborough and R. Miller, "Increasing customer value of industrial control performance monitoring-honeywell's experience," in *AIChE symposium series*, no. 326, pp. 169–189, New York; American Institute of Chemical Engineers; 1998, 2002.

[7] M. Jelali and B. Huang, "Detection and diagnosis of stiction in control loops: state of the art and advanced methods," 2009.

[8] Y.-J. Park, S.-K. S. Fan, and C.-Y. Hsu, "A review on fault detection and process diagnostics in industrial processes," *Processes*, vol. 8, no. 9, p. 1123, 2020.

[9] M. Basseville, "On-board component fault detection and isolation using the statistical local approach," *Automatica*, vol. 34, no. 11, pp. 1391–1415, 1998.

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

[11] F. Zhang and Z. Ge, "Decision fusion systems for fault detection and identification in industrial processes," *Journal of Process Control*, vol. 31, pp. 45–54, 2015.

[12] X. Zhang and K. A. Hoo, "Effective fault detection and isolation using bond graph-based domain decomposition," *Computers & chemical engineering*, vol. 35, no. 1, pp. 132–148, 2011.

[13] M. Döhler, L. Mevel, and Q. Zhang, "Fault detection, isolation and quantification from gaussian residuals with application to structural damage diagnosis," *Annual Reviews in Control*, vol. 42, pp. 244–256, 2016.

[14] W. Ku, R. H. Storer, and C. Georgakis, "Disturbance detection and isolation by dynamic principal component analysis," *Chemometrics and intelligent laboratory systems*, vol. 30, no. 1, pp. 179–196, 1995.

[15] S. A. A. Taqvi, H. Zabiri, L. D. Tufa, F. Uddin, S. A. Fatima, and A. S. Maulud, "A review on data-driven learning approaches for fault detection and diagnosis in chemical processes," *ChemBioEng Reviews*, vol. 8, no. 3, pp. 239–259, 2021.

[16] A. Das, J. Maiti, and R. Banerjee, "Process monitoring and fault detection strategies: a review," *International Journal of Quality & Reliability Management*, vol. 29, no. 7, pp. 720–752, 2012.

[17] L. Ming and J. Zhao, "Review on chemical process fault detection and diagnosis," in *2017 6th international symposium on advanced control of industrial processes (AdCONIP)*, pp. 457–462, IEEE, 2017.

[18] C. Botre, M. Mansouri, M. N. Karim, H. Nounou, and M. Nounou, "Multiscale pls-based glrt for fault detection of chemical processes," *Journal of Loss Prevention in the Process Industries*, vol. 46, pp. 143–153, 2017.

[19] S. Karra, M. Jelali, M. N. Karim, and A. Horch, "Detection of oscillating control loops," *Detection and Diagnosis of Stiction in Control Loops: State of the Art and Advanced Methods*, pp. 61–100, 2010.

[20] W. Bialkowski, "Dreams versus reality: a view from both sides of the gap: manufacturing excellence with come only through engineering excellence," *Pulp & Paper Canada*, vol. 94, no. 11, pp. 19–27, 1993.

[21] B. S. Torres, F. B. de Carvalho, M. de Oliveira Fonseca, and C. Seixas Filho, "Performance assessment of control loops–case studies," *Proc IFAC ADCHEM, Gramado, Brasil*, 2006.

[22] M. A. Paulonis and J. W. Cox, "A practical approach for large-scale controller performance assessment, diagnosis, and improvement," *Journal of Process Control*, vol. 13, no. 2, pp. 155–168, 2003.

[23] M. Choudhury, "Automatic detection and estimation of amplitudes and frequencies of multiple oscillations in process data," *ADCONIP 2014*, pp. 514–519, 2014.

[24] W. Bounoua, M. F. Aftab, and C. W. P. Omlin, "Controller performance monitoring: A survey of problems and a review of approaches from a data-driven perspective with a focus on oscillations detection and diagnosis," *Industrial & Engineering Chemistry Research*, vol. 61, no. 49, pp. 17735–17765, 2022.

[25] M. S. Choudhury, N. F. Thornhill, and S. L. Shah, "Modelling valve stiction," *Control engineering practice*, vol. 13, no. 5, pp. 641–658, 2005.

[26] R. Srinivasan, R. Rengaswamy, and R. Miller, "Control loop performance assessment. 1. a qualitative approach for stiction diagnosis," *Industrial & engineering chemistry research*, vol. 44, no. 17, pp. 6708–6718, 2005.

[27] L. Desborough, "Control system reliability-process out of control," *Industrial Control*, pp. 52–55, 2001.

[28] L. Desborough, "Control loop economics," *Honeywell proprietary report*, 2003.

[29] A. Cinar, A. Palazoglu, and F. Kayihan, *Chemical process performance evaluation*. CRC press, 2007.

[30] R. B. di Capaci and C. Scali, "Review and comparison of techniques of analysis of valve stiction: From modeling to smart diagnosis," *Chemical Engineering Research and Design*, vol. 130, pp. 230–265, 2018.

[31] N. F. Thornhill, J. W. Cox, and M. A. Paulonis, "Diagnosis of plant-wide oscillation through data-driven analysis and process understanding," *Control Engineering Practice*, vol. 11, no. 12, pp. 1481–1490, 2003.

[32] A. S. Brasio, A. Romanenko, and N. C. Fernandes, "Modeling, detection and quantification, and compensation of stiction in control loops: The state of the art," *Industrial & Engineering Chemistry Research*, vol. 53, no. 39, pp. 15020–15040, 2014.

[33] T. Hägglund, "A control-loop performance monitor," *Control Engineering Practice*, vol. 3, no. 11, pp. 1543–1551, 1995.

[34] N. F. Thornhill and T. Hägglund, "Detection and diagnosis of oscillation in control loops," *Control Engineering Practice*, vol. 5, no. 10, pp. 1343–1354, 1997.

[35] T. Miao and D. E. Seborg, "Automatic detection of excessively oscillatory feedback control loops," in *Proceedings of the 1999 IEEE International Conference on Control Applications (Cat. No. 99CH36328)*, vol. 1, pp. 359–364, IEEE, 1999.

[36] N. F. Thornhill, B. Huang, and H. Zhang, "Detection of multiple oscillations in control loops," *Journal of Process control*, vol. 13, no. 1, pp. 91–100, 2003.

[37] T. Matsuo, H. Sasaoka, and Y. Yamashita, "Detection and diagnosis of oscillations in process plants," in *Knowledge-Based Intelligent Information and Engineering Systems: 7th INternational Conference, KES 2003, Oxford, UK, September 2003. Proceedings, Part I. 7*, pp. 1258–1264, Springer, 2003.

[38] T. Matsuo, I. Tadakuma, and N. Thornhill, "Diagnosis of a unit-wide disturbance caused by saturation in a manipulated variable," 2004.

[39] T. Matsuo, "Application of wavelet transform to control system diagnosis," in *2005 The IEE Seminar on Control Loop Assessment and Diagnosis (Ref. No. 2005/11008)*, pp. 81–88, IET, 2005.

[40] W. Bounoua, M. F. Aftab, and C. W. P. Omlin, "Online detrended fluctuation analysis and improved empirical wavelet transform for real-time oscillations detection in industrial control loops," *Computers & Chemical Engineering*, vol. 172, p. 108173, 2023.

[41] T. I. Salsbury and A. Singhal, "A new approach for arma pole estimation using higher-order crossings," in *Proceedings of the 2005, American Control Conference, 2005.*, pp. 4458–4463, IEEE, 2005.

[42] R. Srinivasan, R. Rengaswamy, and R. Miller, "A modified empirical mode decomposition (emd) process for oscillation characterization in control loops," *Control Engineering Practice*, vol. 15, no. 9, pp. 1135–1148, 2007.

[43] H. Jiang, M. S. Choudhury, and S. L. Shah, "Detection and diagnosis of plant-wide oscillations from industrial data using the spectral envelope method," *Journal of Process Control*, vol. 17, no. 2, pp. 143–155, 2007.

[44] A. Zakharov and S.-L. Jämsä-Jounela, "Robust oscillation detection index and characterization of oscillating signals for valve stiction detection," *Industrial & Engineering Chemistry Research*, vol. 53, no. 14, pp. 5973–5981, 2014.

[45] J. Wang and C. Zhao, "Variants of slow feature analysis framework for automatic detection and isolation of multiple oscillations in coupled control loops," *Computers & Chemical Engineering*, vol. 141, p. 107029, 2020.

[46] S. Sharma, V. Kumar, and K. Rana, "Automatic oscillations detection and quantification in process control loops using linear predictive coding," *Engineering Science and Technology, an International Journal*, vol. 23, no. 1, pp. 123–143, 2020.

[47] Q. P. He, J. Wang, M. Pottmann, and S. J. Qin, "A curve fitting method for detecting valve stiction in oscillating control loops," *Industrial & engineering chemistry research*, vol. 46, no. 13, pp. 4549–4560, 2007.

[48] A. Singhal and T. I. Salsbury, "A simple method for detecting valve stiction in oscillating control loops," *Journal of process control*, vol. 15, no. 4, pp. 371–382, 2005.

[49] A. Horch and A. J. Isaksson, "Detection of valve stiction in integrating processes," in *2001 European Control Conference (ECC)*, pp. 1327–1332, IEEE, 2001.

[50] A. Horch, "A simple method for detection of stiction in control valves," *Control Engineering Practice*, vol. 7, no. 10, pp. 1221–1231, 1999.

[51] M. Kano, H. Maruta, H. Kugemoto, and K. Shimizu, "Practical model and detection algorithm for valve stiction," *IFAC Proceedings Volumes*, vol. 37, no. 9, pp. 859–864, 2004.

[52] H. Maruta, M. Kano, H. Kugemoto, and K. Shimizu, "Modeling and detection of stiction in pneumatic control valve," *Transactions of the Society of Instrument and Control Engineers*, vol. 40, no. 8, pp. 825–833, 2004.

[53] Y. Yamashita, "An automatic method for detection of valve stiction in process control loops," *Control Engineering Practice*, vol. 14, no. 5, pp. 503–510, 2006.

[54] A. S. Brásio, A. Romanenko, and N. C. Fernandes, "Detection of stiction in level control loops," *IFAC-PapersOnLine*, vol. 48, no. 8, pp. 421–426, 2015.

[55] J. W. Dambros, M. Farenzena, and J. O. Trierweiler, "Data-based method to diagnose valve stiction with variable reference signal," *Industrial & Engineering Chemistry Research*, vol. 55, no. 39, pp. 10316–10327, 2016.

[56] J. W. Dambros, M. Farenzena, and J. O. Trierweiler, "The effect of the sampling period on stiction detection methods," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 2848–2853, 2017.

[57] J. W. Dambros, M. Farenzena, and J. O. Trierweiler, "Stiction detection in low sampling rate signals," *The Canadian Journal of Chemical Engineering*, vol. 96, no. 8, pp. 1735–1745, 2018.

[58] O. P. Garcia, A. Zakharov, and S.-L. Jämsä-Jounela, "Data and reliability characterization strategy for automatic detection of valve stiction in control loops," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 3, pp. 769–780, 2016.

[59] N. Thornhill, S. Shah, and B. Huang, "Detection of distributed oscillations and root-cause diagnosis," *IFAC Proceedings Volumes*, vol. 34, no. 27, pp. 149–154, 2001.

[60] M. F. Aftab, M. Hovd, N. E. Huang, and S. Sivalingam, "An adaptive non-linearity detection algorithm for process control loops," *IFAC-PapersOnLine*, vol. 49, no. 7, pp. 1020–1025, 2016.

[61] M. F. Aftab, M. Hovd, and S. Sivalingam, "Detecting non-linearity induced oscillations via the dyadic filter bank property of multivariate empirical mode decomposition," *Journal of Process Control*, vol. 60, pp. 68–81, 2017.

[62] M. Bauer and N. F. Thornhill, "A practical method for identifying the propagation path of plant-wide disturbances," *Journal of process control*, vol. 18, no. 7-8, pp. 707–719, 2008.

[63] A. McDougall, D. Stoffer, and D. Tyler, "Optimal transformations and the spectral envelope for real-valued time series," *Journal of Statistical Planning and Inference*, vol. 57, no. 2, pp. 195–214, 1997.

[64] R. Srinivasan, R. Rengaswamy, S. Narasimhan, and R. Miller, "Control loop performance assessment. 2. hammerstein model approach for stiction diagnosis," *Industrial & engineering chemistry research*, vol. 44, no. 17, pp. 6719–6728, 2005.

[65] B. Srinivasan, T. Spinner, and R. Rengaswamy, "A new measure to improve the reliability of stiction detection techniques," *Industrial & Engineering Chemistry Research*, vol. 54, no. 30, pp. 7476–7488, 2015.

[66] C. Li, F. Qian, M. S. Choudhury, and W. Du, "Stiction quantification based on time and frequency domain criterions," *IFAC-PapersOnLine*, vol. 48, no. 8, pp. 635–640, 2015.

[67] Z. Yan, J. Chen, and Z. Zhang, "Valve stiction detection using the bootstrap hammerstein system identification," in *2017 6th International Symposium on Advanced Control of Industrial Processes (AdCONIP)*, pp. 84–89, Ieee, 2017.

[68] R. B. di Capaci, C. Scali, and G. Pannocchia, "System identification applied to stiction quantification in industrial control loops: A comparative study," *Journal of Process Control*, vol. 46, pp. 11–23, 2016.

[69] R. B. Capaci, C. Scali, and G. Pannocchia, "Identification techniques for stiction quantification in the presence of nonstationary disturbances," *IFAC-PapersOnLine*, vol. 48, no. 8, pp. 629–634, 2015.

[70] S. S. Jeremiah, H. Zabiri, M. Ramasamy, W. K. Teh, B. Kamaruddin, and A. A. A. M. Amiruddin, "Generic framework for valve stiction detection and compensation with anfis-activated dual-mode mpc," *Journal of Process Control*, vol. 79, pp. 85–97, 2019.

[71] S. El-Ferik, M. Al-Naser, A. Al-Amoudi, A. Al-Najim, and K. A. Sattar, "Intelligent control valve stiction diagnosis approach," *IEEE Access*, 2025.

[72] D. Zheng, X. Sun, S. K. Damarla, A. Shah, J. Amalraj, and B. Huang, "Valve stiction detection and quantification using a k-means clustering based moving window approach," *Industrial & Engineering Chemistry Research*, vol. 60, no. 6, pp. 2563–2577, 2021.

[73] S. K. Damarla, X. Sun, F. Xu, A. Shah, J. Amalraj, and B. Huang, "Practical linear regression-based method for detection and quantification of stiction in control valves," *Industrial & Engineering Chemistry Research*, vol. 61, no. 1, pp. 502–514, 2021.

[74] S. Liu and Y. Wang, "Control valve stiction detection based on the improved linear regression method," in *2024 36th Chinese Control and Decision Conference (CCDC)*, pp. 2589–2594, IEEE, 2024.

[75] M. R. Maurya, R. Rengaswamy, and V. Venkatasubramanian, "Application of signed digraphs-based analysis for fault diagnosis of chemical process flowsheets," *Engineering Applications of Artificial Intelligence*, vol. 17, no. 5, pp. 501–518, 2004.

[76] H. Jiang, R. Patwardhan, and S. L. Shah, "Root cause diagnosis of plant-wide oscillations using the concept of adjacency matrix," *Journal of Process Control*, vol. 19, no. 8, pp. 1347–1354, 2009.

[77] S. Kabir, "An overview of fault tree analysis and its application in model based dependability analysis," *Expert Systems with Applications*, vol. 77, pp. 114–135, 2017.

[78] A. Tangirala, S. Shah, and N. Thornhill, "Pscmap: A new tool for plant-wide oscillation detection," *Journal of Process Control*, vol. 15, no. 8, pp. 931–941, 2005.

[79] P. Duan, T. Chen, S. L. Shah, and F. Yang, "Methods for root cause diagnosis of plant-wide oscillations," *AIChE Journal*, vol. 60, no. 6, pp. 2019–2034, 2014.

[80] B. Srinivasan, U. Nallasivam, and R. Rengaswamy, "Diagnosis of root cause for oscillations in closed-loop chemical process systems," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 13145–13150, 2011.

[81] B. Srinivasan, U. Nallasivam, and R. Rengaswamy, "An integrated approach for oscillation diagnosis in linear closed loop systems," *Chemical Engineering Research and Design*, vol. 93, pp. 483–495, 2015.

[82] S. Xu, M. Baldea, T. F. Edgar, W. Wojsznis, T. Blevins, and M. Nixon, "Root cause diagnosis of plant-wide oscillations based on information transfer in the frequency domain," *Industrial & Engineering Chemistry Research*, vol. 55, no. 6, pp. 1623–1629, 2016.

[83] S. L. Bressler and A. K. Seth, "Wiener–granger causality: a well established methodology," *Neuroimage*, vol. 58, no. 2, pp. 323–329, 2011.

[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] 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.

[86] B. Lindner, L. Auret, and M. Bauer, "A systematic workflow for oscillation diagnosis using transfer entropy," *IEEE Transactions on Control Systems Technology*, vol. 28, no. 3, pp. 908–919, 2019.

[87] G. Weidl, A. L. Madsen, and S. Israelson, "Applications of object-oriented bayesian networks for condition monitoring, root cause analysis and decision support on operation of complex continuous processes," *Computers & Chemical Engineering*, vol. 29, no. 9, pp. 1996–2009, 2005.

[88] J. Mori, V. Mahalec, and J. Yu, "Identification of probabilistic graphical network model for root-cause diagnosis in industrial processes," *Computers & Chemical Engineering*, vol. 71, pp. 171–209, 2014.

[89] M. T. Amin, F. Khan, and S. Imtiaz, "Fault detection and pathway analysis using a dynamic bayesian network," *Chemical Engineering Science*, vol. 195, pp. 777–790, 2019.

[90] G. Chen and Z. Ge, "Hierarchical bayesian network modeling framework for large-scale process monitoring and decision making," *IEEE Transactions on Control Systems Technology*, vol. 28, no. 2, pp. 671–679, 2018.

[91] P. Kumari, D. Lee, Q. Wang, M. N. Karim, and J. Sang-Il Kwon, "Root cause analysis of key process variable deviation for rare events in the chemical process industry," *Industrial & Engineering Chemistry Research*, vol. 59, no. 23, pp. 10987–10999, 2020.

[92] P. Kumari, B. Bhadriraju, Q. Wang, and J. S.-I. Kwon, "A modified bayesian network to handle cyclic loops in root cause diagnosis of process faults in the chemical process industry," *Journal of Process Control*, vol. 110, pp. 84–98, 2022.

[93] P. Kumari, Q. Wang, F. Khan, and J. S.-I. Kwon, "A direct transfer entropy-based multiblock bayesian network for root cause diagnosis of process faults," *Industrial & Engineering Chemistry Research*, vol. 61, no. 43, pp. 16166–16178, 2022.

[94] M. Farenzena and J. O. Trierweiler, "A novel technique to estimate valve stiction based on pattern recognition," in *Computer aided chemical engineering*, vol. 27, pp. 1191–1196, Elsevier, 2009.

[95] H. Zabiri, A. Maulud, N. Omar, and M. Ramasamy, "Nn-based algorithm for control valve stiction quantification," *WSEAS Trans. Syst. Control*, vol. 4, no. 2, pp. 88–97, 2009.

[96] A. R. Venceslau, L. A. Guedes, and D. R. Silva, "Artificial neural network approach for detection and diagnosis of valve stiction," in *Proceedings of 2012 IEEE 17th International Conference on Emerging Technologies & Factory Automation (ETFA 2012)*, pp. 1–4, IEEE, 2012.

[97] J. W. Dambros, J. O. Trierweiler, M. Farenzena, and M. Kloft, "Oscillation detection in process industries by a machine learning-based approach," *Industrial & Engineering Chemistry Research*, vol. 58, no. 31, pp. 14180–14192, 2019.

[98] J. W. Dambros, M. Farenzena, and J. O. Trierweiler, "Oscillation detection and diagnosis in process industries by pattern recognition technique," *IFAC-PapersOnLine*, vol. 52, no. 1, pp. 299–304, 2019.

[99] S. Sharma, V. Kumar, and K. Rana, "Control loop oscillation detection and quantification using prony method of iir filter design and deep neural network," *Journal of Intelligent & Fuzzy Systems*, vol. 42, no. 2, pp. 1141–1154, 2022.

[100] Y. Arbabi Yazdi, H. Toossian Shandiz, and H. Gholizade Narm, "Automatic oscillations detection and classification of control loop using generalized machine learning algorithms," *Transactions of the Institute of Measurement and Control*, vol. 45, no. 3, pp. 476–491, 2023.

[101] T. Wang, Q. Chen, X. Lang, L. Xie, P. Li, and H. Su, "Detection of oscillations in process control loops from visual image space using deep convolutional networks," *IEEE/CAA Journal of Automatica Sinica*, vol. 11, no. 4, pp. 982–995, 2024.

[102] M. Daneshwar and N. M. Noh, "Detection of stiction in flow control loops based on fuzzy clustering," *Control Engineering Practice*, vol. 39, pp. 23–34, 2015.

[103] J. Miskin, B. Lindner, L. Auret, C. Dorfling, and S. Bradshaw, "Fault detection for simulated valve faults in a high pressure leaching process," *IFAC-PapersOnLine*, vol. 49, no. 7, pp. 394–399, 2016.

[104] W. K. Teh, H. Zabiri, Y. Samyudia, S. S. Jeremiah, B. Kamaruddin, A. A. Mohd Amiruddin, and N. M. Ramli, "An improved diagnostic tool for control valve stiction based on nonlinear principle component analysis," *Industrial & Engineering Chemistry Research*, vol. 57, no. 33, pp. 11350–11365, 2018.

[105] L. Shang, Y. Zhang, and H. Zhang, "Valve stiction detection method based on dynamic slow feature analysis and hurst exponent," *Processes*, vol. 11, no. 7, p. 1913, 2023.

[106] A. A. A. M. Amiruddin, H. Zabiri, S. S. Jeremiah, W. K. Teh, and B. Kamaruddin, "Valve stiction detection through improved pattern recognition using neural networks," *Control Engineering Practice*, vol. 90, pp. 63–84, 2019.

[107] S. K. Damarla and B. Huang, "Control valve stiction detection using learning vector quantization neural network," *IFAC-PapersOnLine*, vol. 58, no. 14, pp. 379–383, 2024.

[108] Y. Henry, C. Aldrich, and H. Zabiri, "Detection and severity identification of control valve stiction in industrial loops using integrated partially retrained cnn-pca frameworks," *Chemometrics and Intelligent Laboratory Systems*, vol. 206, p. 104143, 2020.

[109] Y. Henry, C. Aldrich, and H. Zabiri, "Control valve stiction detection by use of alexnet and transfer learning," in *E3S Web of Conferences*, vol. 287, p. 03012, EDP Sciences, 2021.

[110] A. Memarian, S. K. Damarla, and B. Huang, "Control valve stiction detection using markov transition field and deep convolutional neural network," *The Canadian Journal of Chemical Engineering*, vol. 101, no. 11, pp. 6114–6125, 2023.

[111] B. Kamaruddin, H. Zabiri, A. M. Amiruddin, W. Teh, M. Ramasamy, and S. Jeremiah, "A simple model-free butterfly shape-based detection (bsd) method integrated with deep learning cnn for valve stiction detection and quantification," *Journal of Process Control*, vol. 87, pp. 1–16, 2020.

[112] Y. A. Yazdi, H. T. Shandiz, and H. G. Narm, "Stiction detection in control valves using a support vector machine with a generalized statistical variable," *ISA transactions*, vol. 126, pp. 407–414, 2022.

[113] A.-q. Guan, F.-n. Xiang, Z.-y. Li, C.-r. Liu, Z.-h. Lin, Z.-j. Jin, and J.-y. Qian, "Stiction detection and recurrence analysis for control valves by phase space reconstruction method," *Advanced Engineering Informatics*, vol. 63, p. 102949, 2025.

[114] N. R. Vazquez, D. P. Fernandes, and D. H. Chen, "Control valve stiction: Experimentation, modeling, model validation and detection with convolution neural network," *International Journal of Chemical Engineering and Applications*, vol. 10, no. 6, pp. 195–199, 2019.

[115] T. Xue, C. Shang, D. Huang, and B. Huang, "Stictiongpt: Detecting valve stiction in process control loops using large vision language model," *Available at SSRN 5265092*.

[116] H. Chen, Z. Liu, C. Alippi, B. Huang, and D. Liu, "Explainable intelligent fault diagnosis for nonlinear dynamic systems: From unsupervised to supervised learning," *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[117] X. Yuan, J. Shi, and L. Gu, "A review of deep learning methods for semantic segmentation of remote sensing imagery," *Expert Systems with Applications*, vol. 169, p. 114417, 2021.

[118] Z.-Q. Zhao, P. Zheng, S.-t. Xu, and X. Wu, "Object detection with deep learning: A review," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 11, pp. 3212–3232, 2019.

[119] M. Jelali and C. Scali, "Comparative study of valve-stiction-detection methods," in *Detection and Diagnosis of Stiction in Control Loops: State of the Art and Advanced Methods*, pp. 295–358, Springer, 2010.

[120] M. Jelali and B. Huang, eds., *Detection and Diagnosis of Stiction in Control Loops: State of the Art and Advanced Methods*. Advances in Industrial Control, London: Springer London, 1 ed., 2010.

[121] A. V. Oppenheim, *Discrete-time signal processing*. Pearson Education India, 1999.

[122] L. Xie, X. Lang, J. Chen, A. Horch, and H. Su, "Time-varying oscillation detector based on improved lmd and robust lempel–ziv complexity," *Control Engineering Practice*, vol. 51, pp. 48–57, 2016.

[123] Q. Chen, J. Chen, X. Lang, L. Xie, S. Lu, and H. Su, "Detection and diagnosis of oscillations in process control by fast adaptive chirp mode decomposition," *Control Engineering Practice*, vol. 97, p. 104307, 2020.

[124] D. B. Ender, "Process control performance: Not as good as you think," *Control Engineering*, vol. 40, no. 10, pp. 180–190, 1993.

[125] Y. C. Hutabarat, A. N. Wardana, and W. Rosita, "Detection and quantification of valve stiction based on normality test and hammerstein system identification," in *AIP Conference Proceedings*, vol. 1755, AIP Publishing, 2016.

[126] R. B. di Capaci, M. Vaccari, G. Pannocchia, and C. Scali, "Identification and estimation of valve stiction by the use of a smoothed model," *IFAC-PapersOnLine*, vol. 51, no. 18, pp. 684–689, 2018.

[127] L. Fang and J. Wang, "Identification of hammerstein systems using preisach model for sticky control valves," *Industrial & Engineering Chemistry Research*, vol. 54, no. 3, pp. 1028–1040, 2015.

[128] M. Xiong and Y. Zhu, "Valve stiction model estimation in closed-loop operation," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 1188–1193, 2020.

[129] P. Aksornsri and S. Wongsa, "Valve stiction quantification using particle swarm optimisation with linear decrease inertia weight," in *2016 13th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, pp. 1–6, IEEE, 2016.

[130] S. K. Damarla, X. Sun, F. Xu, A. Shah, and B. Huang, "Statistical test-based practical methods for detection and quantification of stiction in control valves," *Industrial & Engineering Chemistry Research*, vol. 62, no. 10, pp. 4410–4421, 2023.

[131] B. R. Navada, V. Sravani, and S. K. Venkata, "Enhancing industrial valve diagnostics: Comparison of two preprocessing methods on the performance of a stiction detection method using an artificial neural network," *Applied System Innovation*, vol. 7, no. 6, p. 104, 2024.

[132] J. Tian, Z. Wang, F. Liu, X. Lan, and P. Liu, "A valve stiction quantification method based on deviation product analysis and neural network," in *2024 39th Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, pp. 1399–1405, IEEE, 2024.

[133] S. Karra and M. N. Karim, "Comprehensive methodology for detection and diagnosis of oscillatory control loops," *Control Engineering Practice*, vol. 17, no. 8, pp. 939–956, 2009.

[134] T. L. Kok, C. Aldrich, H. Zabiri, S. A. A. Taqvi, and J. Olivier, "Application of unthresholded recurrence plots and texture analysis for industrial loops with faulty valves," *Soft Computing*, vol. 26, no. 19, pp. 10477–10492, 2022.

[135] M. Rossi and C. Scali, "Automatic detection of stiction in actuators: A technique to reduce the number of uncertain cases," *IFAC Proceedings Volumes*, vol. 37, no. 9, pp. 751–756, 2004.

[136] K. Pearson, "The grammar of science," *Nature*, vol. 46, no. 1185, pp. 247–247, 1892.

[137] J. M. De La Rubia, "Rice university rule to determine the number of bins," *Open Journal of Statistics*, vol. 14, no. 1, pp. 119–49, 2024.

[138] A. Bansal, R. Suresh, and P. Saha, "Threshold-driven frequency feature integration for neural network-based oscillation detection and quantification in process industries," *Control Engineering Practice*, vol. 165, p. 106531, 2025.

[139] D. W. Scott, "On optimal and data-based histograms," *Biometrika*, vol. 66, no. 3, pp. 605–610, 1979.

[140] M. S. Choudhury, S. L. Shah, N. F. Thornhill, and D. S. Shook, "Automatic detection and quantification of stiction in control valves," *Control engineering practice*, vol. 14, no. 12, pp. 1395–1412, 2006.

[141] C. A. McNabb and S. J. Qin, "Fault diagnosis in the feedback-invariant subspace of closed-loop systems," *Industrial & Engineering Chemistry Research*, vol. 44, no. 8, pp. 2359–2368, 2005.

[142] K. Watanabe, S. Hirota, L. Hou, and D. Himmelblau, "Diagnosis of multiple simultaneous fault via hierarchical artificial neural networks," *AIChE Journal*, vol. 40, no. 5, pp. 839–848, 1994.

[143] M. Dinkar, *DKIT: A Blackboard-based, distributed, multi-expert environment for Abnormal Situation Management*. PhD thesis, Purdue University, 1996.

[144] R. Suresh, A. Sivaram, and V. Venkatasubramanian, "A hierarchical approach for causal modeling of process systems," *Computers & Chemical Engineering*, vol. 123, pp. 170–183, 2019.

[145] 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.

[146] A. Bathelt, N. L. Ricker, and M. Jelali, "Revision of the tennessee eastman process model," *IFAC-PapersOnLine*, vol. 48, no. 8, pp. 309–314, 2015.

[147] T. Larsson, K. Hestetun, E. Hovland, and S. Skogestad, "Self-optimizing control of a large-scale plant: The tennessee eastman process," *Industrial & Engineering Chemistry Research*, vol. 40, no. 22, pp. 4889–4901, 2001.

[148] N. L. Ricker, "Decentralized control of the tennessee eastman challenge process," *Journal of Process Control*, vol. 6, no. 4, pp. 205–221, 1996.

[149] J. J. Downs and E. F. Vogel, "Tennessee eastman process simulation for matlab." http://depts.washington.edu/control/LARRY/TE/download.html, 1993. Original process introduced in 1993; MATLAB implementation available online.

[150] L.-F. Deng, J.-G. Wang, J.-R. Su, Y. Yao, and J.-L. Liu, "Root cause diagnosis of plant wide oscillations using kernel granger causality," in *2020 IEEE 9th Data Driven Control and Learning Systems Conference (DDCLS)*, pp. 812–816, IEEE, 2020.

[151] C. Tian, C. Zhao, H. Fan, and Z. Zhang, "Causal network construction based on convergent cross mapping (ccm) for alarm system root cause tracing of nonlinear industrial process," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 13619–13624, 2020.

Research Publications and Conferences

Journal Publications

1. **Abhishek Bansal**, Resmi Suresh, Raghunathan Rengaswamy, and Prabirkumar Saha, "A Cross-Correlation-Based Framework for Root Cause Detection in Multivariable Control Networks," *International Journal of Dynamics and Control*, vol. 14, p. 171, 2026. https://doi.org/10.1007/s40435-026-02097-y

2. **Abhishek Bansal**, Resmi Suresh, and Prabirkumar Saha, "Threshold-driven frequency feature integration for neural network-based oscillation detection and quantification in process industries," *Control Engineering Practice*, vol. 165, p. 106531, 2025. https://doi.org/10.1016/j.conengprac.2025.106531

Conference Proceedings

1. Aswin Krishna M, Ayush Kedawat, **Abhishek Bansal**, and Resmi Suresh, "Feature Engineering for Neural Network-Based Oscillation Detection in Process Industries," *Computer Aided Chemical Engineering*, vol. 51, pp. 1153–1158, Elsevier, 2022. https://doi.org/10.1016/B978-0-323-95879-0.50193-4

Submitted Manuscripts

1. **Abhishek Bansal**, Resmi Suresh, and Prabirkumar Saha, "Heatmap-Based Temporal Analysis for Valve Stiction Detection in Industrial Control Loops," Under Revision, 2026.

Conference Presentations

1. **Abhishek Bansal**, Rigved Samant, Resmi Suresh, and Prabirkumar Saha, "A Comprehensive Web-Based Platform for Advanced Analysis of Oscillations and Root Causes in Control Systems," AIChE Spring Meeting and 21st Global Congress on Process Safety, Dallas, TX, USA, April 6–10, 2025. (Oral Presentation)
2. **Abhishek Bansal**, Prabirkumar Saha, and Resmi Suresh, "A Quantitative Analysis Using Cross-Correlation Weighted Lag for Root Cause Identification in Connected Control Loops," IIChE CHEMCON 2023, Kolkata, India, December 27–30, 2023. (Oral Presentation)
3. **Abhishek Bansal**, Navadha Mankodi, Mohd Faheem Ullah, Resmi Suresh, and Raghunathan Rengaswamy, "Root Cause Identification Using Cross-Correlation Weighted Lag in Chemical Plants," AIChE Annual Meeting 2022, Phoenix, AZ, USA, November 13–18, 2022. (Poster Presentation)
4. Aswin Krishna M, Ayush Kedawat, **Abhishek Bansal**, and Resmi Suresh, "Feature Engineering for Neural Network-Based Oscillation Detection in Process Industries," 32nd European Symposium on Computer Aided Process Engineering (ESCAPE-32), Toulouse, France, June 12–15, 2022. (Oral Presentation)