Data-driven Process Monitoring and Fault Detection with Convex Geometry

by

Qi Zeng

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

in

Control Systems

Department of Electrical and Computer Engineering

University of Albertat

Abstract

Nowadays, industrial processes are becoming highly complex and integrated due to the applications of advanced distributed control systems. As multiple production units with thousands of actuators are operating at the same time, the reliability issue of process plants naturally arises. To ensure the operational safety and maintain the product quality, process monitoring is one of the most critical and challenging topics in today's industrial control designs. In recent decades, multivariate data-driven monitoring techniques have gained a lot of attentions, due to their relatively inexpensive implementation and good performance. However, these technique based on the statistical modeling, are mostly built upon certain assumptions on the process data. Once the measurements violate these assumptions, the monitoring performance is hard to be guaranteed. To seek for a feasible solution of such potential issue, this thesis is to develop an assumption-free data-driven fault detection method which could be more applicable in the industrial practices.

To avoid data distribution fitting in process monitoring designs, we propose a new approach to model the process normal operating behavior with a geometrical enclosure, namely, a convex hull, from the normal process data. In order to achieve on-line monitoring, an appropriate detection metric with the corresponding threshold has been developed based on the property of convex hulls. We also introduced a parallel coordinates based high-dimensional visualization tool to facilitate the visual presentations of the detection results.

The performance of the proposed method is demonstrated with simulation experiments of a continuous stirred tank heater and a benchmark plant from the Tennessee Eastman challenge problem. The results have been compared with three conven-

tional data-driven techniques, including the principle components analysis, partial least square and one-class support vector machine, in terms of the detection rates and detection delays. Both case studies reveal the advantages of the proposed method in detecting randomized disturbances over the other methods.

Dedicated to my parents Qingdong Zeng and Xuhong Jin
for their love and support throughout my life.

Acknowledgements

First and foremost, I want to express my sincere gratitude to my supervisor Dr. Tongwen Chen. He allowed me the freedom to focus on the research topic that is interesting to me, while providing the support and guidance to keep me on the right direction. Working with him is my most fulfilling experience which I ever had during my oversea study life.

Another mentor that I am grateful to is Dr. Sirish L. Shah. He share a significance amount of time to discuss with me about the concept of my research and helped me develop some of the key results of my work. Without his kindness guidance, this thesis would not be possible.

I would also like to thank the team members of the Advances in Alarm Management and Design group and the Advanced Control System Lab, particularly, Ying, Shiqi, Yuzhe, Wenkai, Ning, Jiarao, Ahmad, Xiaotao, Xiaoqian and David. It has been a great experience to work with these friends.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation and background

In modern process industry, increasing demands of high product quality and operating profit continuously drive the advances of manufacturing technology. Along with the booming development and wide application of distributed control systems (DCS), nowadays process plants are highly integrated and complicated. Based on the production requirements, different operating units may be added; multiple control loops and hundreds of sensors are also implemented with little cost. Then, the safety and reliability issues of such complex system naturally arise. As reported from Abnormal Situation Management Consortium (ASM) [4], a petrochemical plant will suffer a major incident every three years, and the estimated average annually cost in the U.S. alone due to the accidents would be around $10–20 billion. In order to avoid fatal accidents and guarantee operating safety and efficiency, all industrial processes indeed should be effectively monitored at all time. When abnormal situations are detected, appropriate treatments should be executed in time. Hence, process monitoring is the most important aspect of automated control design and always draws huge attention in both industry and academia.

Nowadays most of the advance closed loop control schemes, such as PID, model predicted control (MPC) and internal model control (IMC), are able to regulate the plant and maintain steady and smooth operation. However, when unexpected process changes appear and the controllers fail to handle, abnormal situations would still

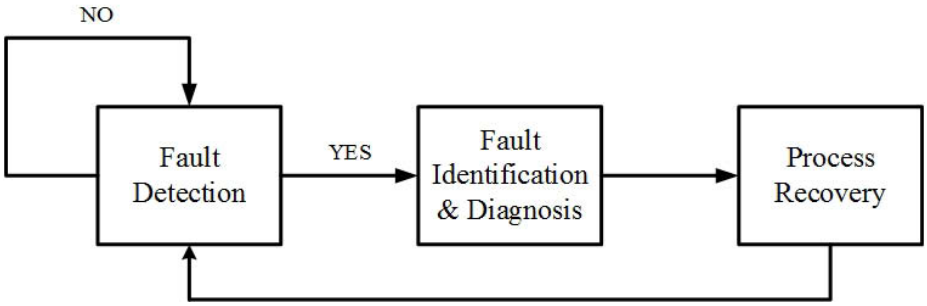


Figure 1.1: General process monitoring scheme [1]

occur [1]. In process monitoring, the fault that triggers an abnormal event is defined as an unpermitted deviation of at least one characteristic property or variable of the system [5]. Common causes of faults in an industrial system include process parameter changes, disturbance variations, and actuator or sensor problems [6]. Such large scope of malfunction sources and the increasing complexity of industrial plants make process monitoring design a great challenge.

A complete process monitoring loop commonly consists of three major stages as demonstrated in Figure 1.1. At the first stage, a reliable fault detection method should be deployed to inspect the process behavior on-line and announce the alarm if a fault is detected. Once an alarm is presented, operators should quickly move on to fault identification and diagnosis stage where they locate the malfunction source and target the root cause. Once the abnormal behavior has been clarified, effective counteractions should be taken in the final stage to bring the system back to normal operation. As the initiator of such a monitoring routine, an early and accurate fault detection would save operators invaluable time for further actions. In industrial practice, fault detection is realized in the alarm system and implemented as the primary layer of the accident protection framework [7, 8]. And thus, the performance and reliability requirements of a fault detection method are always critical.

In existing literature, fault detection methods are mainly classified into two approaches: model based and data-driven [1]. The design of model based methods is highly dependent on comprehensive process knowledge and precise system models which are hard to obtain, making the approach costly or even unrealistic to implement

in practice [9,10]. For data-driven methods, the monitoring is achieved by comparing the real time measurements with the process normal behavior captured from the operational data historian, and the design only requires a plant data repository which is relatively easier to access in today's DCS [11,12]. Due to the low cost and the ease of implementation, data driven approaches have been well recognized an accepted in industry.

Traditional data-driven fault detection methods are mainly based on a univariate approach where readings from different sensors are monitored individually by limited sensing; thus each variable's normal operating limits are generated based on their historical distributions [13]. Such simple setup have been applied in industry for a long time and is still dominating today's alarm system design [14]. However, this approach disregards the interaction among different process variables. Hence the monitoring lacks robustness and detection results might be mis-leading. Consider a quick example depicted in Figure 1.2. If we only use the mean $\pm 3\sigma$ as the high and low limits of the normal range of variables A and B, the sample represented by the red dot is then detected as a abnormal reading, and the sample represented by the green cross is regarded as a normal one. While, when we further inspect the jointed distribution of variables A and B, clearly the green cross is separated from the normal data cluster as a faulty sample, while the red dot is normal. Such ignorance of variable correlation is among the main sources of nuisance alarms, causing poor performance in many current industrial alarm monitoring systems [15].

In recent decades, multivariate fault detection methods have been getting more popular due to their advantages in monitoring the dependent and correlated of variables in large scale systems. Started at early 1990s, certain iconic methods like Principle Components Analysis (PCA) and Partial Least Square (PLS) have been leading the trend in academia [16], where many researchers began to introduce more multivariate statistical tools into process monitoring. However, similar to the univariate approach, the distribution assumptions are unavoidable when statistical modeling is involved in capturing the normal data behavior. For example, some commonly used multivariate control charts, such as Hotelling T2 and Square Prediction Error (SPE),

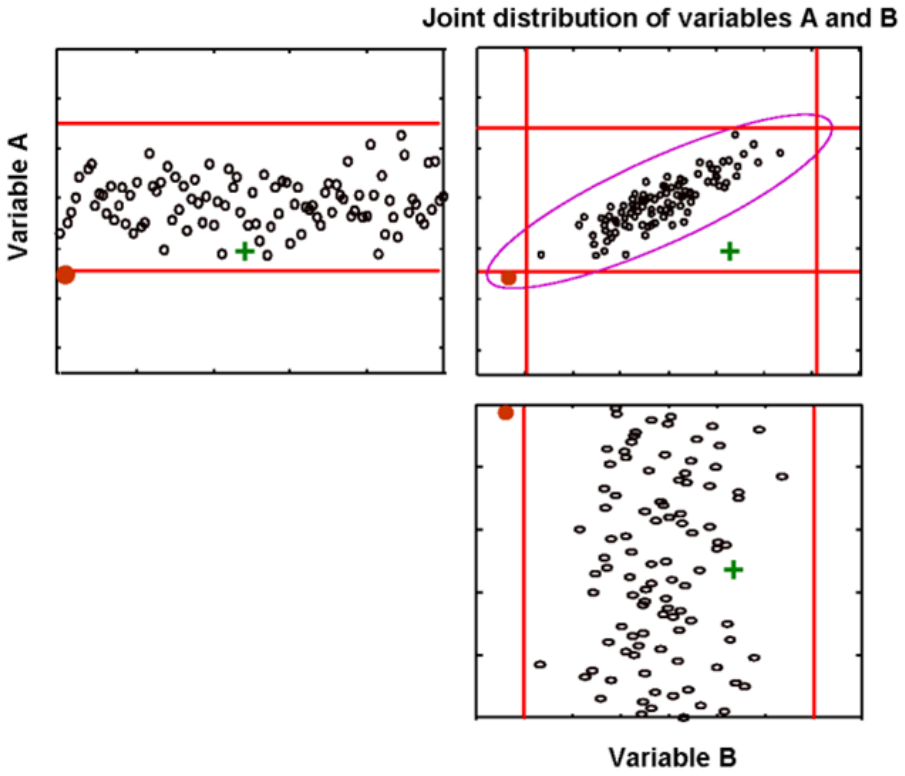


Figure 1.2: Univeriate vs. multivariate monitoring [2]

were still based on the assumption where normal process data were Gaussian or normally distributed [1]. In practical operations, uncertainty between the real process and model always exists [11], and the collected real measurements might not follow any clear distributions pattern. Once the real data violate the expected distribution, the reliability of data-driven methods might be questionable. In existing literature, this problem of assumptions mismatch has never been well addressed with a clear solution [17]. It lead to the key motivation of this thesis where we aim to develop a fault detection approach which is less restricted by assumptions and more effective for practical applications.

1.2 Literature review

Due to high demand of plant operational safety and manufacturing profitability, process monitoring and fault detection has been one of hottest research directions in

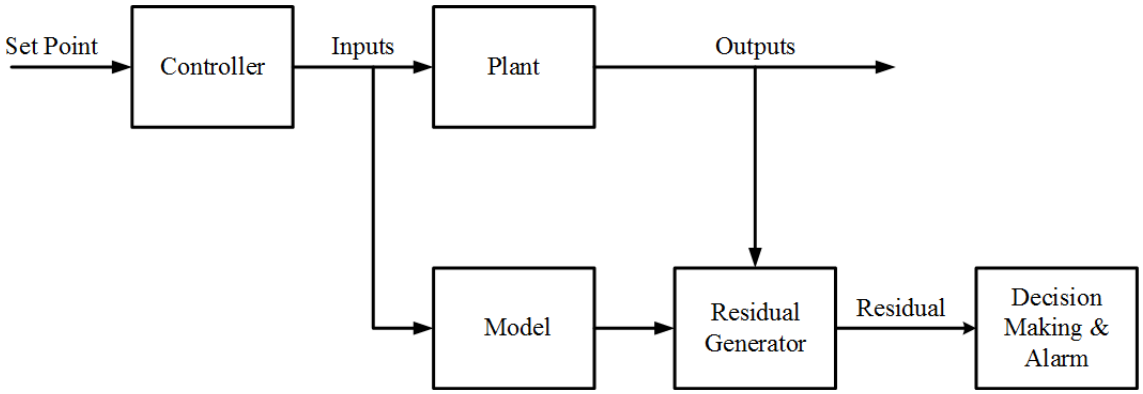


Figure 1.3: A general model-based fault detection scheme.

last decades. Intensive research has been established, and so many new methods had been developed and applied in the industry. The essential objective driving the innovation is to design a technique which offers the best fault detection performance in terms of the maximized detection rate (DR), the minimized false positive rate (FAR) and the minimal detection delay (DD). In this section, a brief discussion some existing methods is made to provide a big picture of the past and current trends in the literature.

1.2.1 Model-based approach

The so called model based fault detection originally emerged in the area of automatic control as a well-established system dynamic monitoring and accommodation problem [18]. The general concept is based on the analytical redundancy scheme depicted in Figure 1.3. By running an explicit mathematical model in parallel with the monitored plant, the inconsistencies between expected and the actual system behavior could be generated as residual measurements for fault detection and further diagnosis [19]. More intuitively, the residual should be close to zero during process normal operation, otherwise, abnormal behavior occurs when the residual value shows a significant change. The model being used could either be directly derived from the first principles or an empirically identified from the process data [9].

For residual generation, existing work could be classified into three categories, namely, observer based, parity relations based, and parameter estimation based [5].

The observer based approach was first purposed by Beard and Jones back in 1970s as failure detection filters [18, 20]. The aim was to design a set of state observers to track process operation, and then compute the detection residual as the variation between the observer outputs and the system actual outputs. With a similar concept, Willsky later adopted the Kalman filter as the optimal state estimator to monitor the process [21]. The parity relation approach was introduced by Chow in 1980s [22], where the process input-output relations was directly modeled and monitored by parity equations. Both observer and parity relations based approaches use a similar feed forward scheme to generate the residual, so they are commonly recognized as an open-loop detection structure [23]. In contrast, the parameter estimation approach used a closed loop scheme where system identification tools were first applied to estimate unmeasurable process variables, and then estimation result was fed back to calculated the detection residuals [24].

Based on the above basic approaches, a numbers of modifications and advances have been discovered to deal with the issues of model uncertainty, system nonlinearity and robustness, etc. [23, 25, 26]. Certain applications in industrial process and automatic control systems also showed their promising results in detecting sensor faults and abrupt system changes [27–29]. However, the detection performance of model based methods are highly dependent on the reliability of adopted system models. For the highly integrated and complicated industrial processes, developing comprehensive system models are very costly and sometimes unrealistic. Therefore, applying model based fault detection to a large scale system is still a practical challenge.

1.2.2 Data-driven approach

Instead of running an explicit model for analytical redundancy, a data-driven approach turns to extract the knowledge of system normal operation behavior from the data historian [30]. By capturing the variable interaction and normal sample distribution with a statistical model, the boundary of the system normal operating regime could be identified. During online monitoring, the detection result is generated by comparing the new measurements with the normal operating regime.

The history of data-driven approach could be traced back to 1920s, when the Shewhart control chart was introduced as a univariate monitoring method [31]. The detection design simply took the mean and standard deviation to capture the sensor reading distribution and thus defined the normal operating limits. Some advanced control charts like the cumulative sum (CUSUM) chart and the exponentially weighted moving average (EWMA) chart were developed later [32,33]. These control charts filter the monitored sensor data with certain linear or nonlinear filters, and use the filtered measurements to compare with detection limits [31]. Till today, the filtering techniques are still popular in the field of industrial univariate alarm design [34]. For more of advances in univariate alarm monitoring, readers are strongly recommend to the comprehensive overview provided in [35] and the references there in.

Since the univariate methods have difficulty to handle highly correlated process variables, the main trend of process monitoring and fault detection research have been shifted to multivariate approach. The topic of multivariate statistical process control (MSPC) started to arise in early 1990s with the applications of statistical tools in process monitoring, such as the monitoring scheme based on principle components analysis (PCA) and partial least square (PLS) [36,37]. PCA as a linear order reduction technique has the nature advantage in extracting correlations of high dimensional data matrix and projecting the information into a lower dimensional space [38]. In contrast, PLS as a linear regression method has the capability of identifying the correlation between inputs and outputs presented by the process data [39]. These two methods quickly got success in their applications of monitoring certain industrial systems, and their modifications had also been heavily studied to handle process nonlinearity and dynamic correlations [40–43]. Similar to PCA, the other order reduction technique, the so called canonical variate analysis (CVA), was also introduced into the scope of fault detection by Russell in 2000 [44]. By assuming the process data follow an autoregressive-moving-average model, CVA is able to capture the dynamic correlation in the normal data and improve the detection performance.

After 2000s, a movement of seeking new multivariate analysis tools from other fields also began to appear. From speech signal processing, the independent compo-

nent analysis (ICA) was first adapted to fault detection by Kano in 2004 [45–47]. Different from previous approaches, ICA has the specialty to capture the non-Gaussian behavior in process data. From the field of machine learning, the binary classifier approach based on the support vector machine (SVM) was successfully modified and adapted for fault detection by Mahadevan and Shah [48]. Combined with a radial based kernel function, the modified 1-class SVM can well monitor nonlinear processes. Most recently, the locality preserving projection (LLP) method is also applied by Hu [49]. As an optimum data neighboring structure persevering algorithm, LLP has proved to be less sensitive to the effect of outliers [50].

Among all data-driven methods that we mentioned, their statistical models are always built upon different assumptions of the normal process behavior, and each model has a unique interpretation of variables correlation [30, 50]. In existing reviews and comparison studies, their monitoring performance is expected to be different even in the same benchmark test [51]. One feasible way to accommodate this situation is through the deployment of an ensemble monitoring scheme which consists of a group of methods with different assumptions. A good example is the decision fusion framework purposed by Zhang in [52], where monitoring results from 6 different data-driven techniques are combined using Dempster-Shafer evidence theory [52]. In testing with the well know yet challenging Tennessee Eastman problem, Zhang's fault detection and diagnosis framework demonstrated a superior performance over any individual method being considered.

1.2.3 Convex hull based anomaly detection

A convex hull is a well studied geometric structure that represents the smallest convex set containing a finite set of data points in the Euclidean space. How to efficiently identify and compute the convex hull of a given dataset is one of the fundamental algorithmic problems in the scope of computational geometry [53]. The resulting algorithms have a great impact in image processing, especially in image feature extraction and pattern reorganization problems [54]. Due to the nice property of the convex hull, former researchers also applied it in statistical analysis and robust estimation, par-

ticularly for outlier detection [55]. The well knew ISODEPTH algorithm proposed by Ruts and Rousseeuw is based on convex hull peeling, where a data cluster is first organized by layers of convex hulls, and samples on the outer most hull with the least depth are identified as outliers [56]. The main advantage of ISODEPTH is that it directly captures data structures with convex hulls and does not require distribution fitting [57]. In other words, it is free of assumptions on data distribution.

Recently, the idea of using convex hulls to identify normal data structures had been further extended in anomaly detection. The essential concept is to construct a multivariate convex hull of training data as the boundary of system normal operating regime, and use the convex hull to isolate any potential faulty samples [58]. The first attempt of this detection scheme was reported by Lou in 2001, when he combined a convex hull algorithm with trend analysis to develop an aircraft monitoring method for B737 [59]; the simulated results showed good performance in detecting different instrument faults. Lately, Sthele also introduced a computational geometry based software monitoring and network security architecture called Aniketos [60]. Again, by using normal data convex hull as the detection metric, Aniketos was able to identify abnormal events of a website caused by programming errors and security attacks. In robotics, the similar approach had also been applied to monitor the functionality of a humanoid robot by Lynch and his colleague [61]. According the reports of above applications, the feasibility of using the convex hull as assumption free multivariate detection metric seems promising. But, in existing literature, an application of this approach in industrial monitoring system is still missing.

1.3 Thesis contribution

The main objective of this thesis is to develop a multivariate data-driven fault detection technique which has less restrictive data distribution assumptions and thus is more suitable for practical applications. In the existing literature, the convex hull based anomaly detection was evidenced to be an assumption-free approach with certain promising results. Therefore, the focus of this thesis is to adapt this new idea

to industrial process monitoring and fault detection. The major contributions of this thesis are listed below:

1. Introduced a new data-driven fault detection technique with the aid of the convex hull based detection metric.

2. Developed a high dimensional process data visualization tool with parallel coordinates to facilitate the convex hull based fault detection. The resulting visual monitoring presentation is able to provide users more intuitive information for further fault diagnosis and identification.

3. Demonstrated the superior detection performance of proposed method with two simulated case studies in comparison with some of popular data-driven techniques including PCA, PLS, and 1-class SVM.

1.4 Thesis organization

This thesis has been prepared according to the guidelines from the Faculty of Graduate Studies and Research (FGSR) at the University of Alberta. The rest of this thesis is organized as follows.

In Chapter 2, we discuss the fault detection techniques being considered in this research. First, to better introduce the preliminaries and concepts of data-driven fault detection, a more detailed review of the three classical methods, including PCA, PLS, and 1-class SVM, is provided, and these methods are all included in our later case studies. Next, we present the details of newly proposed convex hull based fault detection approach. In the design, the Quickhull algorithm is adapted to construct the convex hull of training process data representing the system normal operating regime; and hyperplane equations of the convex hull are used as the detection metric. Chapter 2 is concluded with a simple illustrative example where we make a brief comparison of all four fault detection methods.

Chapter 3 is concerned with the visual presentation of the real time monitoring. To overcome the dimensional limitation of scatter plots in the Cartesian space, we

present a high dimensional process visualization based on parallel coordinates. First, basic background and preliminaries of the duality between the Cartesian space and parallel coordinates is provided. To better accommodate our proposed fault detection approach, we later present the way of reconstruct a convex hull envelope as the normal operating regime in parallel coordinates with a simple graphical algorithm. Chapter 3 ends again with an example where we demonstrate how faulty samples could be easily detected with our visualization setup.

Chapter 4 is to present two simulated industrial case studies which include the Continuous Stirred Tank Heater (CSTH) and the Tennessee Eastman Challenged Problem (TEP). For both case studies, details of experimental design and implementation are provided. The result from CSTH validates the superior detection performance of the newly proposed method. Whereas, the TEP bench mark study shows the effectiveness of our fault detection method in monitoring large scale processes, and an example is given to demonstrate how the parallel coordinate visualization could be utilized for further fault diagnosis.

Chapter 5 summarizes this research again with some concluding remarks, followed by discussion of potential improvements and future work.

Chapter 2

Data-driven Fault Detection Methods

2.1 Overview

In this chapter, we investigate multivariate process monitoring through the data-driven fault detection techniques. First, a review of three iconic methods, including PCA, PLS and 1-Class SVM, is provided to introduce the common concepts of data-drive methods. Next, we propose a new fault detection approach by using multivariate convex hulls as the detection metric. A simple illustrative example is then presented to demonstrate the advantage of our method over the three existing ones.

2.2 Review of basic data-driven techniques

Multivariate data-driven fault detection methods started to become more popular ever since the topic of multivariate statistical process control (MSPC) was introduced in early 1990s. Unlike the model-based monitoring scheme, in which the design is based on an explicit system model, data-driven methods are able to trace process normal behavior from a set of historical data [37,62]. When comparing with the conventional limit sensing based univariate approaches, MSPC also has the advantage in capturing the multivariate dependency and correlation among different process variables. Due to their relatively good operational simplicity and monitoring reliability, multivariate data-driven methods have been widely applied in industrial monitoring systems with

some great successes [12].

Generally, most of the data-driven techniques consist of two preliminary design stages, namely, an off-line training stage and an on-line monitoring stage. For the off-line training stage, the associated tasks are mainly two folded [48]:

- Developing a statistical model based on a given set of normal operating data;
- Proposing a set of detection metrics and setting appropriate thresholds based on the predefined confidence level.

Once the off-line training is complete with the appropriate validations, the implementation moves on to the on-line monitoring stage, where new test samples would be projected simultaneously onto the model to calculate the detection metrics. By comparing the calculated metrics with the designed thresholds, users are able to inspect the results and make the judgment if any fault has occurred.

In this review section, we briefly introduce three iconic data-driven methods, including Principle Component Analysis (PCA), Partial Least Square (PLS) and One-class Support Vector Machine (1-class SVM), in the form of the above design concepts.

2.2.1 Principle components analysis

Principle Component Analysis (PCA) is one of the most successful multivariate analysis tools which is famous of its powerful features, such as data decorrelation and compression. As an optimal linear order reduction technique, PCA is able to project the original high-dimensional data into a lower-dimensional space with a set of orthogonal loading vectors, named as principle components (PCs), without losing the majority of data information. Due to its superiority in handling large datasets, since 1980s, PCA have been developed as one of the first MSPC methods and widely applied in industrial processes for monitoring and fault detection. As a classical technique, reviews and textbooks often consider it as the paradigm of data-driven methods [1, 12, 17, 30].

In a PCA based monitoring scheme, the off-line training is conducted with data collected from process measurements under the system normal operating condition.

Consider a process with d measured variables, where n normal samples are collected to form a standardized (mean centered and scaled to the unit variance) training data matrix $\mathbf{X}$ as:

$$\mathbf{X} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1d} \\ x_{21} & x_{22} & \dots & x_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \dots & x_{nd} \end{bmatrix} \in \mathbb{R}^{n \times d} \tag{2.1}$$

where the $x_i \in \mathbb{R}^{1 \times d}$ denotes each data sample as a row vector. The first step to built a PCA model is to perform the singular value decomposition (SVD) on the sample covariance matrix $\mathbf{M}$, i.e.

$$\mathbf{M} = \frac{1}{n-1}\mathbf{X}^\top \mathbf{X} = \mathbf{P}\mathbf{\Lambda}\mathbf{P}^\top \tag{2.2}$$

where $\mathbf{\Lambda} = \text{diag}(\lambda_1, \lambda_2, \cdots, \lambda_d) \in \mathbb{R}^{d \times d}$ contains all non-negative real eigenvalues in a decreasing order ($\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$), and $\mathbf{P} = [p_1, \cdots, p_d] \in \mathbb{R}^{d \times d}$ consists of all orthogonal PCs as the column vectors [12].

In order to decompose $\mathbf{X}$ into a lower-dimensional vector space, we further divide $\mathbf{\Lambda}$ and $\mathbf{P}$ as:

$$\mathbf{\Lambda} = \begin{bmatrix} \mathbf{\Lambda}_{pc} & 0 \\ 0 & \mathbf{\Lambda}_{res} \end{bmatrix}, \ \mathbf{\Lambda}_{pc} = \text{diag}(\lambda_1, \cdots, \lambda_a), \ \mathbf{\Lambda}_{res} = \text{diag}(\lambda_{a+1}, \cdots, \lambda_d) \tag{2.3}$$

$$\mathbf{P} = \begin{bmatrix} \mathbf{P}_{pc} & \mathbf{P}_{res} \end{bmatrix}, \ \mathbf{P}_{pc} \in \mathbb{R}^{d \times a}, \ \mathbf{P}_{res} \in \mathbb{R}^{d \times (d-a)}. \tag{2.4}$$

where $\mathbf{\Lambda}_{pc}$ consists of all relatively large eigenvalues of $\mathbf{M}$ indicating most of the data variance has been stored. The number a indicates how many eigenvalues and the corresponding PCs are obtained by the model. Then, the projection of $\mathbf{X}$ in into the lower dimensional space is denoted as a score matrix $\mathbf{T}$ which is calculated as:

$$\mathbf{T} = \mathbf{X}\mathbf{P}_{pc} \in \mathbb{R}^{n \times a} \tag{2.5}$$

When reconstructing the data and checking the modeling error, we simply calculate:

$$\hat{\mathbf{X}} = \mathbf{T}\mathbf{P}_{pc}^\top, \ \mathbf{E}_X = \mathbf{X} - \hat{\mathbf{X}} \tag{2.6}$$

where $\hat{\mathbf{X}}$ is the observations captured by the model, and $\mathbf{E}_X$ is the modeling residual.

To detect any abnormal deviations in the new process data, PCA utilizes two multivariate detection metrics, the Hotelling's T-squared statistics (T^2) and the squared prediction error (SPE), to monitor both of the modeled observations and the modeling residuals [1].

For a new collected and normalized data sample vector $\mathbf{x} \in \mathbb{R}^{1\times d}$, its deviation relative to the PCA model is represented by T^2 which is calculated as

$$T^2 = \mathbf{x}\mathbf{P}_{pc}\mathbf{\Lambda}_{pc}^{-1}\mathbf{P}_{pc}^{\top}\mathbf{x}^{\top}. \tag{2.7}$$

The corresponding detection threshold, with a confidence level α, is estimated as

$$J_{th,T_\alpha^2} = \frac{a(n^2-1)}{n(n-a)}F_\alpha(a, n-a), \tag{2.8}$$

where $F_\alpha(a, n-a)$ is the CDF value of a F distribution with a and $n-a$ degrees of freedom.

When regarding the deviation relative to modeling residuals, SPE statistics is calculated as

$$SPE = \mathbf{x}\mathbf{P}_{res}\mathbf{P}_{res}^{\top}\mathbf{x}^{\top}. \tag{2.9}$$

The SPE threshold is approximated by

$$J_{th,SPE} = \theta_1\left[\frac{h_0 c_\alpha\sqrt{2\theta_2}}{\theta_1} + 1 + \frac{\theta_2 h_0(h_0-1)}{\theta_1^2}\right]^{1/h_0}, \tag{2.10}$$

where $\theta_i = \sum_{j=a+1}^{d}\lambda_j^i$, $h_0 = 1 - \frac{2\theta_1\theta_3}{3\theta_2^2}$ and c_α is the standard deviation according to the $1-\alpha$ percentage of a normal distribution.

For the on-line monitoring stage, both detection metrics are calculated with new process data and compared with their thresholds. Detection result of each sample is then generated as a binary alarm signal based on the detection logic given by:

$$\begin{cases}\text{Normal}, & T^2 \le J_{th,T_\alpha^2} \text{ and } SPE \le J_{th,SPE}\\ \text{Faulty}, & \text{otherwise}\end{cases} \tag{2.11}$$

Notice that PCA is primarily developed based on the assumption where normal process data are linearly correlated and following the joint Gaussian distributions [63].

Hence, when training samples violate this assumption, the reliability and robustness of the resulted PCA model could not be guaranteed, and the detection performance may unfortunately be downgraded.

2.2.2 Partial least squares

Partial least square (PLS), also known as projection to latent structures, is another classical data-driven monitoring method from MSPC. Similar to PCA, PLS is also a dimension reduction technique, but more advanced in its ability to extract correlation between system inputs and outputs [36]. As a regression based approach, PLS is capable of constructing a linear regressive model to capture the process normal behavior. Since its inferential model can be utilized for the on-line prediction of product quality, PLS is also a popular tool for basic soft sensor design [64].

To build a PLS model for the fault detection purpose, firstly we prepare a set of historical data including n samples, in both of a process measurements matrix $\mathbf{X}$ and a output matrix $\mathbf{Y}$, i.e.

$$\mathbf{X} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^{n \times d}, \mathbf{Y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \in \mathbb{R}^{n \times m} \tag{2.12}$$

where $x_i \in \mathbb{R}^{1 \times d}$ denotes the data sample as a row vector collected from d measurement variables; and $y_i \in \mathbb{R}^{1 \times m}$ denotes the output vector consisted of m variables. In order to perform the regression, $m \leq d$ is required.

The linear regressive model provided by PLS is presented in the following formulas as,

$$\mathbf{X} = \mathbf{T}\mathbf{P}^\top + \mathbf{E}_X \text{ and} \tag{2.13}$$

$$\mathbf{Y} = \mathbf{T}\mathbf{Q}^\top + \mathbf{E}_Y = \mathbf{X}\mathbf{R}\mathbf{Q}^\top + \mathbf{E}_Y. \tag{2.14}$$

$T = [t_1, \cdots, t_\beta] \in \mathbb{R}^{n \times \beta}$ is a lower dimensional score matrix consisting of all orthogonal latent variables (LVs) t_i's. Each t_i stores part of the covariance between $\mathbf{X}$ and $\mathbf{Y}$.

$\mathbf{P} \in \mathbb{R}^{n\times d}$ and $\mathbf{Q} \in \mathbb{R}^{n\times m}$ are matrices containing the loading vectors of both $\mathbf{X}$ and $\mathbf{Y}$, and $\mathbf{R} \in \mathbb{R}^{d\times\beta}$ projects $\mathbf{X}$ to $\mathbf{T}$. Finally, $\mathbf{E}_X$ and $\mathbf{E}_Y$ store the model residuals of both inputs and outputs, which are assumed to be uncorrelated to neither $\mathbf{X}$ nor $\mathbf{Y}$.

To find this PLS model, several well developed algorithms are available in the existing literature, and reviews could be found in [65–67]. In this work, we utilize one of the fastest PLS algorithm, known as SIMPLS [68]. The major iterative calculations are summarized in Algorithm 1.

Algorithm 1 SIMPLS

Initialize:
  Compute $\mathbf{S} = \mathbf{X}^\top\mathbf{Y}$
for $i := 1 : \beta$ **do**
  Compute SVD of $\mathbf{S}$
  Get $\mathbf{r_i}$ as the first left singular vector
  Compute $\mathbf{X}$ score vector as $t_i = \mathbf{X}r_i$
  Compute $\mathbf{X}$ loading vector as $p_i = \mathbf{X}^\top t_i/(t_i^\top t_i)$
  Compute $\mathbf{Y}$ loading vector as $q_i = \mathbf{Y}^\top t_i$
  Compute $\mathbf{Y}$ score vector as $u_i = \mathbf{Y}q_i$
  Update $v_i = p_i$
  if $i > 1$ **then**
    Remove previous loadings and scores in v_i and u_i as
    $v_i = v_i - \mathbf{V}(\mathbf{V}^\top p_i)$
    $u_i = u_i - \mathbf{T}(\mathbf{T}^\top p_i)$
  end if
  Deflate $\mathbf{S}$ with current loading as $\mathbf{S} = \mathbf{S} - v_i(v_i^\top\mathbf{S})$
  Store t_i, r_i, p_i, q_i, v_i and u_i into $\mathbf{T}$, $\mathbf{R}$, $\mathbf{P}$, $\mathbf{Q}$, $\mathbf{V}$ and $\mathbf{U}$, respectively.
end for

Notices that β is a key design parameter in such PLS models. It presents the dimension of projected structure and indicates how many layers of LVs (in $\mathbf{T}, \mathbf{P}$ and $\mathbf{Q}$) are built in the model. A standard way to determine β is to apply a cross validation to collect the prediction residual sum of squares (PRESS) as referenced in [1]. Once the cumulated residual error of Y is stabilized below 5-10%, the PLS model could be finalized with the corresponding number of LVs as β.

Similar to the PCA based fault detection, PLS also utilizes T^2 and SPE as the two detection metrics, where the calculation formulas are nearly the same but using

different loading matrices. The exact equations are as

$$T^2 = \mathbf{x}\mathbf{R}\left(\frac{\mathbf{T}^\top\mathbf{T}}{n-1}\right)^{-1}\mathbf{R}^\top\mathbf{x}^\top \tag{2.15}$$

$$SPE = ||(\mathbf{I} - \mathbf{P}\mathbf{R}^\top)\mathbf{x}^\top||^2. \tag{2.16}$$

And the corresponding thresholds according to confidence level α are derived as

$$J_{th,T^2_\alpha} = \frac{\beta(n^2-1)}{n(n-\beta)}F_\alpha(\beta, n-\beta) \text{ and} \tag{2.17}$$

$$J_{th,SPE} = \frac{\sigma_{spe}}{2\mu_{spe}}\chi^2_\alpha(\frac{2\mu^2_{spe}}{\sigma_{spe}}), \tag{2.18}$$

where the F distribution alters its degree of freedom based on β, and χ^2_α is the CDF value of a Chi-squared distribution with its scaling factors calculated by the mean (μ_{spe}) and variance (σ_{spe}) of SPE statistics.

Again, for on-line monitoring, PLS has the same detection scheme as PCA, where the comparison results of both T^2 and SPE metrics are considered in the same detection logic:

$$\begin{cases} \text{Normal}, & T^2 \leq J_{th,T^2_\alpha} \text{ and } SPE \leq J_{th,SPE} \\ \text{Faulty}, & \text{otherwise.} \end{cases} \tag{2.19}$$

Since PLS is also a linear method, it shares the same fundamental assumption as PCA, meaning it is not designed to handle system nonlinearity and non-Gaussian distributions.

2.2.3 One-class support vector machine

The so-called support vector machine (SVM) is a famous learning algorithm which was originally designed for binary classification problems [69]. The main concept is to construct a linear classifier that best separates data points from two different classes. In order to use it for novelty detection, one-class SVM was later proposed by Schölkopf as a variant of the original algorithm [70].

In one-class SVM, the aim is to identify a supported hyperplane which best accommodate the majority of the training samples in a small space volume. If a new sample is located on the same side of the hyperplane with the training samples, it would be considered as a normal one. Otherwise, it is an outlier or a fault which does not belong to the normal class. To handle the non-linear correlations of the measurements, one-class SVM projects the training data into a higher dimensional feature space with a radial based kernel function (rbf) to remove the non-linear correlations. Thus, the linear hyperplane function derived in the feature space is equivalent to a non-linear classifier [48].

To get more detail about the modeling process of one-class SVM, again let us consider a training data matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ collected from the monitored process, where n and d indicates the number of samples and the number of variables, respectively. As mentioned earlier, we first project all samples into the feature space with rbf kernel as

$$K(x_i, x_j) = \phi(x_i) \cdot \phi(x_j) = e^{-\gamma ||x_i - x_j||^2} \tag{2.20}$$

where $\phi(x_i)$ for $i = 1, \cdots, n$ is the mapping of x_i in the feature space, and γ is a user defined variance scaling factor.

Next, to find a supported hyperplane which is above all training samples in the feature space, we define a linear classifier function as:

$$f_c(x_i) = w \cdot \phi(x_i) + \xi_i + b \geq 0, \forall\ x_i \in \mathbf{X}; \tag{2.21}$$

where w is the weight vector controlling the shape of the hyperplane, and b is the center offset of the training data cluster. For each sample x_i, ξ_i is introduced as a slack variable to relax the hard decision boundary.

Since our objective is to identify the tightest hyperplane function that covers all training samples, we solve the classifier based on the constrained optimization

problem written as follows:

$$\begin{aligned} \min_{w,\xi_i,b} \quad & \frac{1}{2}||w||^2 + \frac{1}{\alpha n}\sum_{i=1}^{n}\xi_i + b \\ \text{subject to} \quad & w \cdot \phi(x_i) \geq -b - \xi_i,\ \xi_i \geq 0. \end{aligned} \tag{2.22}$$

Notice that $\alpha \in (0, 1]$ is a design parameter to control the tradeoff between the volume of $||w||^2$ and mean of ξ_i. Intuitively, it represents the maximum fraction of error in the training samples and serves as significant level of the classifier.

According to the above constraint, $\xi_i \geq 0$ and $\rho_i \geq 0$ are defined as Lagrange multipliers to form the Lagrangian function:

$$L(w, \xi, \rho) = \frac{1}{2}||w||^2 + \frac{1}{\alpha n}\sum_{i=1}^{n}\xi_i + b - \sum_{i}^{n}\eta_i(w \cdot \phi(x_i) + b + \xi_i) - \sum_{i=1}^{n}\rho_i\xi_i. \tag{2.23}$$

To solve it, all the partial derivatives of primal variables are set to zero to have:

$$\frac{\partial L}{\partial w} = 0 \ \rightarrow \ w = \sum_{i=1}^{n}\eta_i\phi(x_i); \tag{2.24}$$

$$\frac{\partial L}{\partial \xi_i} = 0 \ \rightarrow \ \eta_i = \frac{1}{\alpha n} - \rho_i \leq \frac{1}{\alpha n}; \tag{2.25}$$

$$\frac{\partial L}{\partial b} = 0 \ \rightarrow \ \sum_{i=1}^{n}\eta_i = \alpha n. \tag{2.26}$$

Substitute Equations (2.24) to (2.26) back to Equation (2.23), a simplified Lagrangian in the dual form is finalized as:

$$\begin{aligned} \min_{\eta} \quad & \frac{1}{2}\sum_{i=1,j=1}^{n}\eta_i\eta_j K(x_i, x_j) \\ \text{subject to} \quad & o \leq \eta_i \leq \frac{1}{\alpha n},\ \sum_{i=1}^{n}\eta_i = 1. \end{aligned} \tag{2.27}$$

where η_i could be solved by using standard quadratic programming (QP) techniques [71]. The solution is further utilized to derive w and b; and thus where we complete the classifier modeling process.

For fault detection, the classifier function in the feature space is adapted as the detection metric. For a testing sample $\mathbf{x}$, we first project it in the feature space and then calculate the function value $f_c(\mathbf{x})$ as

$$f_c(\mathbf{x}) = w \cdot \phi(\mathbf{x}) + b = \sum_{i=1}^{n} \eta_i K(\mathbf{x}, x_i) + b. \tag{2.28}$$

Based on the sign of $f_c(\mathbf{x})$ we could check if this sample belongs to the same class as the training data, i.e., $f_c(\mathbf{x}) \geq 0$ indicating it is normal or vise versa. To follow the convention where a fault arises when the measurement is over the threshold, a detection logic is finalized as

$$\begin{cases} \text{Normal}, & -f_c(\mathbf{x}) \leq 0; \\ \text{Faulty}, & -f_c(\mathbf{x}) > 0. \end{cases} \tag{2.29}$$

2.3 Convex hull based fault detection

As previously discussed, data distribution assumptions are among the unavoidable factors in data-driven techniques. The design of modeling and detection metrics in most methods are targeting on some particular distribution patterns which the training samples are expected to follow. However, in industrial applications, the real data collected from various sensors may not comply with a clear distribution pattern, and thus the performance of data-driven monitoring may be downgraded [35]. To address such an issue, in this thesis we try to seek a feasible solution through the computational geometry where we adapt the idea of using convex hulls of process data as assumption free detection metrics. The following section is to introduce this new convex hull based fault detection (CH-FD) technique.

2.3.1 Basic design concept

Like most data-driven monitoring methods, CH-FD is to monitor sensor readings from process variables and identify any possible faulty sample from the normal ones. We consider the case where a process is operating in a normal steady state, where the system behavior is relatively unchanged. The essential concept is to generate a

comprehensive geometric enclosure of the collected normal process data with a convex hull which we regard as the normal operating regime. If a test sample falls into such an enclosure, it would be classified as a normal sample. Otherwise, if an exterior point of the convex hull is detected, it would be treated as a faulty reading. Figure 2.1 presents an intuitive graphical illustration of this concept.

Although the idea of using a geometric structure as the empirical enclosure of historical data is very simple, it will be shown that the resulting convex hull in the data space would be the tightest envelope of normal operating regime. The key advantage of this approach is that any strong assumptions of data correlation or pre-defined data distributions are not necessary [60].

To follow the common scheme of data-driven techniques, again the design of CH-FD is summarized into two main stages:

- Training stage: collecting a set of historical process data under system's normal operation, and computing the convex hull of the dataset as the process normal operating regime.
- Detection stage: using the convex hull as a geometric detection metric to identify any exterior test sample as a potential fault.

2.3.2 Constructing the convex hull of normal process data

From the above design concept, the key task of the CH-FD training stage is to capture the process normal operating regime with an accurately computed convex hull from a set of normal historical data. For this particular problem, a number of well known convex hull algorithms are available from computational geometry [53]. In this work, the Quickhull (Qhull) algorithm is chosen as our primary tool for convex hulls constructions. Qhull was developed by Barber and Dobkin in 1993 as a extension of the optimum 3D Clarkson and Shor's algorithm for high-dimensional convex hulls computations [72]. It is famous of its capability in handling the multi-dimensional data (up to 9 dimensions) and the imprecision errors [73].

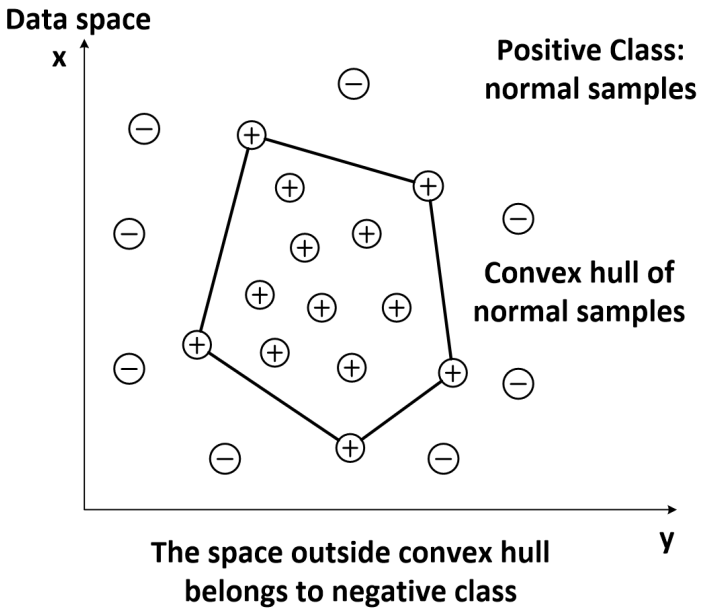


Figure 2.1: Concept of convex hull based fault detection.

Preliminary of convex hull

Before we tape into Qhull, let's first introduce some basic mathematical definitions of convex hulls and the associated preliminaries. This brief review is adapted from Barber's original work in [73].

Consider a set of linearly independent points $\{x_1, \cdots, x_n\}$, its affine combination is the set

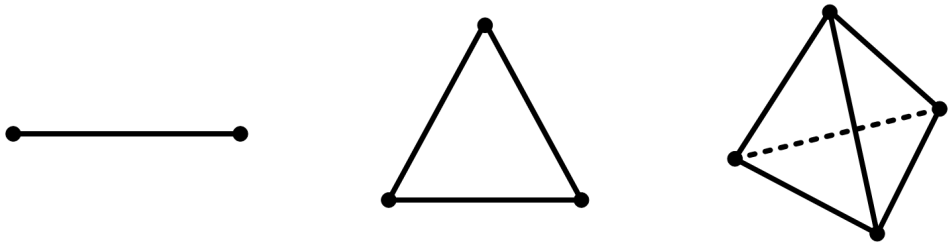
$$\{x | x = \sum_{i=1}^{n} \lambda_i x_i, \sum_{i=1}^{n} \lambda_i = 1\}. \tag{2.30}$$

When $\forall\ \lambda_i > 0$, the affine combination becomes a convex combination. Then a convex set is the set of points which contains its convex combination, and a convex hull is the smallest convex set containing such set of points.

A hyperplane in $\mathbb{R}^d$ is the affine hull of d independent points, and it separates the space into two halfspaces. To define a hyperplane and its halfspaces, we use the equation of outward pointing normal in the Hessian form, i.e.

$$\vec{n} \cdot x - s = 0, \tag{2.31}$$

where $\vec{n} \in \mathbb{R}^{1\times d}$ is to denote the Hessian normal of d independent points, and s is



(a) Δ^1 a line (b) Δ^2 a triangle (c) Δ^3 a tetrahedron

Figure 2.2: Graphical illustrations of simplexes

the offset of hyperplane relative to a reference point (the space origin). Based on the equation value, the location of a point x relative to the hyperplane and halfspace can be revealed by the following three conditions:

$$\begin{cases} \text{if } \vec{n} \cdot x - s > 0, & x \text{ is above the hyperplane;} \\ \text{if } \vec{n} \cdot x - s = 0, & x \text{ is on the hyperplane as a coplaner;} \\ \text{if } \vec{n} \cdot x - s < 0, & x \text{ is below the hyperplane.} \end{cases} \tag{2.32}$$

For a finite set of points, the convex hull is also a polytope which is bounded by finite intersections of the halfspaces. The supporting hyperplanes of the halfspaces then defines the convex hull boundary. An extreme point of the convex hull (polytope) is called a vertex. The intersections of supporting hyperplanes on a convex hull are called facets, where each facet is also a convex polytope. For an edge on the hull where two different facets intersect, we name it as a ridge.

Moreover, for a set of $d+1$ independent points in $\mathbb{R}^d$, the convex hull is a d-simplex denoted as Δ^d; see Figure 2.2 for examples of simplexes when $d \in \{2, 3, 4\}$. It is clear to observe that each facet of Δ^d is a Δ^{d-1}, and each ridge is a Δ^{d-2}.

Last but not least, a simplicial complex in $\mathbb{R}^d$ is a set of Δ^i for $0 \leq i \leq d$, where each Δ^i belongs to some Δ^d. For a simple example shown in Figure 2.3, the simplicial complex of a set points in $\mathbb{R}^2$ is represented as a triangulated diagram which consists of triangles (Δ^2s) and line segments (Δ^1s). The vertices of the simplicial complex then form the convex hull of these points.

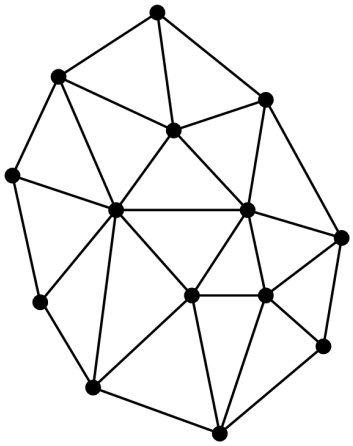


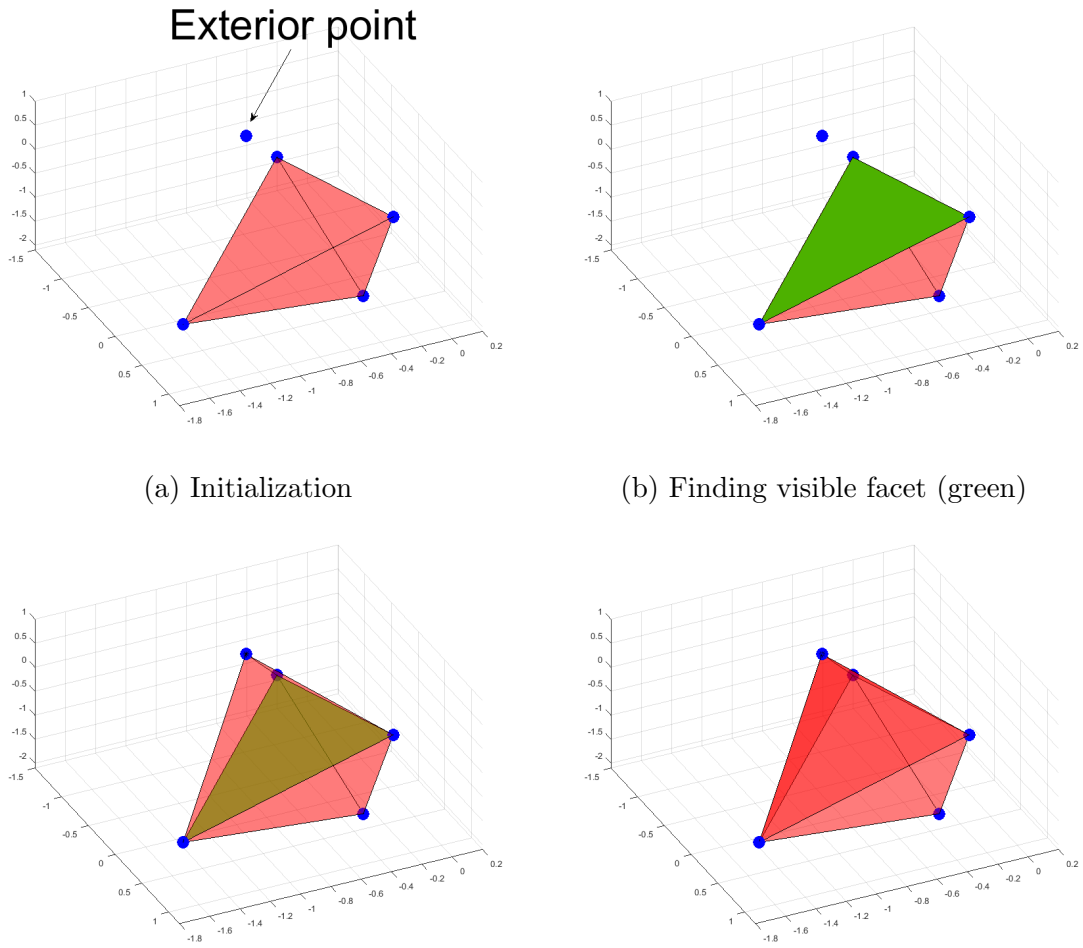
Figure 2.3: An example of simplicial complex in $\mathbb{R}^2$.

Compute convex hull with Quickhull (Qhull)

In Qhull, a set of input data points $X = \{x_1, \cdots x_n\} \in \mathbb{R}^d$ is assumed to be in general positions, where no sets of $d+1$ points define a hyperplane in $\mathbb{R}^{d-1}$. So the convex hull, denoted as $CH(X)$, is a simplicial complex comprised by a set of d-simplexes. By following such an assumption, Qhull is able to present $CH(X)$ with a set of enclosing facets $F = \{f_i, \ldots, f_m\}$, for $i = 1, \cdots, m$. Each facet f_i then has d vertices $V_i = \{v_{i,1}, \cdots, v_{i,d}\}$ and d ridges $R_i = \{r_{i,1}, \cdots, r_{i,d}\}$.

To compute the $CH(X)$, Qhull starts with $d+1$ random chosen points in X to create a d-simplex. For each facet f_i of the d-simplex, all points above the hyperplane are identified and assigned to f_s's outside set. Then, Qull gradually enlarges the hull with an expansion routine. In each iteration, the furthest outside point of f_i is taken as the expansion vertex where new facets are generated to link the existing facets. An iteration ends by deleting the covered facets and updating the new ones with their corresponding outside sets. As a graphical illustration, Figure 2.4 presents how Qhull expands a simplex with an exterior point in $\mathbb{R}^3$. The $CH(X)$ is completed when all points in X are enclosed, and the finalized lists of facets, vertices and hyperplane equations are generated as outputs.

Although Qhull is designed based on a simple expansion routine, it guarantees convergence. And the worst-case complexity is $O(n \log r)$ for $d \leq 3$ and



(a) Initialization

(b) Finding visible facet (green)

(c) Constructing new facets

(d) Delete redundant facets

Figure 2.4: An example of Qhull expansion routine in $\mathbb{R}^3$

$O(nr^{\lfloor d/2 \rfloor}/(\lfloor d/2 \rfloor ! r))$ for $d \geq 4$, where r denotes the number of vertices been processed. From some earlier experiment results, Qhull is able to effectively avoid a significant amount of redundant hyperplane generations and distance tests than most of the basic randomized search algorithms [72]. For more details of Qhull, a brief algorithmic outline is summarized as Algorithm 3 in Appendix A, and the reader is advised to visit the official release website [74].

Similar to other fault detection methods, constructing a convex hull with normal training data could be considered as a modeling process. The difference is that we are using a geometric enclosure as the model, instead of using statistical modeling.

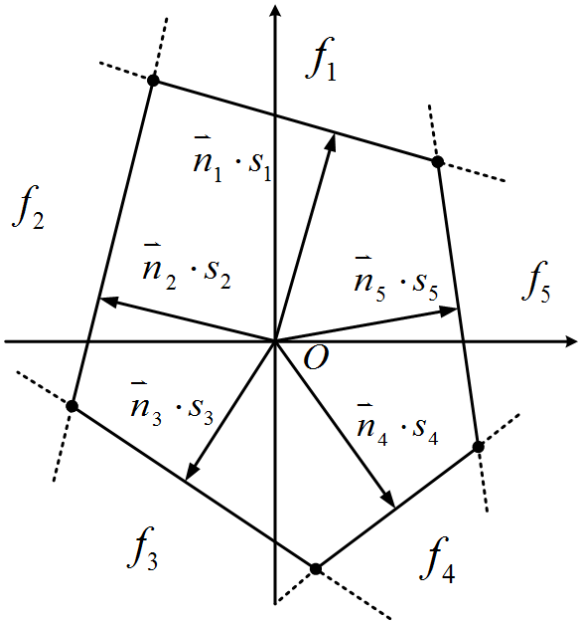


Figure 2.5: A two dimensional convex hull with all hyperplanes and enclosing halfspaces

With the aid of Qhull, the convex hull construction is free of parameter tuning.

2.3.3 Detection metric and threshold

As the process normal operating regime has been modeled by a convex hull, the goal for the detection stage is to identify and separate any possible faulty sample from the normal ones. To achieve this, we propose a distance based detection metric according to the geometric property of the convex hull. The setup is to check the distance of a test sample relative to the convex hull supporting hyperplanes in order to check if such sample is bounded by the normal operating regime. Different from the linear classifier in 1-class SVM, where only one hyperplane function is applied, our distance metric considers all hyperplanes included in by the convex hull.

Suppose a convex hull $CH(\mathbf{X})$ has already been constructed via Qhull for a training set stored in $\mathbf{X} \in \mathbb{R}^{n \times d}$, where n and d again refer to the number of samples and process variable, respectively. One of the handy features of Qhull is that all the enclosing facets with their hyperplane functions on $CH(\mathbf{X})$ are accessible in the algorithm outputs. For a clear presentation, we denote all m facets as a set $F = \{f_1, \cdots, f_m\}$.

For each facet f_i, the set of d vertexes is denoted as $V_i = \{v_{i,1}, \cdots, v_{i,d}\}$. All hyperplane functions are also stored by a matrix of all outward pointing norms and a vector of offsets as

$$\vec{\mathbf{N}} = \begin{bmatrix} \vec{n}_1 \\ \vdots \\ \vec{n}_m \end{bmatrix} \in \mathbb{R}^{m \times d} \text{ and } S = \begin{bmatrix} s_1 \\ \vdots \\ s_m \end{bmatrix} \in \mathbb{R}^{m \times 1}. \tag{2.33}$$

According to Preparata and Shamos [53], the intersection of halfspaces about the origin is equivalent to the convex hull of points in the dual space, see Figure 2.5 for an example in the two dimensional space. Thus, the interior space of $CH(\mathbf{X})$ could be defined as a set of points that satisfy the system of m linear inequality as:

$$\{x \mid \vec{\mathbf{N}} x^\top - S \leq 0_{(m \times 1)}, x \in \mathbb{R}^{1 \times d}\}. \tag{2.34}$$

For checking whether or not a new test sample $\mathbf{x} \in \mathbb{R}^{1 \times d}$ belongs to the interior space of $CH(\mathbf{X})$, we could simply adopt the linear inequality in Equation (2.34) to design the following distance metric:

$$D(\mathbf{x}) = \vec{\mathbf{N}} \mathbf{x}^\top - S \in \mathbb{R}^{m \times 1}. \tag{2.35}$$

The corresponding detection logic is given by

$$\begin{cases} \text{if } D(\mathbf{x}) \leq 0_{m \times 1}, & \mathbf{x} \in CH(\mathbf{X}) \text{ as a normal sample;} \\ \text{otherwise,} & \mathbf{x} \notin CH(\mathbf{X}) \text{ as a fault.} \end{cases} \tag{2.36}$$

Notice that the relative distances between any test sample to all convex hull hyperplanes are captured by $D(\mathbf{x})$. As points are bounded by $CH(\mathbf{X})$ only when $D(\mathbf{x}) \leq 0_{m \times 1}$, the detection threshold is then naturally set as the zero vector. Unlike conventional norm metrics and calculated thresholds used in other fault detection methods (e.g. T^2 and SPE), here the resulted detection logic is strict; and from the conditions in (2.36), we are able to inspect a faulty sample to decide which particular set of facet(s) it exceeds.

2.3.4 Curse of dimensionality and recursive training

Like all semi-supervised methods, CH-FD only uses the data from a normal class for training. The quality of training samples might directly affect its detection perfor-

mance in terms of the type I error (FAR) and type II error (MAR). For instance, a set of normal data, which have been collected when the system is lack of excitations or disturbances, may not fully reveal the dynamics of the system. On the other hand, as the number of monitored variables and dimensionality of the measurements increases, the collected data points turn to spread out into a larger volume and the cluster become less dense in the Euclidean space, which increases the complexity of the convex hull computation. Such effect is commonly known as the 'Curse of Dimensionality', and it is also the main burden which causes convex hull based anomaly detection suffer from high type I errors [55].

One way to tackle this issue is trough the recursive training, where we repeatedly reconstruct and validate the convex hull with the enlarged training sets (more normal samples), till the FAR of the validation samples converges to a desired level (e.g. below 1-5%). In our later examples, this strategy will be applied when monitored process units contain more than three variables.

There are several other good techniques available for reducing the FAR from univariate alarm monitoring study. One simple way is to pre-filter the data with some designed linear or non-linear filters. For a more advanced tool, a on- or off-delay timer could also be utilized when CH-FD is announcing the abnormality. However, here we aim to uncover the full potential of CH-FD and analyze its performance. In later comparison studies with other methods, this work only consider the raw detection results generated by the logic in Equation (2.36).

2.3.5 Illustrative example I

In this section, we draw a brief comparison of all 4 data-driven fault detection methods covered so far through a simple numerical example. Consider a linear bivariate system: $x_2 = x_1 + g$, where x_1 and x_2 are the two monitored variables, and g is a random disturbance with the uniform distribution $U(-1.5, 1.5)$. First, 495 normal samples (pairs of x_1 and x_2) were randomly generated with $x_1 \in [-1.5, 1.5]$. 6 faulty samples were later injected where they were located closely to the normal samples cluster but not following the distribution pattern. A scatter plot of all normal and faulty samples

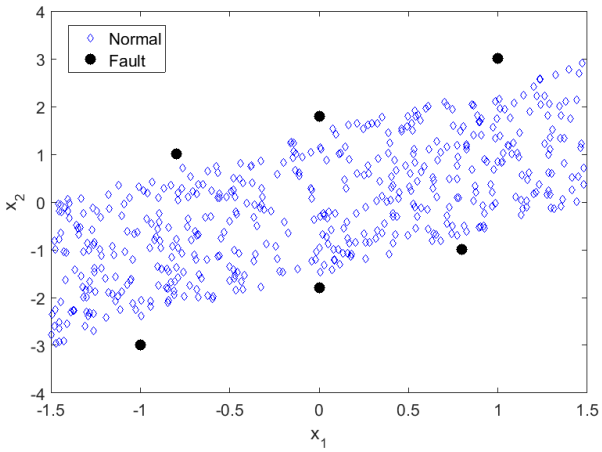


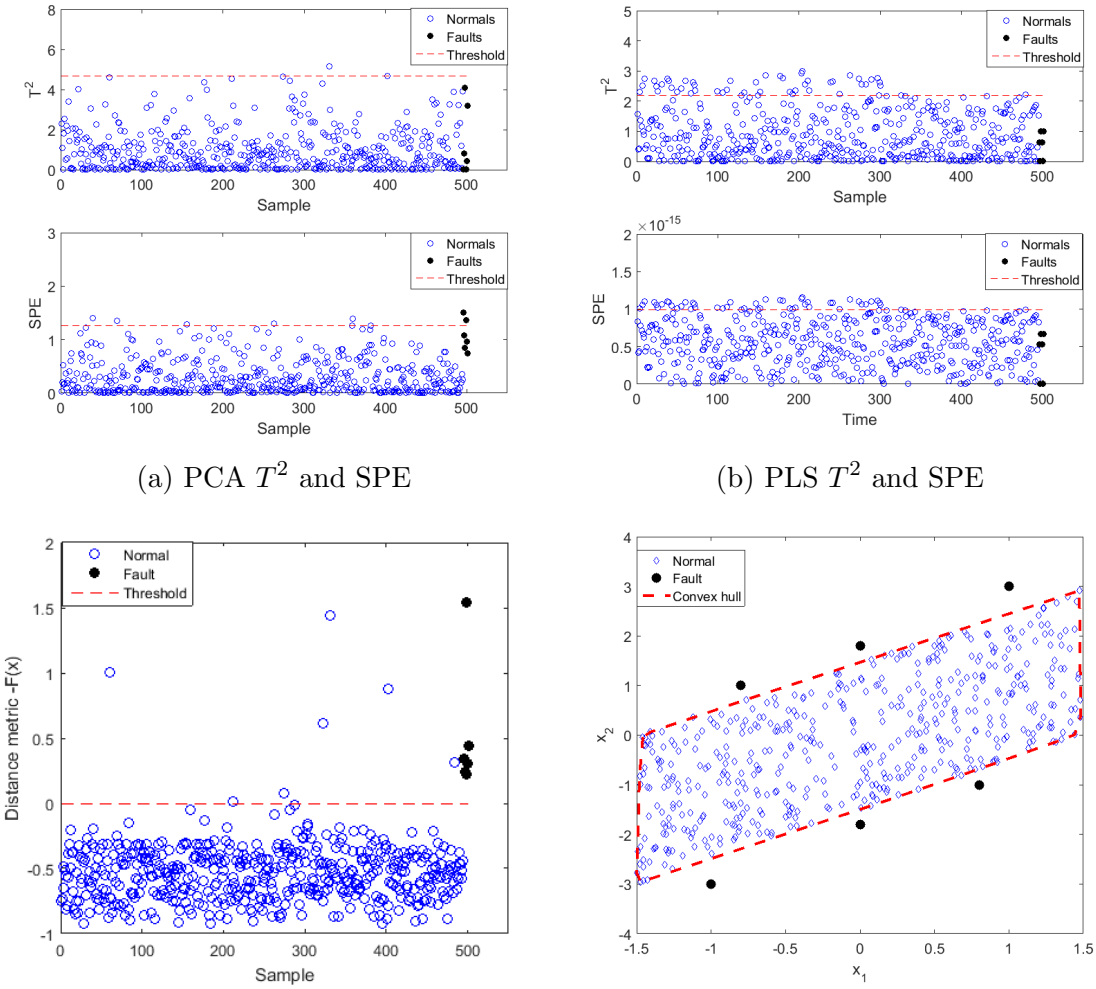
Figure 2.6: Illustrative example I – data visualization.

is presented on Figure 2.6.

In this example, the first 300 normal samples were used as the training data, and the rest 195 normal samples combined with the 6 faults were served for detection tests. The PCA model was built with 1 PC to capture 85% of the total variance of the training data. PLS considered x_1 as the model input and x_2 as the model output, where the regression was finalized with 1 LV. The detection thresholds of PCA and PLS were calculated based on the 99% confidence level. For 1-class SVM, the training data were projected into the feature space with a rbf kernel ($\gamma = 1$), and the model was built to correctly classify 99% of the training samples. The remaining 201 samples were projected in different models accordingly.

As shown in Figures 2.7a and 2.7b, PCA and PLS both had poor detection results for the 6 faulty samples. Only two successful detections were reported by the SPE metrics of PCA; otherwise, faults samples were all fussed with the normal. Even though, the monitored system is a linear one, such a result is expected, since the training samples were not normally distributed (PCA and PLS assume linear relationships).

On the other hand, 1-class SVM presented good results as shown in Figure 2.7c. All 6 faulty samples could be clearly separated from the normal ones by the detection threshold fixed at zero. However, it's also clear to observe that several high metrics



(a) PCA T^2 and SPE

(b) PLS T^2 and SPE

(c) 1-class SVM

(d) CH-FD

Figure 2.7: Illustrative example I – detection results.

appeared among the training and validation samples. If an empirical threshold was designed based on the metrics values in the validation set, these so-called outliers may significantly windup the limit and thus lead to poor detection results. This is because the rbf is a Gaussian based kernel function. Once the training samples are not jointly Gaussian distributed, the kernel may have difficulty to fit the distribution and regard certain normal samples as outliers [75].

For our new proposed method, a convex hull of training samples are first constructed via Qhull. According to the demonstration in Figure 2.7d, all validation

samples were well contained by the convex hull, and faulty points could be clearly separated. Such good detection result is due to the fact that no assumptions of data distribution has been made, and the convex hull is able to accurately capture the system normal operating regime as a geometric pattern of the training samples.

Chapter 3

Visualization Based on Parallel Coordinates

3.1 Overview

In the previous chapter, we introduced a convex hull based data-driven fault detection technique (CH-FD). In this new method, the process normal operating regime is realized as a convex hull which forms the geometric enclosure of normal process data. The monitoring visualization for a small system, where the number of monitored variables is no more three, could be generated by graphing the convex hull with the scatter plot of testing samples in the Cartesian space. On the plot, exterior points of the convex hull could be visually identified as faults. However, when the process contains more than three measurements, this visualization setup is no longer applicable due to the dimensionality constraint in the Cartesian space. In this chapter, we develop a high-dimensional monitoring visualization tool with the aid of parallel coordinates plots, in order to better present the detection results of CH-FD for a multivariate process.

3.2 Background

Parallel coordinates are a commonly used visualization technique for high-dimensional geometry and multivariate data. The history of this tool could be traced back to the original work of Maurice in 1885 [76], when he presented his work on a coordinate

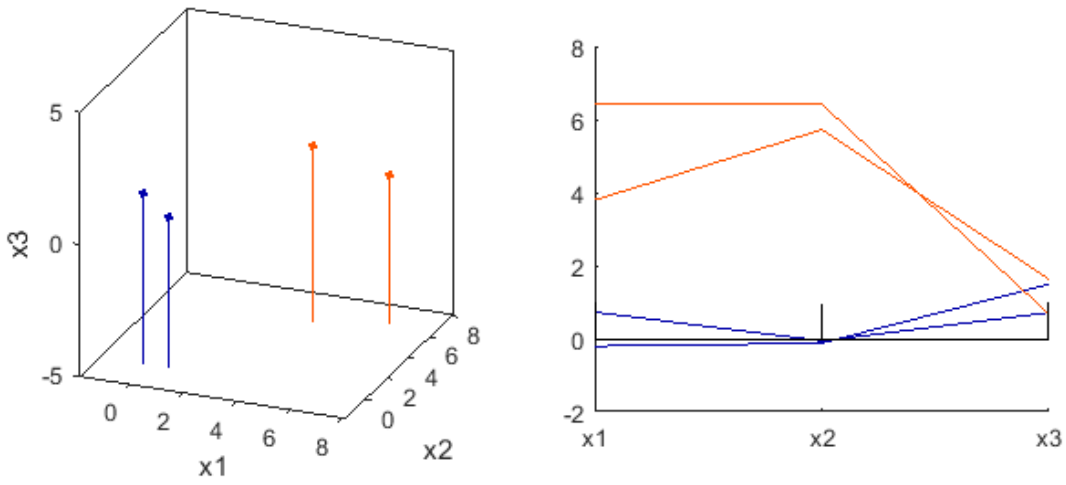


Figure 3.1: A simple example of a parallel coordinates plot [3]

transformation. On a typical parallel coordinates plot, the variable axes are placed vertically and aligned in parallel to each other. As shown in the example in Figure 3.1, a set of d-dimensional data points are presented as polylines under this coordinates setup. Each ployline consists of d vertices which is on the variable axes, indicating the coordinate of each data point on each dimension. As the variable axes could be freely added in this coordinate system, the data dimensionality is no longer a constrains, which allows us to visualize the data in a high-dimensional sense. Ever since the property of this coordinate system had been further uncovered by Inselberg [77], this powerful tool began to lead the fashion of visual based data mining [78–81].

The key concept of using parallel coordinates for data analytics is to identify the multivariate dependencies and correlations through the data pattern presented on the plot [79]. When a set of bivariate data holds a particular correlation or distribution, the corresponding data mapping on parallel coordinates also presents a unique geometrical pattern [82]. As a simple demonstration, Figure 3.2 presents four types of common bivariate data patterns in the Cartesian space with their mapping on the parallel coordinates. For the correlations that show convex patterns, see the ellipsoid and convex hull in Figure 3.2, the mapping of data points forms a pair of envelopes which also show the convexity [81].

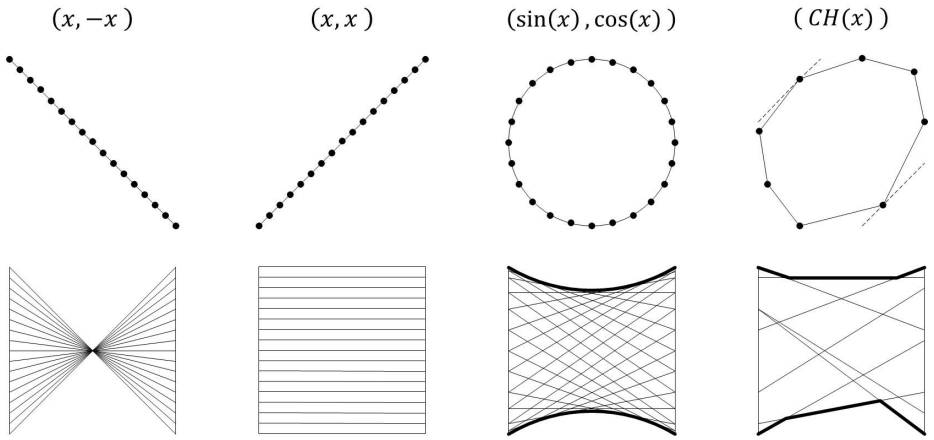


Figure 3.2: Geometric patterns of 4 different correlations on parallel coordinates

In existing literature, one of the successful applications of parallel coordinates in process monitoring is the multivariate alarm management tool proposed by Brooks [83]. In that work, techniques were developed to model the system normal behavior on parallel coordinates with the high-dimensional envelope of process data. Methods for multivariate alarm rationalization and control tuning based on parallel coordinates were also introduced. The main results of Brooks' work later led to a process management package which kept serving the industry till today. More information of the package can be found in [84]. As discussed in the previous chapter, our newly developed convex hull based fault detection shares a similar concept with Brooks' method, where we model the process normal operating regime as a geometrical structure. In the following sections, we adopt Brook's idea and study the way to realize the convex hulls as envelopes on parallel coordinates, so that high-dimensional process normal operating regimes could be displayed on data monitoring plots.

3.3 Preliminary on parallel coordinates

In this section, a brief review of parallel coordinates fundamentals is provided based on Inselberg's book published in 2009 [85]. The material covers the basic knowledge which is required for the later discussions of convex hull envelopes.

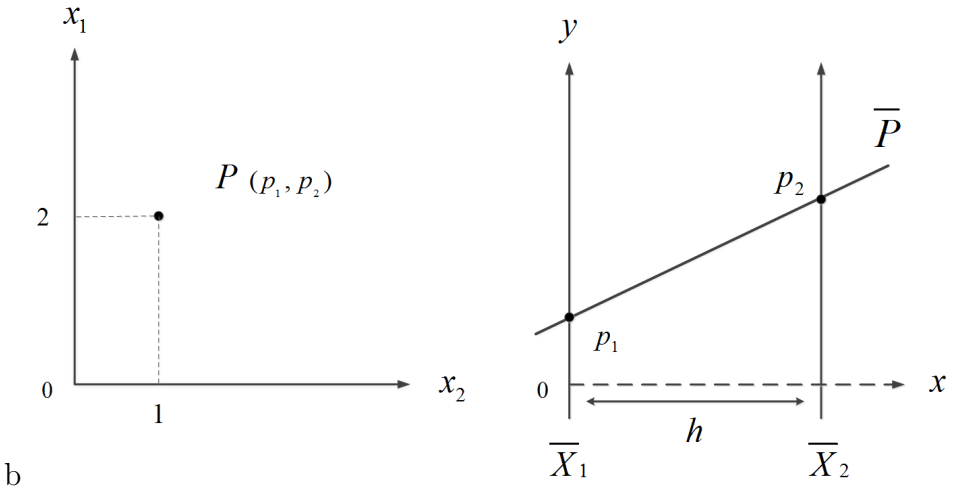


Figure 3.3: Mapping a 2 dimensional point on parallel coordinates

3.3.1 Coordinates setup

Consider a two dimensional plane in the Cartesian coordinate system shown in Figure 3.3 (left), where x_1 and x_2 are to denote the orthogonal axes. On the parallel coordinates, the equivalent variable axes are expressed in parallel on the xy plane shown in Figure 3.3 (right). The first vertical axis $\overline{X}_1$ coincides with y; and the second axis $\overline{X}_2$ on the right has a horizontal distance h relative to $\overline{X}_1$. For the simplicity, both origins on $\overline{X}_1$ and $\overline{X}_2$ are aligned to the origin of y axis, and all vertical axes are pointing in the same direction. Here we use the over-line '-' to indicate an object on parallel coordinates.

For a two-dimensional point $P = (p_1, p_2)$, its mapping from the Cartesian space to the parallel coordinates is achieved with the point to line transformation, where the line image is denoted by $\overline{P}$. The coordinates of both intercepts on $\overline{X}_1$ and $\overline{X}_2$ are given by $(0, p_1)$ and (h, p_2), respectively. Based on these coordinates, the line equation of $\overline{P}$ could be derived as:

$$\overline{P} : y = \frac{p_2 - p_1}{h} x + p_1, h \neq 0. \tag{3.1}$$

When plotting high-dimensional data points on parallel coordinates, we simply add up the vertical variable axes and evenly spread them on the xy plane. For example, the point $P = (P_1, P_2, \cdots, P_{d-1}, P_d) \in \mathbb{R}^d$ is displayed as a polyline $\overline{P}$

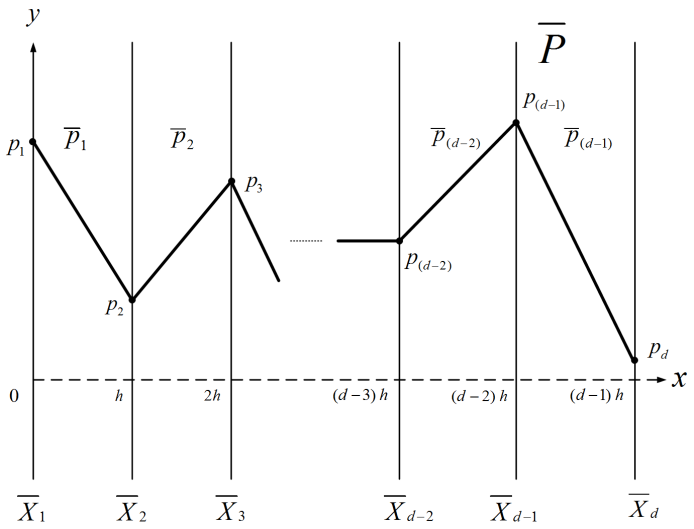


Figure 3.4: Mapping a high-dimensional data point on the parallel coordinates

as shown in Figure 3.4. The polyline is synthesized by a set of $d-1$ straight line segments $\{\overline{p}_1, \overline{p}_2, \cdots, \overline{p}_{d-2}, \overline{p}_{d-1}\}$. Each $\overline{p}_i$ is between a pair of neighboring vertical axes, namely, $\overline{X}_i$ and $\overline{X}_{i+1}$ and still follows Equation (3.1) in the form:

$$\overline{p}_i : y = \frac{(p_{i+1} - p_i)}{h} x + p_i, h \neq 0 \ , \ i \in [0, d-1]. \tag{3.2}$$

As variable axes are placed in parallel, the order of axes on the plot is not trivial to arrange. For a display of d-dimensional data points, the total number of unique axes orders is $d(d-1)/2$. This means that the axes order could be freely alternated, generating different geometric patterns of the data points.

3.3.2 Point to line duality

One of the most interesting property of parallel coordinates is that it holds a so-called point to line duality (*point* $\leftrightarrow$ *line*) with the Cartesian coordinate system. Consider a line on the x_1x_2 plane defined as

$$l : \ x_2 = mx_1 + b, m \neq 1, \tag{3.3}$$

where m is the slope, and b is the intercept on x_2 shown on Figure 3.5. For any two points A_1 and A_2 on l, their mapping on parallel coordinates can be derived based

on Equation (3.1) as:

$$\overline{A}_1 : y = \frac{ma_1 + b - a_1}{h}x + a_1,\ \overline{A}_2 : y = \frac{ma_2 + b - a_2}{h}x + a_2 \tag{3.4}$$

With simple calculation, it could be verified that $\overline{A}_1$ and $\overline{A}_2$ have a unique intercept on the xy plane at

$$\bar{l} = (\frac{h}{1-m}, \frac{b}{1-m}), m \neq 1. \tag{3.5}$$

As A_1 and A_2 are able to define l in the Cartesian space, $\bar{l}$ is then the unique mapping of l on parallel coordinates. When $m = 1$, $x = h/(1-h) \to \infty$, indicating the mapping of all points on l are parallel lines on the xy plane. Otherwise, depended on the value of m, $\bar{l}$ is located at different regions of the xy plane, as shown in Figure 3.6. Combined with the point to line transformation described in Equation (3.1), we then have the fundamental *point* $\leftrightarrow$ *line* duality between the Cartesian coordinates and the parallel coordinates.

3.3.3 Curves to envelopes

According the Inselberg's illustration in Chapter 7 of [85], the fundamental duality could be further extended to the transformation of curves to envelopes between the two coordinates systems. Consider a conterminous and differentiable curve C on

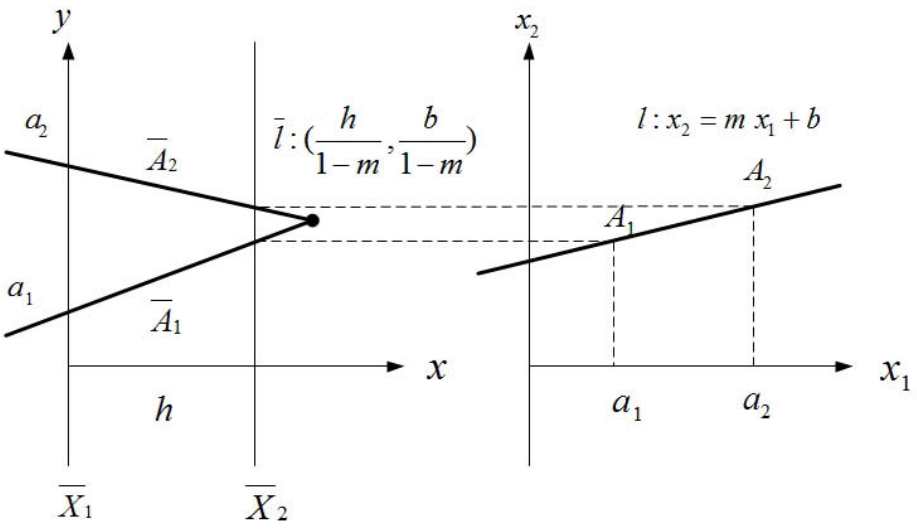


Figure 3.5: Mapping a line in $\mathbb{R}^2$ on parallel coordinates

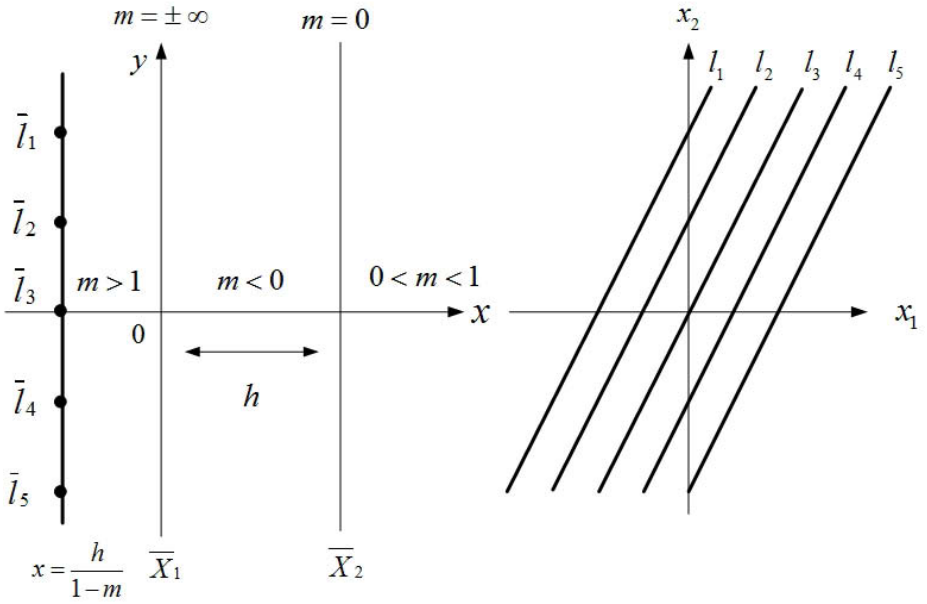


Figure 3.6: The effect of the line slope m on the horizontal position of line image in parallel coordinates

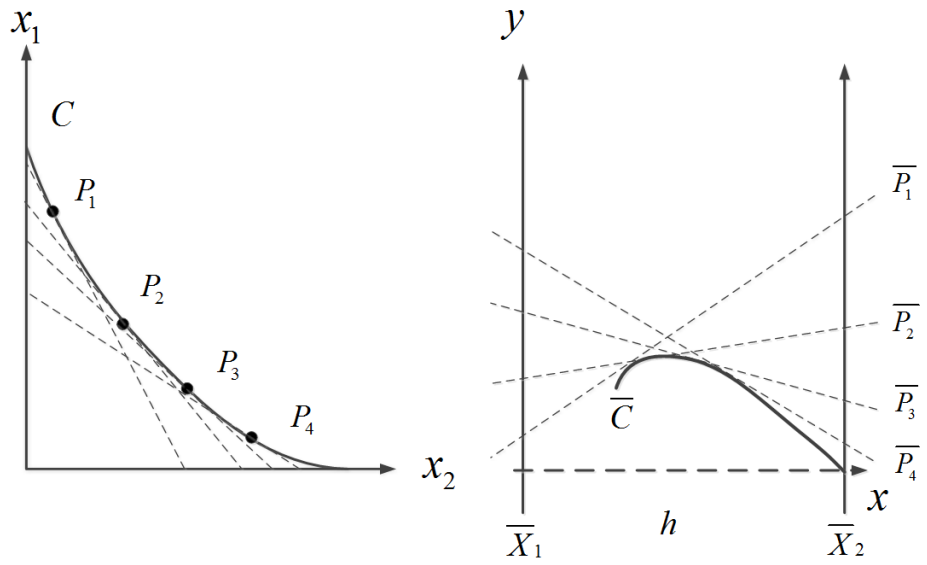


Figure 3.7: Mapping a continuous curve C as an envelope $\overline{C}$ on parallel coordinates

Figure 3.7 (left). The line mappings of points on C, $\overline{P}_i$ on parallel coordinates, are plotted on the right as dash lines, which are all tangent to a unique envelope denoted as $\overline{C}$ on the xy plane. All points on $\overline{C}$ are the point images of all tangential

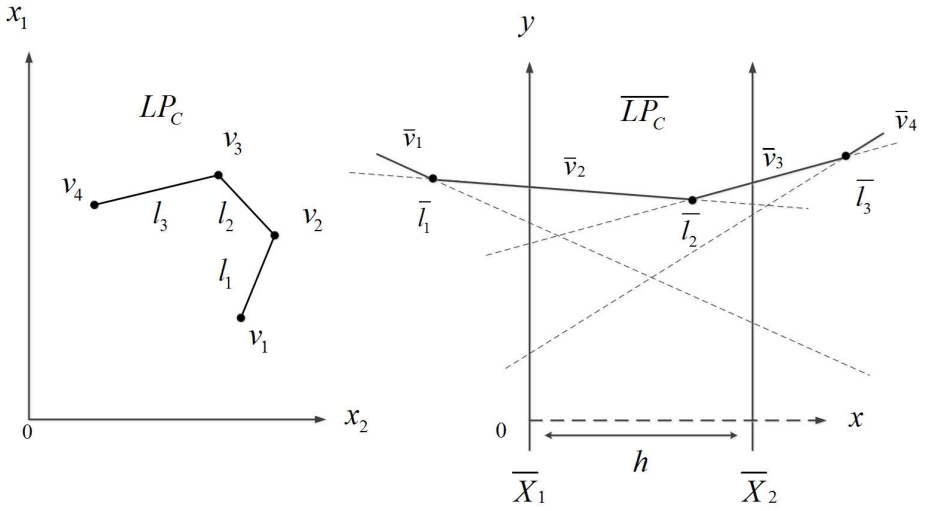


Figure 3.8: A line point chain LP_C and its envelope $\overline{LP}_C$ on parallel coordinates

supporting lines of points P on C. If the curve is defined as a continuous function as $C : x_2 = f(x_1)$, the corresponding envelope $\overline{C}$ could be analytically derived based on the fundamental duality as:

$$x = \frac{h}{1 - f^{'}(x_1)} \text{ and } y = \frac{x_2 + x_1 f^{'}(x_1)}{1 - f^{'}(x_1)}, \tag{3.6}$$

where $f^{'}(x_1)$ is the derivative. In addition, if the envelope is also defined by a continuous function as $\overline{C} : y = g(x)$, the curve C could be retraced as:

$$x_1 = y - x g^{'}(x) \text{ and } x_2 = y + (h - x) g^{'}(x). \tag{3.7}$$

Similar transformation also holds for a curve formed by a set of piecewise line segments, e.g, line point chains or polylines. Consider the line point chain LP_c, depicted in Figure 3.8, which is formed by a set of three lines $\{l_1, l_2, l_3\}$. The points $\{v_1, v_2, v_3, v_4\}$ are to denote the vertices of the chain. As each l_i is the tangential supporting line of all points on it, its mapping $\overline{l_i}$ on the xy plane is a vertex on the envelope $\overline{LP}_c$. For the vertices of LP_c which have two supporting lines, i.e. v_2 and v_3, their line mapping connect two different vertices. Therefore, Based on the *point* $\leftrightarrow$ *line* duality, it's clear to conclude that the envelope of a line point chain on parallel coordinates is also a line point chain [86]. For a two-dimensional convex

hull, the enclosing facets (lines) are connected by the vertices to form this type of line point chain. This result then provides an important insight of how to derive the envelope of bivariate convex hulls on parallel coordinates.

3.4 Convex hull envelopes in parallel coordinates

After being equipped with the basics of parallel coordinates, this section is to illustrate the mapping of bivariate convex hulls and present an envelope plotting technique.

3.4.1 Property of convex hull envelopes

In the existing literature, the transformation of convex sets from the Cartesian space to the parallel coordinates is no longer a new topic. Back to 1985, Inselberg had already discovered the mapping pattern when he generalized the *point* $\leftrightarrow$ *line* duality with the theory from projection geometry [77, 82]. The main result was summarized in [86] by the duality of *estar* $\leftrightarrow$ *hstar*, where the *estar* represents an ellipsoid shaped convex sets, and the *hstar* represents a point set which is bounded by general hyperbola envelopes. Here the rigorous definition of *hstar* is adapted from [86].

Definition 1 *A point set HP is an hstar iff its boundary is a closed piecewise line point curve formed by* $Cc_1 \cup Cc_2 \cup l_1, l_2$*, where* Cc_1*,*Cc_2 *are convex upward and convex downward piecewise line point curves having no vertical edges and left and right endpoints* L_i*,*R_i *respectively; for* $i = 1, 2, l_i = L_j R_k, j \neq k$*, are their common coursing supporting lines, and not vertial lines separate the chains (see Figure 3.9 for an example).*

Based on the duality of convex sets, for a two-dimensional convex hull denoted as CH, the corresponding envelope on the parallel coordinates is the *hstar* boundary denoted as $\overline{CH}$, as shown on Figure 3.10. According to Definition 1, $\overline{CH}$ consists of a concave up envelope $\overline{LP}_u$ on the top and a concave down envelope $\overline{LP}_l$ at the bottom. Here we use the subscripts to distinguish the envelopes based on their positions. Both $\overline{LP}_u$ and $\overline{LP}_l$ are the corresponding mappings of the upper and lower line point chains

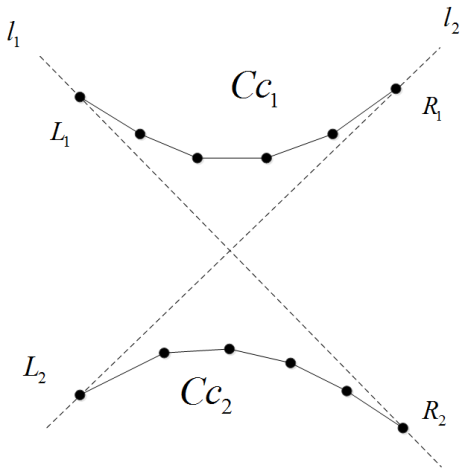


Figure 3.9: *hstar* – a general hyperbola shape

on CH, see LP_u and LP_l on Figure 3.10 (left). The so-called starting and ending points of each chain denoted as v_l and v_r are the vertices where a pair of supporting lines, with slope $m = 1$, could be draw to bound the convex hull in the Cartesian space.

The special case happens when the CH contains facet(s) with slope $m = 1$, see Figure 3.11 (left). Since points on these facets are parallel lines on the xy plane, i.e, $\overline{v}_3 \parallel \overline{v}_4$ and $\overline{v}_7 \parallel \overline{v}_8$ on Figure 3.11 (right), the asymptotes of $\overline{LP}_u$ and $\overline{LP}_l$ are then decoupled. Correspondingly, the vertices of these facets are considered as the starting

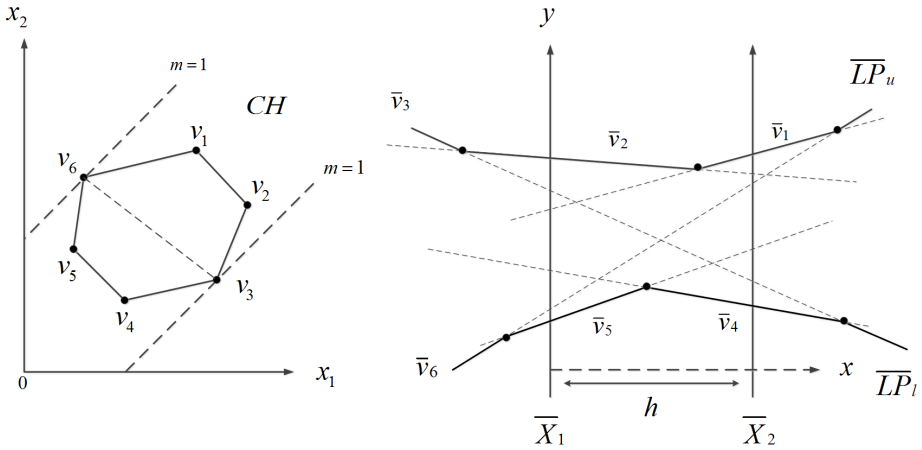


Figure 3.10: Mapping the vertices of a convex hull on parallel coordinates

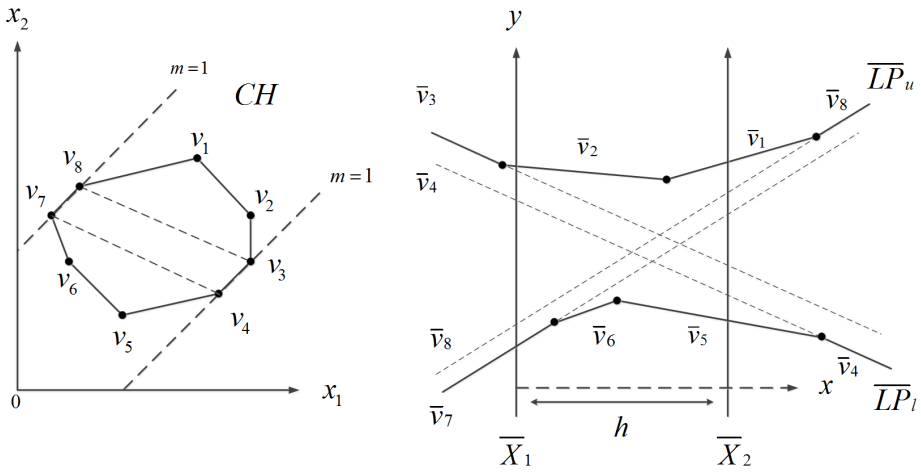


Figure 3.11: A special case when a convex hull contains facets with slope $m = 1$.

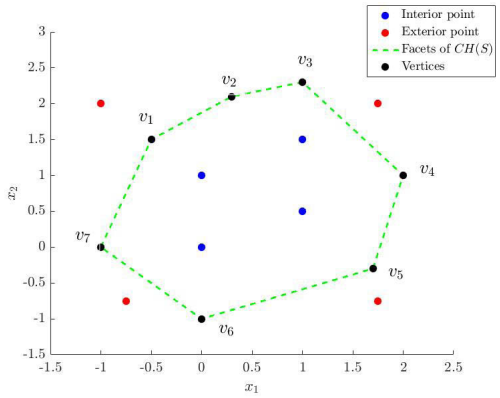
and ending points, e.g, LP_u starts with v_{ur} and ends at v_{ul}, while LP_l starts with v_{ll} and ends at v_{lr}. According to these observations, the unique envelope of any convex hull could be systematically plotted once the upper and lower chain are correctly identified.

The most important property of this mapping is that a convex hull envelope on parallel coordinates also reflects the convexity according to Corollary 1 in [86].

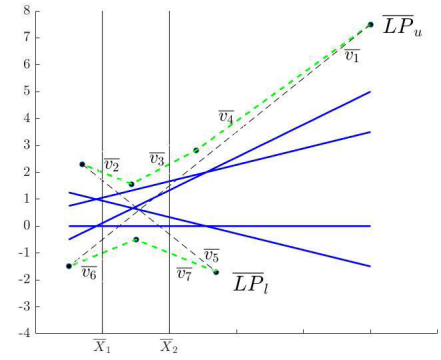
Corollary 1 *P is an interior point of CH iff $\overline{P}$ separates $\overline{LP}_u$ and $\overline{LP}_l$ (interior points of estars $\leftrightarrow$ interior lines to hstars).*

More intuitively, a line $\overline{P}$, representing an interior point $P \in CH$, is always bounded by $\overline{CH}$, such that $\overline{P}$ is on or below $\overline{LP}_u$, and also on or above $\overline{LP}_l$. For an exterior point $P \notin CH$, the $\overline{P}$ may not pass the space that is bounded by $\overline{CH}$, or it intercepts with either $\overline{LP}_u$ or $\overline{LP}_l$. An illustrative example is presented in Figure 3.11, where the envelope, interior points and exterior points are depicted in green, blue and red respectively. For rigorous proof of such convex duality, readers could refer to Inserlberg's detailed discussion in [86] and the references therein.

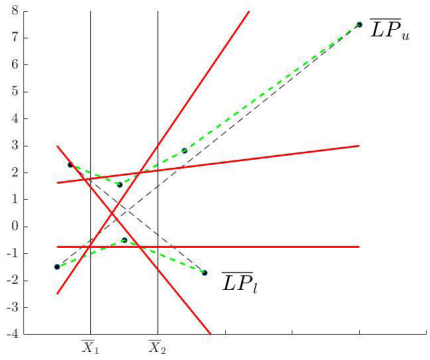
Recall in the CH-FD which we discussed in Chapter 2, a convex hull of process data is utilized to define the system normal operating regime. This result from Inselberg indicates any bivariate convex hull could be realized as a unique pair of convex



(a) A convex hull vs. its interior and exterior points



(b) The convex hull envelope vs. lines of its interior points



(c) The convex hull envelope vs. lines of its exterior points

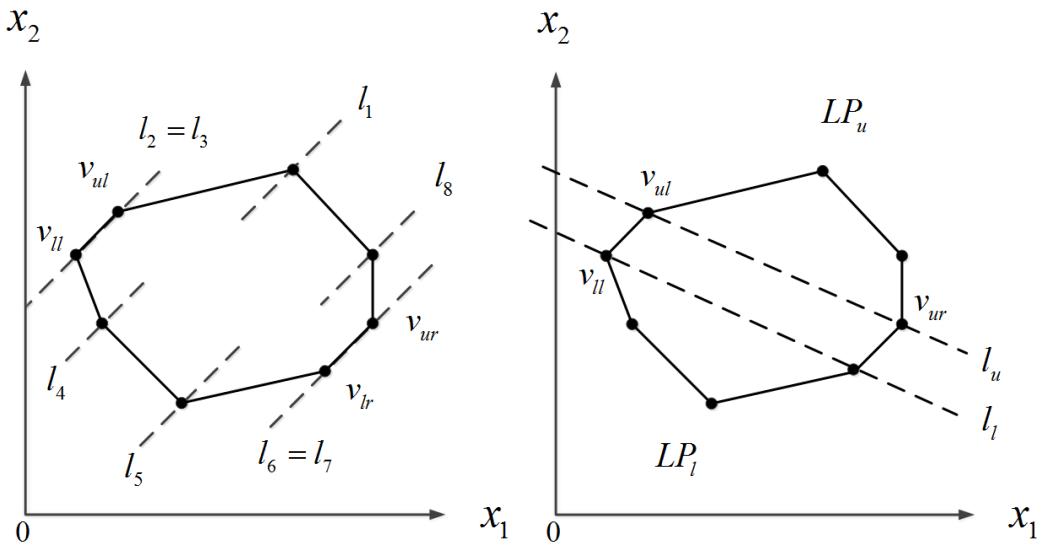
Figure 3.12: A demonstration of convexity on parallel coordinates

envelopes on the parallel coordinates, and the exterior points of the convex hull could be visually detected, as their line mappings violate the geometric pattern of the envelopes. This good feature would allow us to reconstruct the normal operating regime on parallel coordinates.

3.4.2 Plotting convex hull envelopes

After we demonstrated the mapping of convex hulls on parallel coordinates, one might conclude that this transformation could be easily derived through the basic *point* $\leftrightarrow$ *line* duality. However, when dealing with random convex sets, how to systematically construct the envelops is still a practical challenge. In the existing literature, the

discussion of such convex envelope plotting on parallel coordinates is very limited. The real time convexity algorithm presented by Inselberg seems to be the only good solution to this problem, see Algorithm A2 in [86]. The idea is to initialize the envelopes with three non-collinear points in a given dataset. For each remaining point, check if there is any outer intersection with the upper or lower envelope, and then gradually update the envelope till all line images of the samples in the set are bounded. Although Inselberg's algorithm holds convincing results for any given two-dimensional convex sets, the expansion routine, which involving all points in the set, may seems overwhelming and redundant. For the rest of this section, a simplified approach for plotting convex envelopes is introduced. We mainly take the advantage of using the Qhull algorithm to pick out the vertices of a convex set in advance, and then directly render the facets of the convex hull on parallel coordinates to construct the envelope. Details of the plotting procedure is illustrated in the following steps, and the main algorithmic routines are summarized in Algorithm 2.



(a) Step 2: searching for starting points (b) Step 3: identifying LP_u and LP_l

Figure 3.13: Locating the chain starting points and clarifying upper and lower chains on a convex hull

Step 1. Initialization:

First, for a set of data points $X \in \mathbb{R}^2$, the convex hull $CH(X)$ is computed with

Qhull to determine the set of vertices V.

Step 2. Identifying the starting points of line point chains:

Next, we identify the starting points of upper and lower chains on the convex hull. To search for the possible candidates, a passing line l_i with slope $m = 1$ is generated for each $v_i \in V$, see Figure 3.13a. If l_i is above or below all remaining vertices, v_i is then considered as a possible left or right starting point, and would be stored in the candidate sets, i.e. LS and RS, accordingly. Obviously, when $CH(X)$ contains facet(s) with slope $m = 1$, more vetices would be assigned in the candidate sets. Then, in each candidate set, based on their relative positions we clarify which vertex starts the upper or lower chain, see v_{ul}, v_{ur}, v_{ll} and v_{lr} on Figure 3.13a.

Step 3. Identifying all vertices on the upper and lower chains:

Once the chain starting (ending) points are identified, they are utilized as the reference to allocate each $v_i \in V$ on the upper or lower chain correctly. A simple way is to draw the lines to connect the left and right starting points, i.e. l_u passes $v_{ul}v_{ur}$ and l_l passes $v_{ll}v_{lr}$ as shown in Figure 3.13b. Then, for each $v_i \in V$ which is on or above l_u, it is obviously a member of the upper chain and would be stored in the set LP_u. Whereas, for each v_i, which is on or below l_l, it would be assign to LP_l as a member of the lower chain. Once all vertices complete this position test, all members of the upper and lower line point chains are finalized.

Step 4. Mapping the line point chains on parallel coordinates:

Finally, to render the unique envelope of the convex hull on parallel coordinates, the line point chains which bounded all line images of vertices need to be correctly derived. First, we sort the vertices on convex hull's upper and lower chains in a counter clockwise order (CCW) based on their polar angle relative to the starting points. In our convention, we start the storing on the upper chain LP_u with the right starting point v_{ur}, and for LP_l we choose to start with the left point v_{ll}. Once the sorting is complete, facets on each chain are then defined by two neighboring vertices in the list, see U_1-U_3 and L_1-L_3 on Figure 3.14 (left). As these facets are line segments in $\mathbb{R}^2$, their point images on parallel coordinates could be derived with Equation (3.5), see $\overline{L}_1$-$\overline{L}_3$ and $\overline{U}_1$-$\overline{U}_3$ on Figure 3.14 (right). By plotting the line segments which link

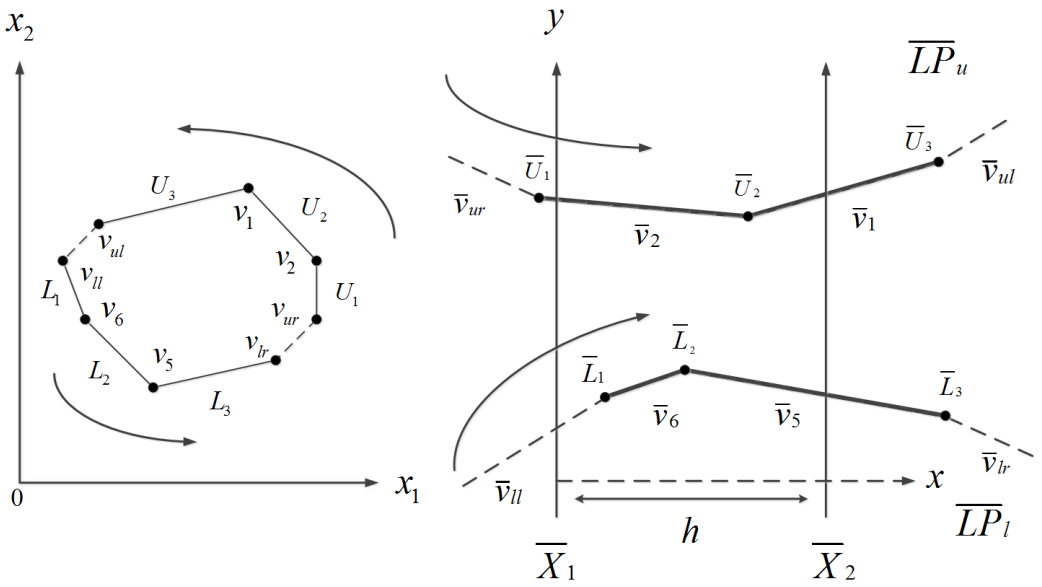


Figure 3.14: Step 4: converting facets on a convex hull into vertices of the envelope on parallel coordinates

these points on the xy plane, the upper and lower chains, namely, $\overline{LP}_u$ and $\overline{LP}_l$, of the convex hull envelope are then correctly rendered.

As this plotting technique is developed based on the geometrical property of convex hull envelopes, it guarantees to display the correct line point chain of the envelope on parallel coordinates. Notices that, this rendering algorithm does not directly compute the convex envelope of a dataset on parallel coordinates; it only aims to render the envelope of a known convex hull for the ease of implementation. For real time convex envelope computation, it's better to refer to Inselberg's original algorithm presented in [86].

Algorithm 2 Plot convex hull envelope

Input:
  V - set of convex hull vertices
Initialize:
  LS - set of possible LP chain starting points on the left
  RS - set of possible LP chain starting points on the right
  LP_u - set of vertices on convex hull upper chain
  LP_l - set of vertices on convex hull lower chain
  $\overline{U}$ - set of points on upper envelope
  $\overline{L}$ - set of points on lower envelope
for each $v_i \in V$ **do**
  compute line l_i pass through v_i with slope $m = 1$
  if $\forall\ v_j \in V$ is on or below l_i **then**
    assign v_i to LS
  else if $\forall\ v_j \in V$ is on or above l_i **then**
    assign v_i to RS
  end if
end for
for each $v_i \in LS$ **do**
  if v_i is above $v_j \in LS,\ \forall j \neq i$ **then**
    $v_{ul} \leftarrow v_j$ as the left start of upper chain
  else if v_i is below $v_j \in LS,\ \forall j \neq i$ **then**
    $v_{ll} \leftarrow v_j$ as the left start of lower chain
  end if
end for
for each $v_i \in RS$ **do**
  if v_i is above $v_j \in RS, \forall j \neq i$ **then**
    $v_{ur} \leftarrow v_i$ as the right start of upper chain
  else if v_i is above $v_j \in RS, \forall j \neq i$ **then**
    $v_{lr} \leftarrow v_i$ as the right start of lower chain
  end if
end for
assign v_{ul} and v_{ur} to LP_u, and assign v_{ll} and v_{lr} to LP_l
compute the line l_u that pass through v_{ul} and v_{ur}
compute the line l_l that pass through v_{ll} and v_{lr}
for each $v_i \in V$ **do**
  if v_i is above l_u **then**
    assign v_i to LP_u
  else if v_i is below l_l **then**
    assign v_i to LP_l
  end if
end for

Algorithm 2 Plot convex hull envelope (continued)

sort veritces in LP_u in CCW respect to v_{ur}
sort veritces in LP_l in CCW respect to v_{ll}
for $i = 1 : k - 1$ where k is the size of set LP_u **do**
    compute facet U_i that pass through $v_i vi + 1 \in LP_u$
    calculate the point image $\overline{U_i}$ of U_i with Equation (3.5)
end for
link all points in $\overline{U}$ in parallel coordinates as upper envelope $\overline{LP}_u$
for $i = 1 : z - 1$ where z is the size of set LP_l **do**
    compute facet L_i that pass through $v_i vi + 1 \in LP_l$
    calculate the point image $\overline{L_i}$ of L_i with Equation (3.5)
end for
link all points in $\overline{L}$ in parallel coordinates as upper envelope $\overline{LP}_u$

3.5 A parallel coordinates based visualization for convex hull based fault detection

According to our discussions in Sections 3.3 and 3.4, the parallel coordinates mainly have two advantages for high-dimensional process data visualization. First, it is able to display multivariate data without a constraint on the data dimensionality. Second, the envelopes of convex hulls on parallel coordinates not only present the information of vertices and facets, but also keep the convexity with its geometric patterns. With these good features, the parallel coordinates indeed could be well utilized for CH-FD in visualizing the detection results in a multivariate sense. For the rest of this section, we introduce a parallel coordinates based visualization which is specifically designed to facilitate CH-FD.

3.5.1 Design

The essential concept of this visualization is to imitate the normal operating regime obtained in CH-FD by a multivariate envelope on the parallel coordinate which represents the unique geometric pattern of normal process data. When monitoring the process, new data samples are also displayed on the parallel coordinates plot along with the envelope. By comparing the new samples' ploylines with the envelope, an

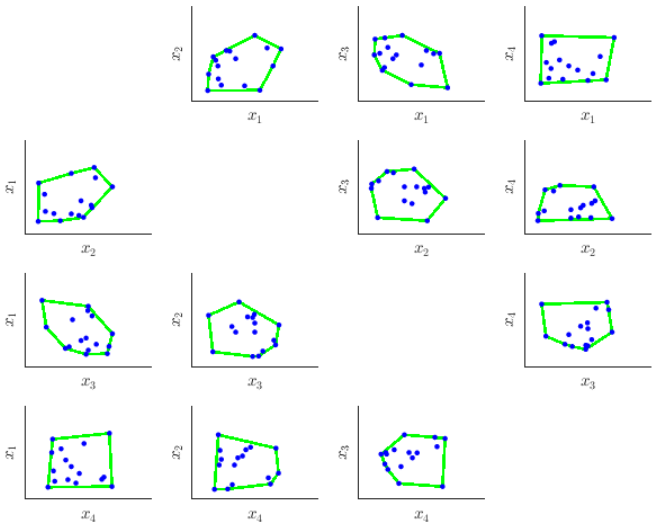


Figure 3.15: CH-FD high-dimensional visualization – computing convex hulls of all possible bivariate data clusters (an example in $\mathbb{R}^4$).

abnormality could be visually detected if any violation of the normal pattern is observed.

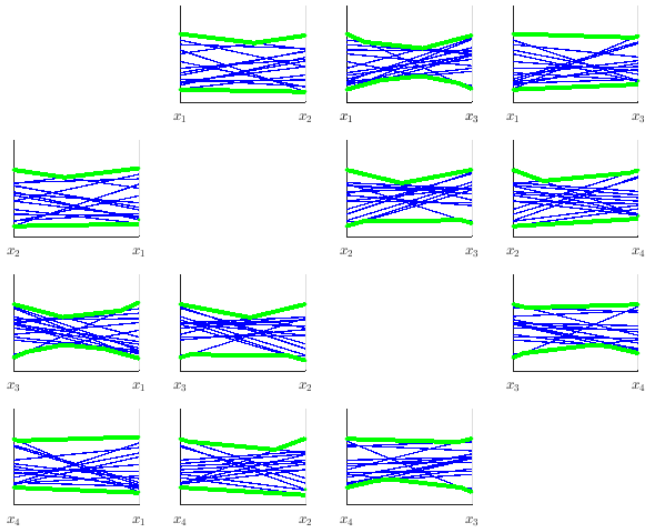


Figure 3.16: CH-FD high dimensional visualization – converting convex hulls in $\mathbb{R}^2$ into bivariate envelopes in parallel coordinates

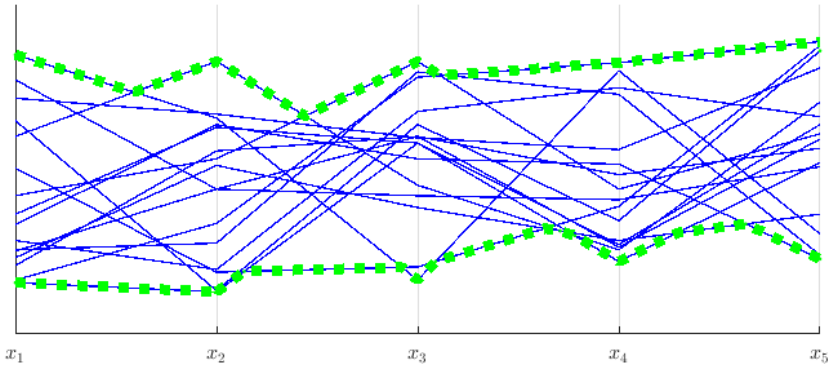


Figure 3.17: CH-FD high dimensional visualization – padding bivariate envelopes

In order to synthesis the high-dimensional envelope for a set of normal data, the two-dimensional convex hulls of all bivariate data clusters are first computed by Qhull. See Figure 3.15 for an example of randomly generated dataset with 4 variables, where the convex hull of each bivariate data cluster on the scatter plot is computed and shown in green. Next, these convex hulls are converted into bivariate envelopes and plotted on parallel coordinates with Algorithm 2. Figure 3.16 displays the rendering results. Lastly, to generate a complete envelope covering all monitored variables, we pad the bivariate envelopes in a user defined axes order. The resulted multivariate envelope is then able to contain all ploylines of normal samples, as shown in Figure 3.17, and represents the geometric pattern of the normal data. Notices that when padding the bivariate envelopes, we only keep the *hstar* region between the variables axes to ensure a clear presentation. Unfortunately, some information of each bivariate convex hull may be lost during this padding process.

3.5.2 Illustrative example II

To better demonstrate how this visualization tool could be utilized to facilitate CH-FD, we adapted the simulation of a simple open loop linear system suggested by [87] to present an illustrative example. The numerical description of the monitored system

is given as:

$$X = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0.3723 & 0.6815 \\ 0.4890 & 0.2954 \\ 0.9842 & 0.1793 \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \end{bmatrix} + \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} \tag{3.8}$$

The vector $[x_1, x_2, x_3]^\top$ denotes the three system states; $[s_1, s_2]^\top$ denotes the Gaussian distribution inputs; and $[e_1, e_2, e_3]^\top$ is an array of zero-mean white noises with the standard deviations of 0.01. The system's normal operating condition was simulated with the inputs $s_1 : N(10, 0.8)$ and $s_2 : N(12, 1.3)$, where $N(.)$ is to denote the Gaussian distribution given the mean and the variance. 1000 samples of both state variables and inputs were recorded as a set of training data. To achieve the monitoring, CH-FD was first trained by computing a 5-dimensional convex hull of all training samples with Qhull. The corresponding envelope of 5 monitored variables was then plotted based on the axes order as x_1, x_2, x_3, s_1 and s_2, as shown in Figure 3.18. Clearly, all polyplines representing the normal samples were accurately contained by the envelope which formed the geometric pattern of the system's normal operating regime.

In the testing dataset, the first 200 samples were generated based on the normal operating condition. At the 201$^{\text{st}}$ sample, a change in input variables was triggered as $s_1 : N(5, 0.6)$ and $s_2 : N(20, 0.7)$ to imitate the situation where an unexpected system mode transition appeared. For the fault detection experiment, CH-FD was first performed with each sample to generated a detection result. When presenting

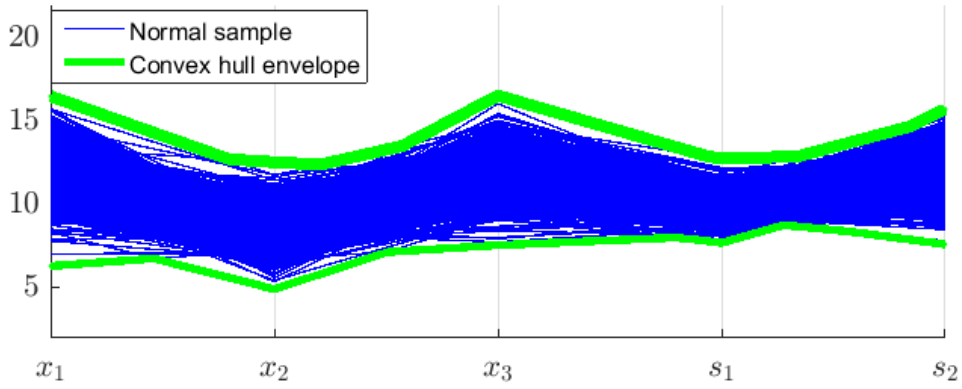


Figure 3.18: Illustrative example II – Training samples and convex hull envelope

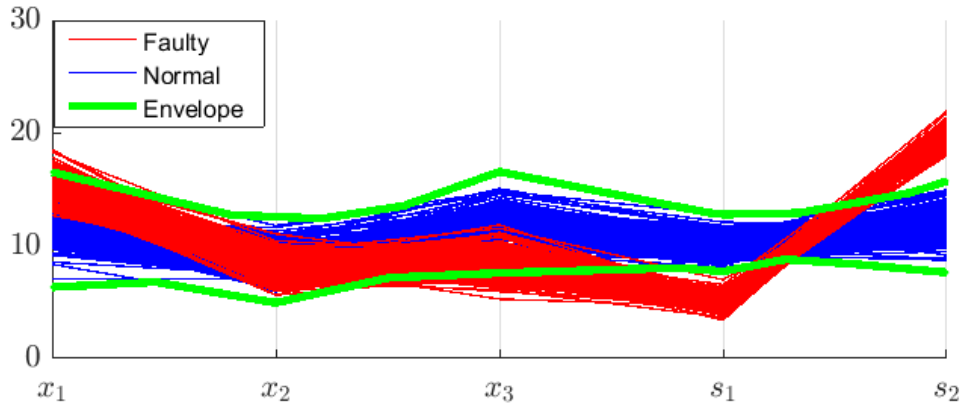


Figure 3.19: Illustrative example II – Testing samples with the convex hull envelope.

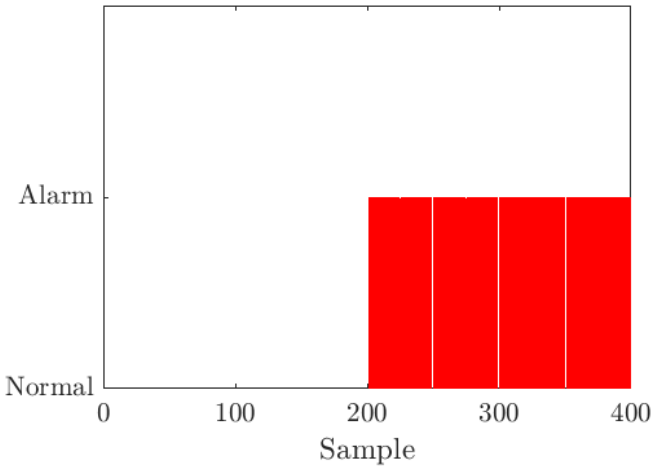


Figure 3.20: Illustrative example II – Alarm plot of CH-FD results.

a sample on the parallel coordinates plot, the color of its ploryline was depended on its detection result, i.e. the blue color indicated a normal sample, and the red color showed a fault. To display the time information, the detection results of CH-FD were also presented as a discrete time binary alarm trend on a plot.

According to the detection results presented by the parallel coordinates plot in Figure 3.19, two separate clusters of ploylines could be clearly observed. The blue cluster representing the normal samples was perfectly bounded by the convex hull envelope. In contrast, the ploylines in the red cluster violated the normal pattern in multiple sections. Especially, for those line segments in between axes x_3, s_1, and s_2,

their interceptions with the normal envelope clearly indicated the change of system operating point. From red line segments between other axes, mean shifting of x_1 and x_2 readings could also be observed. Notice the ploylines of testing samples were plotted in the time order of the dataset, i.e, the ploryline of newest sample was always above the previous ones. Hence, the red cluster was above the blue cluster, which is consistent with the alarm plot of CH-FD shown in Figure 3.20.

From this simple example, the parallel coordinates based visualization would allow us better inspect the detection results of CH-FD. When comparing the geometric pattern of faulty samples with the normal envelope, more information of system abnormal behavior could be directly visualized on the plot.

Chapter 4

Industrial Case Studies

4.1 Overview

In this chapter, the proposed convex hull based fault detection (CH-FD) is applied in two industrial benchmark processes to demonstrate the detection performance. The first case study utilizes the simulation of a continuous stirred tank heater (CSTH) to validate the performance in monitoring a typical process unit. For the second case study, further testings are conducted with the benchmark plant from the Tennessee Eastman (TE) challenge problem. To better assess the performance in both examples, the detection results from the methods discussed in chapter 2 are collected to draw the comparison analysis.

4.2 Case I: Continuous stirred tank heater

The conterminous stirred tank heater (CSTH) simulator developed by Thornhill [88] is a hybrid simulation which utilized the real disturbance data captured from a pilot plant to drive the first principle model. As depicted by the schematic in Figure 4.1, the stirred tank is supplied with both hot water (HW) and cold water (CW), and further heated with a steam coil. The three process inputs are the HW, CW and steam valve position. The process outputs, including the tank level, the water flow rates and the temperature, are all controlled by PI loops. The measurements from all sensors are presented as electrical signals on a 4-20mA scale. The disturbances

of the level, CW valve position and temperature are provided with 2500 samples of real data collected from a pilot plant. More details of the CSTH model could be found in [88], and the MATLAB version of the simulation is available at `http://personal-pages.ps.ic.ac.uk/~nina/CSTHSimulation/index.htm`.

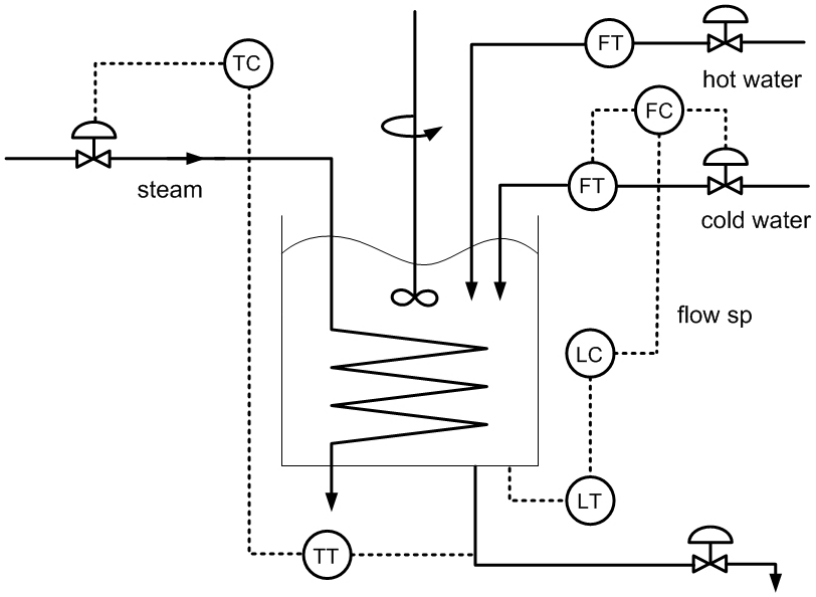


Figure 4.1: CSTH configuration

4.2.1 Data preparation

In this experiment, two sets of simulated data had been generated according to the two normal operating conditions listed in Table 4.1. Notices the tank is operating without HW under the first condition. For our fault detection experiments, three process variables, including the level, CW flow and temperature, were chosen to be monitored. In each operating condition, 2500 normal samples were collected with a sample time of 1 sec. The first 2000 samples were used for modeling, and the rest 500 samples were used in validations.

4.2.2 Model training

For this example, we compare CH-FD with PCA and 1-class SVM. When training the models, PCA took two PCs to capture 80% of total variance of the training

Table 4.1: A list of CSTH normal operating points

Variable	Operate point 1	Operate point 2
Level/mA	12.00	12.00
CW flow/mA	11.89	7.330
CW valve/mA	12.96	7.704
Temperature/mA	10.50	10.50
HW valve/mA	0	5.5

Table 4.2: A list of injected faults in CSTH simulation.

Fault	Sensor & actuator	Type	Intensity
1	Level	Step bias	-1 mA
2	Temperature	Step bias	0.5 mA
3	Temperature	Ramp bias	0.01 mA/sample + e
4	Temperature	Ramp bias	0.001 mA/sample + e
5	CW valve	Sticking	2 %
e is a random Gaussian disturbance with $\mu = 0$ and $\sigma = 0.01$			

samples. 1-class SVM was built with a rbf kernel where the scaling factor $\gamma = 1$. The classifier was tuned to contain 99% of the training samples. In CH-FD, the convex hull of training data was generated by Qhull, where the resulted FAR of the validation samples was below 2%. For detection metric of PCA and 1-class SVM, the empirical thresholds were set as the value of 10^{th} highest metrics in the validation sets to meet the 99% confidence level.

In our fault detection tests, five faults were designed and implemented according to the detail descriptions listed in Table 4.2. The disturbance types included sensor step bias and valve stiction. For each fault, the simulation triggered the disturbance after 200 sec (200 samples), and then kept running for another 200 sec. As the plant is controlled by closed loop PI controllers, all faults were expected to be mitigated after a time period. For two different operating conditions, the process behavior was also expect to vary. For example, the temperature bias was easier to suppress when

HW was not supplied. In the analysis, performance measures used in this study were the fault detection rate and the detection latency.

4.2.3 Detection results

The results of detection rates from all three methods are tabulated in Table 4.3, and the maximum values have been highlighted in bold face. Among all fault classes, 1-class SVM generally performed better than PCA as expected, due to the fact that CSTH is a non-linear process. With the aid of an rbf kernel, the classifier of 1-class SVM clearly had the advantage in recovering non-linear correlations between process variables. It is also clear to observe that CH-FD was able to obtain better detection rates than 1-class SVM, except faults 1 and 3 at Operate point 2 where it only generated 1 less successful detection. For faults 4 and 5 at both operating conditions, CH-FD had almost improved the detection rates by around 10%.

Table 4.4 lists the detection latencies of all three methods, and again the minimal delay values have been highlighted in bold face. According to the results of faults 1 and 2, all three methods are able to detect step bias right after is has been triggered.

Table 4.3: CSTH results – detection rates (in %)

	Operate point 1 test data			
Fault	PCA – T^2	PCA – Q	1-class SVM	CH-FD
1	44	36.5	50	**50.5**
2	13.5	**16.5**	13.5	**16.5**
3	62	75	78	**83**
4	19.5	23	27	**34.5**
5	32.5	12.5	31.5	**45.5**
	Operate point 2 test data			
Fault	PCA – T^2	PCA – Q	1-class SVM	CH-FD
1	67	48.5	**76**	75.5
2	32	31.5	37.5	**39**
3	89	90	**92**	91.5
4	27.5	23	27	**43.5**
5	19.5	7	23	**30.5**

Table 4.4: CSTH results – detection latencies (in sec)

	Operate point 1 test data			
Fault	PCA – T^2	PCA – Q	1-Class SVM	CH-FD
1	**1**	**1**	**1**	**1**
2	**1**	**1**	**1**	**1**
3	11	12	7	**3**
4	12	**3**	**3**	**3**
5	3	14	12	**2**
	Operate point 2 test data			
Fault	PCA – T^2	PCA – Q	1-class SVM	CH-FD
1	**1**	**1**	**1**	**1**
2	**1**	**1**	**1**	**1**
3	**7**	15	**7**	**7**
4	**7**	10	**7**	**7**
5	**9**	41	20	13

For the ramp bias in faults 3 and 4, all methods detected the abnormality significantly slower, because the disturbance would take a considerable time to wind up till the controllers were no-longer capable of regulating the effect. Similarly, the operating point oscillation caused by the CW valve stiction would emerge slowly. Thus, the large delay numbers of fault 5 were presented by PCA-SPE and 1-class SVM. Among results of different faults, CH-FD all provided the minimal sample delay, except the case of fault 5 at Operate point 2.

Considering the good results in both detection rates and detection latency, such competitive performance confirms that the convex hull of normal data captures more complicate process normal operating regimes. When an abnormal event introduces a shifting of process operating points, the detection setup of this new approach can quickly and accurately identify the faulty samples. Thus, CH-FD should be consider as an effective fault detection approach for small scale systems.

4.2.4 Fault 5: CW valve 2% stiction

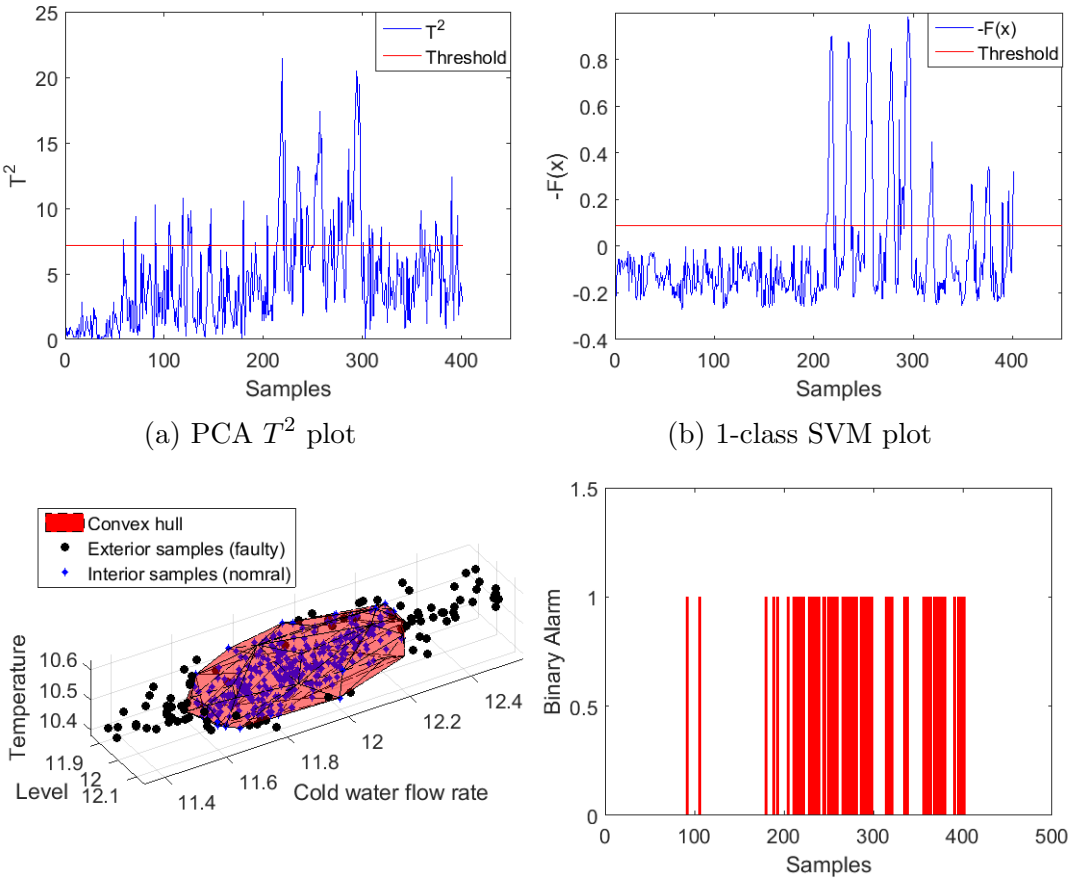
The so-called stiction is one of the most commonly found valve problem in industrial processes. Due to the phase lag between the control signal and the actual flow rate, the presence of stiction in valves always causes process oscillations and degrades the control loop performance. In the existing literature, a number of studies have been conducted focusing on this problem, and many researchers dedicated to properly model and effectively detect the stiction phenomena [89]. As mentioned before, fault 5 in this example was to imitate a 2% stiction happened to the CW control valve by injecting a simple deadband with a magnitude of 0.34mA (equivalent to 2% of total valve scale 4-20mA).

As the test data depicted in Figure 4.2c, the deadband triggered a serious oscillation in both the CW flow rate and the tank temperature. Compared to the normal operation regime defined by the red convex hull, the cluster of test data was signification enlarged by the faulty samples laid outside the hull. Thus, in the alarm plot shown in Figure 4.2d, this fault could be easily detected by CH-FD, and the alarm pattern indicates the oscillation persisted till the end of the monitoring period.

As for PCA, the T^2 statistic was able to report detections at the first 100 samples once the fault was triggered. For the last 100 samples, the metric values could not be distinguished from normal ones. This might due to the fact that the linear correlation among the three variables still held, even if the data samples were oscillatory. Similarly, the metric values of 1-class SVM clearly presented the oscillation. However, the high empirical threshold was unable to separate most faulty samples from the normal ones. This particular example well demonstrates the effectiveness of CH-FD in detecting valve stiction. The convex hull based detection metric is relatively more sensitive to the faulty samples because it is free of distribution assumptions.

4.3 Case II: Tennessee Eastman process

The Tennessee Eastman (TE) challenge problem proposed by Downs and Vogel [90] is a popular benchmarking platform for chemical process control and monitoring studies.



(a) PCA T^2 plot

(b) 1-class SVM plot

(c) CH-FD data space monitoring plot

(d) CH-FD alarm plot

Figure 4.2: CSTH demo – detection result of valve stiction

In this section, the updated version of the TE simulator provided by Bathelt [91] is utilized to demonstrate the application of CH-FD in monitoring a complex plant.

Figure 4.3 depicts a basic schematic diagram of the process with five major units, including the reactor, condenser, stripper, separator and compressor. The process has four input reactants (A, D, E and C) to produce two main products as well as a inert and a byproduct. The plant operates with 52 measurements in total, including 22 continuous process variables, 11 manipulated variables and 19 sampled analyzer measurements which are listed in Tables 4.5 to 4.7. To stabilize this process, the plant-wide control structure developed by Richer is implemented [92]. For the process monitoring benchmarking, the 20 faults originally defined by Downs and Vogel listed

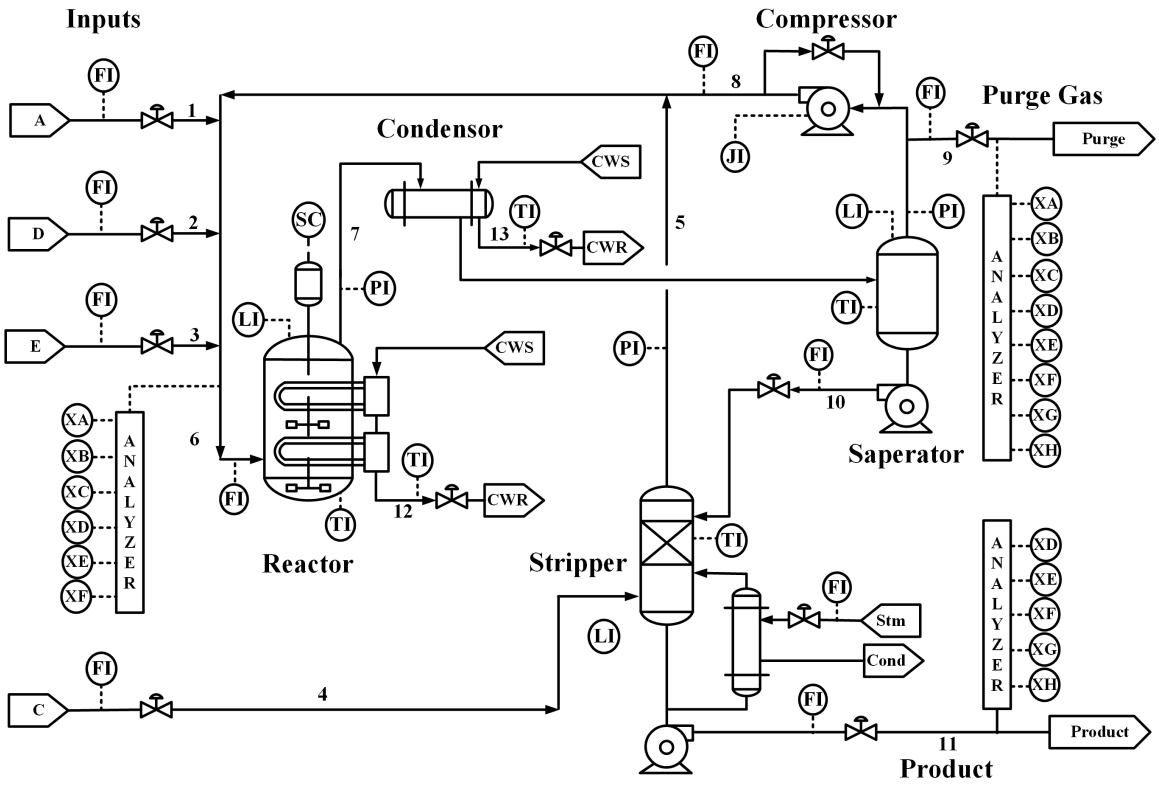


Figure 4.3: Tennessee Eastman process P&ID

in Table 4.8 are utilized [90].

For more information of the TE process model, readers could refer to [90]. More details of the simulator could be found in [91], and the MATLAB version is available to download at `http://depts.washington.edu/control/LARRY/TE/download.html`.

4.3.1 Data preparation

In this case study, our datasets were prepared by collecting the samples from 52 variables with a sample time of 3 minutes. For the training set, measurements were recorded for 1200 hours of normal operation to generate 24000 samples. For the fault detection benchmarking, 20 testing sets according to faults listed in Table 4.8 were generated. Each testing set consisted of 48 hours of operation record (960 samples), and the fault was introduced after 8 hour normal operation (160 samples).

Table 4.5: TE process – list of continuous process variables

Variable name	Tag	Base case value	Units
A feed (stream 1)	XMEAS(1)	0.25052	kscmh
D feed (stream 2)	XMEAS(2)	3664.0	kg h^{-1}
E feed (stream 3)	XMEAS(3)	4509.3	kg h^{-1}
A and C feed	XMEAS(4)	9.3477	kscmh
Recycle flow	XMEAS(5)	26.902	kscmh
Reactor feed rate	XMEAS(6)	42.339	kscmh
Reactor pressure	XMEAS(7)	2705.0	kPa gauge
Reactor level	XMEAS(8)	75.000	%
Reactor temperature	XMEAS(9)	120.40	°C
Purge rate	XMEAS(10)	0.33712	kscmh
Separator temperature	XMEAS(11)	80.109	°C
Separator level	XMEAS(12)	50.000	%
Separator pressure	XMEAS(13)	2633.7	kPa gauge
Separator underflow	XMEAS(14)	25.160	kg h^{-1}
Stripper level	XMEAS(15)	50.000	%
Stripper pressure	XMEAS(16)	3102.2	kPa gauge
Stripper underflow	XMEAS(17)	22.949	m^3 h^{-1}
Stripper temperature	XMEAS(18)	22.949	°C
Stripper steam flow	XMEAS(19)	230.31	kg h^{-1}
Compressor work	XMEAS(20)	341.43	kw
Reactor water temperature	XMEAS(21)	94.599	°C
Separator water temperature	XMEAS(22)	77.297	°C

Table 4.6: TE process – list of manipulated variables

Variable name	Tag	Base value(%)	Units
D feed flow	XMV(1)	63.053	kg h^{-1}
E feed flow	XMV(2)	53.980	kg h^{-1}
A feed flow	XMV(3)	24.644	kscmh
A and C feed flow	XMV(4)	61.302	kscmh
Compressor recycle valve	XMV(5)	22.210	%
Purge valve	XMV(6)	40.064	%
Separator pot liquid flow	XMV(7)	38.100	m^3 h^{-1}
Stripper liquid product flow	XMV(8)	46.534	m^3 h^{-1}
Stripper steam valve	XMV(9)	47.446	%
Reactor cooling water flow	XMV(10)	41.106	m^3 h^{-1}
Condenser cooling water flow	XMV(11)	18.114	m^3 h^{-1}

Table 4.7: TE process – list of sampled analysis variables

Analyzer unit	Variable name	Tag
Reactor feed analysis (stream 6)	Component A	XMEAS(23)
	Component B	XMEAS(24)
	Component C	XMEAS(25)
	Component D	XMEAS(26)
	Component E	XMEAS(27)
	Component F	XMEAS(28)
Purge gas analysis (stream 9)	Component A	XMEAS(29)
	Component B	XMEAS(30)
	Component C	XMEAS(31)
	Component D	XMEAS(32)
	Component E	XMEAS(33)
	Component F	XMEAS(34)
	Component G	XMEAS(35)
	Component H	XMEAS(36)
Product analysis (stream 11)	Component D	XMEAS(37)
	Component E	XMEAS(38)
	Component F	XMEAS(39)
	Component G	XMEAS(40)
	Component H	XMEAS(41)

4.3.2 Model training

CH-FD training

In order to achieve an effective on-line monitoring for the TE process, the applied method should be able to detect any abnormal deviation appearing in the 22 process variables and the 11 manipulated variables. Hence, in the training stage of CH-FD, we aim to capture the normal operating regime of all 33 monitored variables. Such a large number of data dimensions is indeed overwhelming for the convex hull computation. The associated challenges are mainly two folded. First, when the data dimension kept increasing, the sets of facets and hyperplane functions would also be enlarged exponentially. With limited storage and computing power, Qhull could only handle the data up to 9 dimensions [74]. Thus, to directly construct a convex hull with all the monitored variables was not feasible. Secondly, even if we were able to push the algorithm to its limit, the resulting high-dimensional convex hull would also

Table 4.8: TE process – list of faults

Fault	Description	Type
1	A/C feed ratio, B composition constant (stream 4)	Step
2	B composition, A/C ratio constant (stream 4)	Step
3	D feed temperature (stream 2)	Step
4	Reactor cooling water inlet temperature	Step
5	Condenser cooling water inlet temperature	Step
6	A feed loss (stream 1)	Step
7	C header pressure loss-reduced availability (stream 4)	Step
8	A, B, C feed composition (stream 4)	Random
9	D feed temperature (stream 2)	Random
10	C feed temperature (stream 4)	Random
11	Reactor cooling water inlet temperature	Random
12	Condenser cooling water inlet temperature	Random
13	Reaction kinetics	Slow Drift
14	Reactor cooling water valve	Sticking
15	Condenser cooling water valve	Sticking
16	Unknown	Unknown
17	Unknown	Unknown
18	Unknown	Unknown
19	Unknown	Unknown
20	Unknown	Unknown

suffer a high FAR due to the 'Curse of Dimensions'.

To overcome this barrier of data dimensionality, we came up with following strategies to implement the CH-FD:

- **Variable localization**: All process variables and manipulated variables are allocated into different groups according to the layout of the main operating units and the major steam flows in the TE plant. Each group is then considered as a sub-unit consisting no more than 8 variables and would be monitored independently.

- **Recursive training**: A convex hull of each sub-unit would be trained and cross validated with recursively enlarged training sets, till the FAR of the validation samples converges to a desired value.

Table 4.9: TE process CH-FD – monitored variables partition

Reactor Inputs (6D)		
Variable	Steam	Tag
A feed	1	XMEAS(1)
A feed flow	1	XMV(3)
D feed	2	XMEAS(2)
D feed flow	2	XMV(1)
E feed	3	XMEAS(3)
E feed flow	3	XMV(2)
Reactor PVs (6D)		
Variable	Steam	Tag
Reactor feed rate	6	XMEAS(6)
Reactor pressure	7	XMEAS(7)
Reactor level		XMEAS(8)
Reactor temperature		XMEAS(9)
Reactor cooling water outlet temperature	12	XMEAS(21)
Reactor cooling water flow	12	XMV(10)
Separator PVs (5D)		
Variable	Steam	Tag
Separator pot liquid flow	10	XMV(7)
Product separator underflow	10	XMEAS(14)
Product separator pressure		XMEAS(13)
Product separator level		XMEAS(12)
Product separator temperature		XMEAS(11)
Stripper PVs (6D)		
Variable	Steam	Tag
A and C feed flow	4	XMV(4)
A and C feed	4	XMEAS(4)
Stripper pressure	5	XMEAS(16)
Stripper level		XMEAS(15)
Stripper temperature		XMEAS(18)
Stripper steam flow		XMEAS(19)
Product Outlet (2D)		
Variable	Steam	Tag
Stripper underflow	11	XMEAS(17)
Stripper liquid product flow	11	XMV(8)
Condenser PVs(2D)		
Variable	Steam	Tag
Condenser cooling water outlet temperature	13	XMEAS(22)
Condenser cooling water flow	13	XMV(11)

Compressor PVs(2D)		
Variable	Steam	Tag
Recycle flow	8	XMEAS(5)
Compressor work		XMEAS(20)
Purge Gas (2D)		
Variable	Steam	Tag
Purge rate	9	XMEAS(10)
Purge valve	9	XMV(6)

Based on the TE process schematic diagram in Figure 4.3, 8 monitoring units were introduced, including 5 operation units: reactor, condenser, stripper, separator and compressor, as well as three main stream flows: reactor inputs, product outlet and purge gas. Each sub-unit consisted of all related variables listed in Table 4.9. Notice that in the Richer's control scheme, the compressor recycle valve and stripper steam valve (XMV(5) and XMV(9)) were not controlled and held at their optimum set points. Thus, these two manipulate variables were not consider in this implementation. Otherwise, this monitoring scheme covered all 22 process variables and 9 manipulate variables.

Table 4.10: TE process CH-FD – FARs of convex hull in each monitoring unit

Sub-units	90 h	180 h	300 h	600 h	900 h	1200 h
Reactor inputs (6D)	19%	11%	8%	4%	4%	3%
Reactor (6D)	22%	15%	14%	8%	5%	5%
Separator (5D)	12%	7%	5%	5%	4%	2%
Stripper (6D)	22%	13%	13%	10%	7%	5%
Condenser (2D)	1%	0%	0%	0%	0%	0%
Compressor (2D)	0%	0%	1%	1%	0%	0%
Purge gas (2D)	0%	0%	0%	0%	0%	0%
Product outlet (2D)	1%	0%	0%	0%	0%	0%

When constructing the convex hull for each sub-unit, the normal datasets were gradually enlarged from 90 hour to 1200 hour length. In each set, the first 2/3 samples were loaded to Qhull for convex hull computation. The remaining 1/3 samples

were used for cross validation. The resulted FARs for all sub-units are tabulated in Table 4.10. It is clear to observe that the high-dimensional convex hulls required significantly more training samples to stabilize the FAR. When the normal operation was extended to 1200 hour, the FARs of all sub-units converged below to somewhere below 5%.

The convex hull envelopes of all sub-units were then plotted on parallel coordinates as shown in Figure 4.4. To provide a better presentation, the mean centered and standardized training samples were utilized to generate those greened envelopes with the graphical method presented in Chapter 3. Based on the detection results of CH-FD, the validation samples had been projected on the parallel coordinate plots in different colors, i.e, normal samples were in blue and faults were in red. It is clear to observe that most of the normal samples were contained by the envelopes, and only few false positive samples appeared. Thus, for later fault detection tests, the convex hulls derived from the data of 1200 hour normal operation were utilized as the monitoring models, and the corresponding parallel coordinates envelopes were considered as the visualization references.

Training of the other methods

To analyze the performance of CH-FD, a comparison study was made with PCA, PLS and 1-class SVM. When training these methods, the modeling and validation were all conducted with the 1200 hour training data. For PCA, 29 PCs were retained to capture 95% of the total variance of the training samples. When training PLS, the model considered the sampled analysis variable XMEAS(35) as the output, and used 29 LVs to extract 95% of the total linear correlations. The thresholds of T^2 and SPE metrics, in both PCA and PLS, were calculated with a 99% confidence level. The classifier of 1-class SVM was also built to capture 99% of normal samples with a rbf kernel. The kernel scaling factor γ was set as 1. The detection threshold was left at the default value ($-F(\mathbf{x}) = 0$) based on the resulted classifier function.

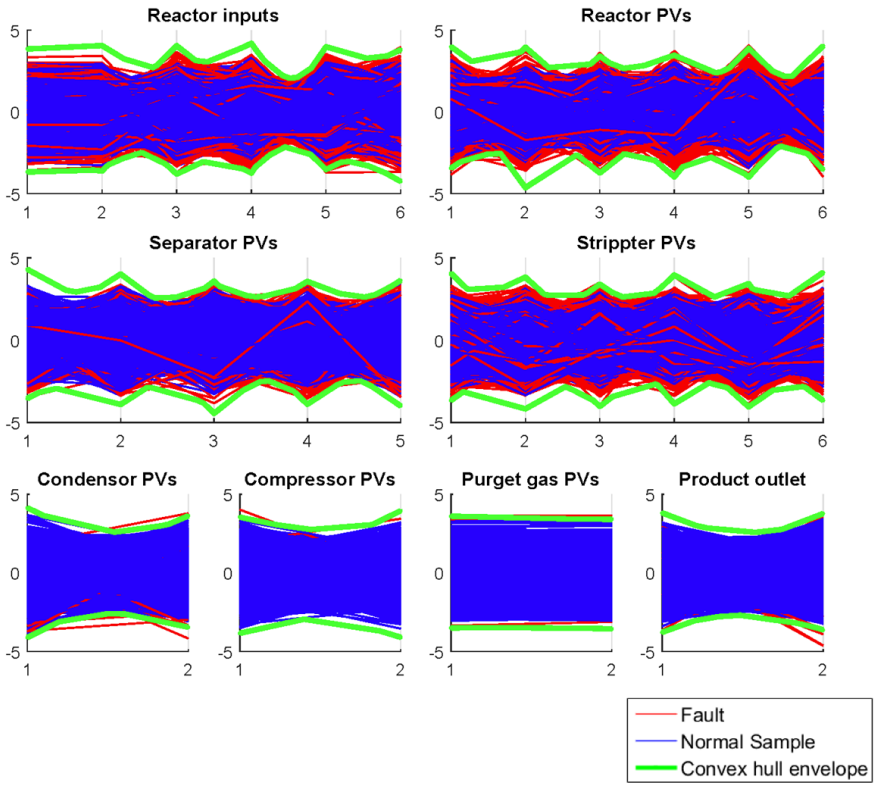


Figure 4.4: TE process CH-FD – visual validation on parallel coordinates

4.3.3 Detection results

Similar to the previous case study, the fault detection rates and the detection delays are the two performance measurements that we use to evaluate all 4 methods. According to the previous work by Russel in [44], faults 3, 9 and 15 are unable to cause any signification deviation to the steady state operating point. Thus they are generally hard to be detected by existing data-driven methods. Later, Yin also reported that fault 16 had low detection rates among all methods tested in [51], particularly when the TE simulation was implemented with the control structure by Richer [92]. According to our experiment, CH-FD was also unable to detect these faults. Thus, in the following analysis, the results of faults 3, 9, 15 and 16 are not included.

Detection delays

In this case study, the detection delay was recorded as the time duration between the instance when a fault was triggered and the first detection sample of each method. The recorded delays are tabulated in Table 4.11, where the minimum delay of each fault is presented in bold face. According to this set of results, PLS and 1-class SVM were the two fastest detection methods for step disturbances (in faults 1-7), where they only introduced one sample delay except faults 1 and 2. However, the SPE metric of PLS presented large delay numbers in most of the faults. This is because the residuals of the PLS model which only carries limited information of the variables correlations; and thus the SPE metric would be less sensitive to the deviations appeared in the data. CH-FD and PCA also detected these faults quickly with few more delay samples. In detecting random disturbances for faults 8-12, CH-FD and 1-class SVM showed clear advantages. In the remaining faults, these two methods also demonstrated their superiority over PCA and PLS. Particularly for

Table 4.11: TE process results – the comparison of detection delays (in min)

IDV	CH-FD	PCA-T^2	PCA-Q	1-class SVM	PLS - T^2	PLS - SPE
1	9	9	6	**3**	12	66
2	15	21	12	12	**3**	270
4	6	6	6	**3**	**3**	270
5	6	**3**	**3**	**3**	**3**	9
6	6	6	6	**3**	**3**	24
7	6	6	6	**3**	**3**	9
8	**27**	42	54	**27**	60	234
10	**33**	42	39	**33**	39	N/A
11	**15**	**15**	**15**	**15**	**15**	**15**
12	**12**	27	72	**12**	60	N/A
13	**39**	75	75	75	75	129
14	6	9	12	**3**	**3**	33
17	**174**	189	177	183	177	249
18	**213**	240	222	223	228	354
19	6	12	12	**3**	12	90
20	135	138	129	**123**	123	168

fault 13, the delay time was significantly reduced by CH-FD.

From this set of results, the advantage of CH-FD in capturing an accurate and strict process normal operating regime could be further validated, even when part of the correlation was disregarded as monitored variables were localized into different sub-units. On the other hand, the superiority of 1-class SVM in detecting process non-linearity and local behavior deviations [48] was again confirmed. Both of these two methods seem to be very competitive. For PCA and PLS, their performance were also appealing. However, due to the fact that their detection metrics were designed based on the assumption of linear Gaussian distributions, they only had advantages in quickly detecting simple step bias disturbances.

Detection rates

Table 4.12: TE process results – the comparison of detection rates (in %)

IDV	CH-FD	PCA - T^2	PCA - SPE	1-class SVM	PLS - T^2	PLS - SPE
1	99.8	99.8	99.9	99.9	**100**	97.5
2	99.5	99.1	99.6	**99.9**	99.8	75.7
4	99.8	99.9	99.9	99.9	**100**	37.2
5	99.8	99.9	99.9	99.9	**100**	99.7
6	99.3	99.3	99.3	99.9	**100**	95.1
7	99.8	99.8	99.8	99.9	**100**	99.8
8	**98.3**	97.8	97.43	98.1	98.0	29.0
10	95.4	72.9	92.3	84.8	**96.1**	0.12
11	99.5	99.1	98.4	**99.6**	99.5	68.0
12	**69.0**	40.9	19.7	47.8	54.4	0.12
13	97.1	97.1	97.1	97.0	**97.3**	25.09
14	**99.9**	99.6	99.5	99.6	99.9	65.0
17	92.8	91.4	92.9	92.6	**93.0**	44.01
18	**81.1**	69.5	78.9	72.3	80.9	7.87
19	**99.6**	98.4	98.5	99.0	**99.6**	27.3
20	94.5	94.5	94.9	**95.3**	**95.3**	14.7

The results are again tabulated in Table 4.12, and the maximum detection rate of each fault is presented in bold face. As our training sets consist of significantly

more samples than other TE datasets, which were applied in [44, 48, 51], the trained model in each method would be more accurate and robust. So, better results on the detection rates were expected from all 4 methods. For the step disturbances in faults 1-7, all methods had high detection rates which were all above 99%. Similar to the results on the detection delay, 1-class SVM and PLS T^2 had some slight advantages, while CH-FD and PCA almost had identical performance. When detecting random disturbances in faults 8-12, the best detection rates were shared among CH-FD, 1-class SVM and PLS T^2. Particularly in the test of fault 12, CH-FD had a signification higher detection rate over other methods. For the rest of the fault list, CH-FD and PLS T^2 were the top two candidates which shared most of the best results.

The above results basically were consistent with what was observed in the earlier comparison of the detection delays. However, CH-FD did not clearly stand out for the best detection rate. This might due to the fact that part of correlation was ignored when we separated the variables into different monitoring units. For example, the flow of all reactants would directly affect the input flow of the reactor, but their measurements were in different monitoring units, so that the correlation could not be captured by the two separate convex hulls.

Fault detection visualization

During the experiments, faulty samples in each testing set were also projected onto the separated parallel coordinates plots with the normal operating envelopes. It was interesting to observe that our plots showed clear advantages in visualizing and identifying the top bad variables for certain simple faults. Taking fault 10 as an example, the disturbance was introduced by the random variance of C feed temperature. According to the parallel coordinates visualization shown in Figure 4.5, the 5th variable on the plot of the stripper unit could be easily identified as the worst upset variable where the geometric pattern presented clear oscillation. Recall our variable partition, which is listed in table 4.9, this variable is exactly the stripper temperature which directly reflected the root cause of the fault. In addition, similar oscillation patterns could also be observed from the 1st variable of the separator unit and the 2nd variable

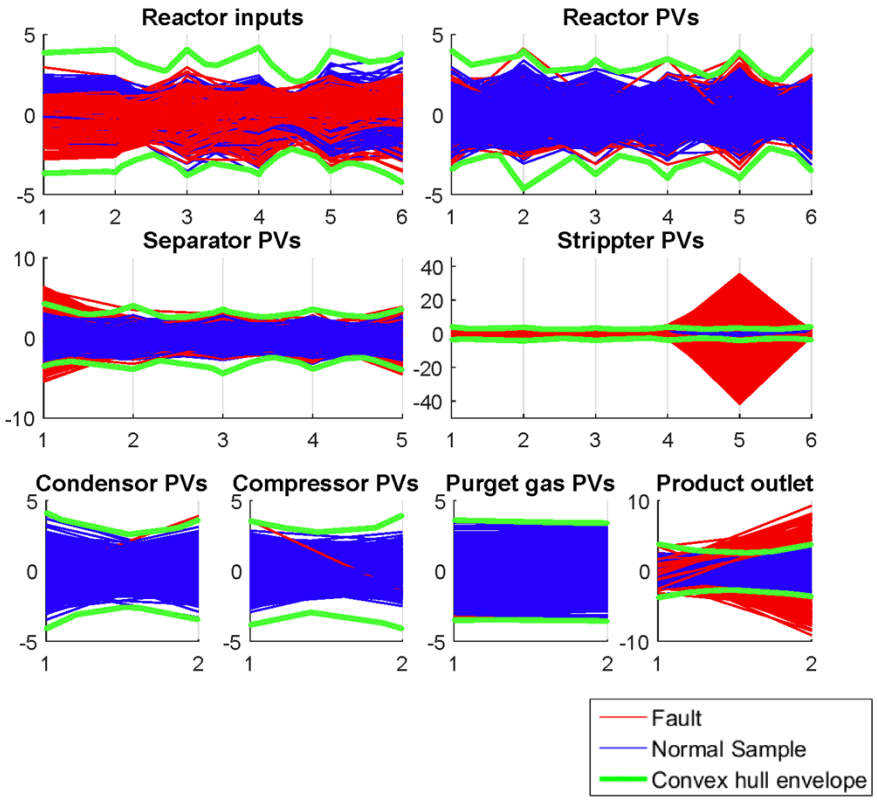
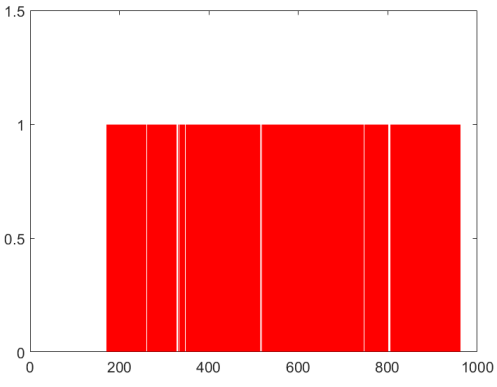
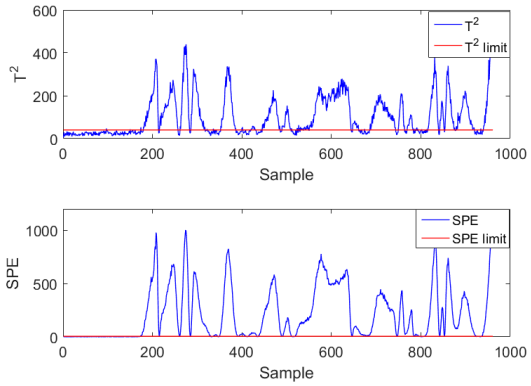


Figure 4.5: TE process – presenting fault 10 on parallel coordinates

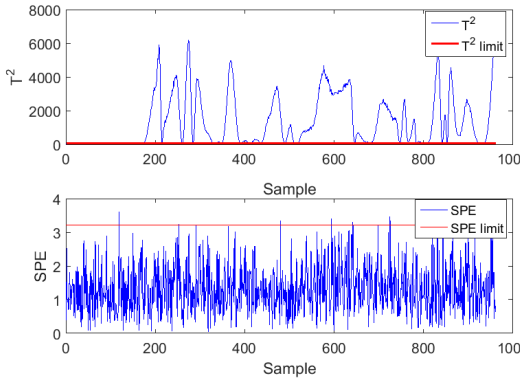
of product outlet unit. These variables are the valve control signals of the input steam and the outlet steam of the stripper unit. From their oscillation patterns, we could conclude that the control was trying to compensate the temperature disturbance by tuning the steam flow rates of the stripper units. When referring to the alarm plot of CH-FD shown in Figure 4.6a, the consistent alarm announcing period throughout the faulty samples also conformed the successful detection of the disturbance. While on the plots of PCA, PLS and 1-class SVM presented in Figures 4.6b to 4.6d, the detection metrics all showed clear dropping periods indicating the oscillation of the faulty variables. However, they can not provide sufficient information to allow the user to locate the source of abnormality.



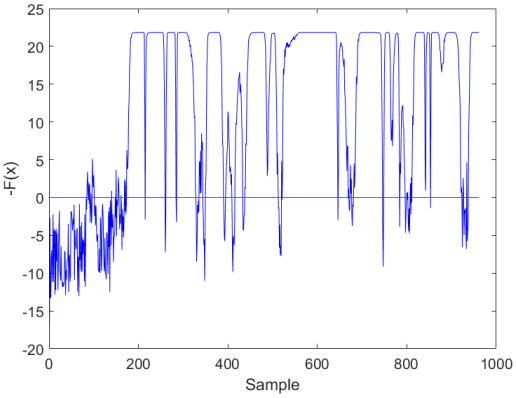
(a) Fault 10 CH-FD alarm plot



(b) Fault 10 PCA metrics



(c) Fault 10 PLS metrics



(d) Fault 10 1-class SVM metrics

Figure 4.6: TE process – detection results of fault 10

Chapter 5

Concluding Remarks

5.1 Contributions

In modern process industries, the critical requirements of operating safety and high demands of production efficiency are continuously driving the advances in process monitoring. Over the last few decades, multivariate data-driven techniques have an emerging success due to their relative low-cost implementation and good performance. Today, a variety of different statistical or machine learning tools have been introduced in this field and proven to be effective in detecting process abnormal behavior. However, most of these methods have been developed based on their prior assumptions of process data distributions. In real industrial practice, the process data collected during on-line operation may not present a clear distribution pattern. When applying these data-driven techniques in a real monitoring scheme, their reliability may be hard to guarantee. This thesis dedicates to seeking for a feasible solution to this potential issue. As a result, the idea of using convex hulls as an assumption free detection metrics is successfully adapted for the process monitoring design, and a parallel coordinates based visualization tool is introduced to facilitate the presentations of detection results. The major contributions through the content of this thesis are summarized as follows.

In chapter 2, a study of three iconic data-driven monitoring techniques, including PCA, PLS and 1-class SVM, are first presented to discuss the common preliminaries and design concepts of data-driven methods. Next, we introduce the convex hull based

fault detection method. In this new method, the Quickhull algorithm is utilized to construct the convex hull of collected normal training data; and a detection metric based on the hyperplane equations of the convex hull is then designed.

In chapter 3, we investigate for a better visual presentation of the real time monitoring results. To overcome the dimensional limitation of scatter plots in the Cartesian space, we present a high dimensional process visualization based on parallel coordinates. First, basic background and preliminary of the duality between Cartesian space and parallel coordinates are provided. To better facilitate our newly proposed fault detection method, the way of reconstructing convex hull envelopes in parallel coordinates are later introduced. A simple example is demonstrated to show how faulty samples could be visually detected with this visualization setup.

In order to further validate the performance of convex hull based fault detection, the testing results with two simulated industrial case studies are presented in chapter 5. Our results from the CSTH example manifest the superior performance of this new method in detecting faults caused by random disturbances and valve stiction. In the second example of the TE benchmark process, we demonstrate how to implement this new method with a large scale process. Although it is hard to overcome the 'curse of dimensionality', once the convex hulls are fed with sufficient training data, the detection results from this new method are also competitive comparing to existing data-driven techniques.

5.2 Scope of the future work

- Due to the 'curse of dimensionality', the proposed convex hull based detection technique clearly has the limitation of its high false alarm rates especially when dealing with high-dimensional process data. This may directly limit the applications of this method in monitoring large scale systems. This issue should be further addressed in the future study. Certain tools from the field of industrial alarm monitoring may be adopted as good solutions. For example, data pre-filtering and delay timers could be combined with the fault detection de-

sign where high false alarm rates of the convex hull metric may be effectively suppressed.

- In the setup of our parallel coordinates based visualization, how to effectively reveal the time information of the monitored process data is still a challenging topic. The solution of this problem may lead us to investigate the multivariate causality analysis on the parallel coordinates. In addition, with different axes orders, the patterns formed by the same set of data would vary on parallel coordinates. Hence, how to effectively order the variable axes to highlight the information of variables dependency is also an interesting topic worthy further investigation.

Bibliography

[1] L. H. Chiang, R. D. Braatz, and E. Russell, *Fault Detection and Diagnosis in Industrial Systems*. Springer, 2001.

[2] I. Izadi, S. Shah, D. Shook, and T. Chen, "An introduction to alarm analysis and design," in *Fault Detection, Supervision and Safety of Technical Processes*, 2009, pp. 645–650.

[3] B. R. Land, "Parallel coordinate graphics using MATLAB," https://people.ece.cornell.edu/land/PROJECTS/Inselberg/, Copyright Cornell University, 2001, Accessed: 2016-05-19.

[4] "Abnormal situation management consortium, impact," http://www.asmconsortium.net/defined/impact/Pages/default.aspx, accessed: 2016-05-19.

[5] R. Isermann and P. Ball, "Trends in the application of model-based fault detection and diagnosis of technical processes," *Control Engineering Practice*, vol. 5, pp. 709 – 719, 1997.

[6] P. Kesavan, , and J. H. Lee, "Diagnostic tools for multivariable model-based control systems," *Industrial & Engineering Chemistry Research*, vol. 36, no. 7, pp. 2725–2738, 1997.

[7] AIChE/CCPS, *Guidelines for Engineering Design for Process Safety*. New York-Center for Chemical Process Safety, New York, 2012.

[8] N. A. Adnan, "Performance assessment and systematic design of industrial alarm systems," Ph.D. dissertation, University of Alberta, Edmonton, AB, Canada, 2013.

[9] 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.

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

[11] V. Venkatasubramanian, R. Rengaswamy, S. N. Kavuri, and K. Yin, "A review of process fault detection and diagnosis: Part III: Process history based methods," *Computers & Chemical Engineering*, vol. 27, no. 3, pp. 327 – 346, 2003.

[12] S. Yin, S. X. Ding, X. Xie, and H. Luo, "A review on basic data-driven approaches for industrial process monitoring," *IEEE Trans. Ind. Electron*, vol. 61, no. 11, pp. 6418–6428, 2014.

[13] J. MacGregor and T. Kourti, "Statistical process control of multivariate processes," *Control Engineering Practice*, vol. 3, pp. 403 – 414, 1995.

[14] I. Izadi, S. L. Shah, and T. Chen, "Effective resource utilization for alarm management," in *49th IEEE Conf. Decision and Control (CDC)*, 2010, pp. 6803–6808.

[15] S. R. Kondaveeti, S. Shah, and I. Izadi, "Application of multivariate statistics for efficient alarm generation," in *Proc. Fault Detection, Supervision and Safety of Technical Processes*, 2009, pp. 657–662.

[16] J. V. Kresta, J. F. MacGregor, and T. E. Marlin, "Multivariate statistical monitoring of process operating performance," *The Canadian Journal of Chemical Engineering*, vol. 69, no. 1, pp. 35–47, 1991.

[17] Z. Ge, Z. Song, and F. Gao, "Review of recent research on data-based process monitoring," *Industrial & Engineering Chemistry Research*, vol. 52, no. 10, pp. 3543–3562, 2013.

[18] R. V. Beard, "Failure accomodation in linear systems through self-reorganization," Ph.D. dissertation, Massachusetts Institute of Technology, 1971.

[19] P. Frank and X. Ding, "Survey of robust residual generation and evaluation methods in observer-based fault detection systems," *Journal of Process Control*, vol. 7, no. 6, pp. 403 – 424, 1997.

[20] H. L. Jones, "Failure detection in linear systems," Ph.D. dissertation, Massachusetts Institute of Technology, 1973.

[21] A. S. Willsky, "A survey of design methods for failure detection in dynamic systems," *Automatica*, vol. 12, no. 6, pp. 601–611, 1976.

[22] E. Y. Chow and A. S. Willsky, "Analytical redundancy and the design of robust failure detection systems," *IEEE Trans. Autom. Control*, vol. 29, no. 7, pp. 603–614, 1984.

[23] S. Ding, *Model-based Fault Diagnosis Techniques: Design Schemes, Algorithms, and Tools*. Springer Science & Business Media, 2008.

[24] R. Isermann, "Process fault detection based on modeling and estimation methods - a survey," *Automatica*, vol. 20, no. 4, pp. 387–404, 1984.

[25] P. M. Frank, "Fault diagnosis in dynamic systems using analytical and knowledge-based redundancy: A survey and some new results," *Automatica*, vol. 26, no. 3, pp. 459–474, 1990.

[26] J. Chen and R. J. Patton, *Robust Model-based Fault Diagnosis for Dynamic Systems*. Springer Science & Business Media, 2012, vol. 3.

[27] J. J. Gertler, "Survey of model-based failure detection and isolation in complex plants," *IEEE Control. Syst. Mag*, vol. 8, no. 6, pp. 3–11, 1988.

[28] R. Isermann, "Model-based fault-detection and diagnosis–status and applications," *Annual Reviews in control*, vol. 29, no. 1, pp. 71–85, 2005.

[29] J. Gertler, *Fault Detection and Diagnosis in Engineering Systems*. CRC press, 1998.

[30] S. Yin, S. X. Ding, X. Xie, and H. Luo, "A review on basic data-driven approaches for industrial process monitoring," *IEEE Trans. Ind. Electron.*, vol. 61, no. 11, pp. 6418–6428, 2014.

[31] Y. Cheng, "Data-driven techniques on alarm system analysis and improvement." Ph.D. dissertation, University of Alberta, 2013.

[32] J. F. MacGregor and T. J. Harris, "Discussion," *Technometrics*, vol. 32, no. 1, pp. 23–26, 1990.

[33] W. H. Woodall and M. M. Ncube, "Multivariate cusum quality-control procedures," *Technometrics*, vol. 27, no. 3, pp. 285–292, 1985.

[34] Y. Cheng, I. Izadi, and T. Chen, "Optimal alarm signal processing: Filter design and performance analysis," *IEEE Trans. on Automation Science and Engineering*, vol. 10, no. 2, pp. 446–451, 2013.

[35] J. Wang, F. Yang, T. Chen, and S. L. Shah, "An overview of industrial alarm systems: Main causes for alarm overloading, research status, and open problems," *IEEE Trans. Automation Science and Engineering*, vol. 13, no. 2, pp. 1045–1061, 2016.

[36] T. Kourti and J. F. MacGregor, "Process analysis, monitoring and diagnosis, using multivariate projection methods," *Chemometrics and intelligent laboratory systems*, vol. 28, no. 1, pp. 3–21, 1995.

[37] P. Nomikos and J. F. MacGregor, "Multivariate spc charts for monitoring batch processes," *Technometrics*, vol. 37, no. 1, pp. 41–59, 1995.

[38] I. Jolliffe, *Principal Component Analysis*. Wiley Online Library, 2002.

[39] J. F. MacGregor, C. Jaeckle, C. Kiparissides, and M. Koutoudi, "Process monitoring and diagnosis by multiblock pls methods," *AIChE Journal*, vol. 40, no. 5, pp. 826–838, 1994.

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

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

[42] J.-M. Lee, C. Yoo, and I.-B. Lee, "Fault detection of batch processes using multiway kernel principal component analysis," *Computers & Chemical Engineering*, vol. 28, no. 9, pp. 1837–1847, 2004.

[43] 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.

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

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

[46] P. Comon, "Independent component analysis, a new concept?" *Signal processing*, vol. 36, no. 3, pp. 287–314, 1994.

[47] T.-W. Lee, M. Girolami, and T. J. Sejnowski, "Independent component analysis using an extended infomax algorithm for mixed subgaussian and supergaussian sources," *Neural computation*, vol. 11, no. 2, pp. 417–441, 1999.

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

[49] K. Hu and J. Yuan, "Multivariate statistical process control based on multiway locality preserving projections," *Journal of Process Control*, vol. 18, no. 7, pp. 797–807, 2008.

[50] F. He and J. Xu, "A novel process monitoring and fault detection approach based on statistics locality preserving projections," *Journal of Process Control*, vol. 37, pp. 46 – 57, 2016.

[51] S. Yin, S. X. Ding, A. Haghani, H. Hao, and P. Zhang, "A comparison study of basic data-driven fault diagnosis and process monitoring methods on the benchmark tennessee eastman process," *Journal of Process Control*, vol. 22, no. 9, pp. 1567 – 1581, 2012.

[52] 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.

[53] F. P. Preparata and M. Shamos, *Computational Geometry: An Introduction*. Springer Science & Business Media, 2012.

[54] H. Edelsbrunner, D. Kirkpatrick, and R. Seidel, "On the shape of a set of points in the plane," *IEEE Trans. Inf. Theory*, vol. 29, no. 4, pp. 551–559, 1983.

[55] V. J. Hodge and J. Austin, "A survey of outlier detection methodologies," *Artificial Intelligence Review*, vol. 22, no. 2, pp. 85–126, 2004.

[56] I. Ruts and P. J. Rousseeuw, "Computing depth contours of bivariate point clouds," *Computational Statistics & Data Analysis*, vol. 23, no. 1, pp. 153–168, 1996.

[57] E. M. Knorr and R. T. Ng, "Algorithms for mining distance-based outliers in large datasets," in *Proc. 24rd Int. Conf. Very Large Data Bases*, 1998, pp. 392–403.

[58] M. Luo, "Multivariate fault detection with convex hull," in *Digital Avionics Systems Conf.*, vol. 2, 2004, pp. 7–E.

[59] ——, "Data-driven fault detection using trending analysis," Ph.D. dissertation, Tennessee Tech University, 2006.

[60] E. Stehle, K. Lynch, M. Shevertalov, C. Rorres, and S. Mancoridis, "On the use of computational geometry to detect software faults at runtime," in *Proc. 7th Int. Conf. Autonomic Computing*, 2010, pp. 109–118.

[61] K. Lynch, D. M. Lofaro, and P. Oh, "A n-dimensional convex hull approach for fault detection and mitigation for high degree of freedom robots humanoid robots," in *2012 12th Int. Conf. Control, Automation and Systems (ICCAS), 2012 12th*. IEEE, 2012, pp. 790–797.

[62] J. F. MacGregor and T. Kourti, "Statistical process control of multivariate processes," *Control Engineering Practice*, vol. 3, no. 3, pp. 403–414, 1995.

[63] S. Yin, "Data-driven design of fault diagnosis systems," Ph.D. dissertation, Duisburg, Essen, Universität Duisburg-Essen, Diss., 2012, 2012.

[64] L. Fortuna, S. Graziani, A. Rizzo, and M. G. Xibilia, *Soft Sensors for Monitoring and Control of Industrial Processes*. Springer Science & Business Media, 2007.

[65] R. Rosipal and N. Krämer, "Overview and recent advances in partial least squares," in *Proc. 2005 Int. Conf. Subspace, Latent Structure and Feature Selection*, 2005, pp. 34–51.

[66] S. Wold, M. Sjstrm, and L. Eriksson, "Pls-regression: a basic tool of chemometrics," *Chemometrics and Intelligent Laboratory Systems*, vol. 58, no. 2, pp. 109 – 130, 2001.

[67] P. Geladi and B. R. Kowalski, "Partial least-squares regression: a tutorial," *Analytica Chimica Acta*, vol. 185, pp. 1 – 17, 1986.

[68] S. de Jong, "SIMPLS: an alternative approach to partial least squares regression," *Chemometrics and Intelligent Laboratory Systems*, vol. 18, no. 3, pp. 251 – 263, 1993.

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

[70] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural computation*, vol. 13, no. 7, pp. 1443–1471, 2001.

[71] Y. Chen, X. S. Zhou, and T. S. Huang, "One-class SVM for Learning in Image Retrieval," in *International Conference on Image Processing 2001*, vol. 1, 2001, pp. 34–37.

[72] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa, "The quickhull algorithm for convex hulls," *ACM Transactions on Mathematical Software*, vol. 22, no. 4, pp. 469–483, 1996.

[73] C. B. Barber, "Computational geometry with imprecise data and arithmetic," Ph.D. dissertation, Princeton University, 1992.

[74] "Qhull," http://www.qhull.org/, accessed: 2016-05-19.

[75] S. Yin, X. Zhu, and C. Jing, "Fault detection based on a robust one class support vector machine," *Neurocomputing*, vol. 145, pp. 263–268, 2014.

[76] P. M. d'Ocagne, *Coordonnes Parallles et Axiales: Mthode de transformation gomtrique et procd nouveau de calcul graphique dduits de la considration des coordonnes paralllles.* Gauthier-Villars, 1885.

[77] A. Inselberg, "The plane with parallel coordinates," *The Visual Computer*, vol. 1, no. 2, pp. 69–91, 1985.

[78] J. Heinrich and D. Weiskopf, "State of the art of parallel coordinates," *STAR Proc. of Eurographics*, vol. 2013, pp. 95–116, 2013.

[79] A. Inselberg, "Multidimensional detective," in *Proc. 1997 IEEE Symp. Information Visualization*, Oct 1997, pp. 100–107.

[80] R. E. Moustafa, "Qgpcp: quantized generalized parallel coordinate plots for large multivariate data visualization," *Journal of Computational and Graphical Statistics*, vol. 18, no. 1, pp. 32–51, 2009.

[81] R. Dunia, T. F. Edgar, and M. Nixon, "Process monitoring using principal components in parallel coordinates," *AIChE Journal*, vol. 59, no. 2, pp. 445–456, 2013.

[82] E. J. Wegman, "Hyperdimensional data analysis using parallel coordinates," *Journal of the American Statistical Association*, vol. 85, no. 411, pp. 664–675, 1990.

[83] R. Brooks, R. Thorpe, and J. Wilson, "A new method for defining and managing process alarms and for correcting process operation when an alarm occurs," *Journal of Hazardous Materials*, vol. 115, no. 1, pp. 169–174, 2004.

[84] "Process plant computing limited," http://www.ppcl.com/, accessed: 2016-05-19.

[85] A. Inselberg, *Parallel Coordinates.* Springer, 2009.

[86] A. Inselberg, M. Reif, and T. Chomut, "Convexity algorithms in parallel coordinates," *Journal of the ACM (JACM)*, vol. 34, no. 4, pp. 765–801, 1987.

[87] D. Zhou, G. Li, and S. Qin, "Total projection to latent structures for process monitoring," *AIChE Journal*, vol. 56, no. 1, pp. 168–178, 2010.

[88] N. F. Thornhill, S. C. Patwardhan, and S. L. Shah, "A continuous stirred tank heater simulation model with applications," *Journal of Process Control*, vol. 18, no. 34, pp. 347 – 360, 2008.

[89] S. B. Chitralekha, S. L. Shah, and J. Prakash, "Detection and quantification of valve stiction by the method of unknown input estimation," *Journal of Process Control*, vol. 20, no. 2, pp. 206–216, 2010.

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

[91] 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.

[92] N. L. Ricker, "Decentralized control of the tennessee eastman challenge process," *Journal of Process Control*, vol. 6, no. 4, pp. 205 – 221, 1996.

Appendix A

Outline of Qhull Algorithm

Algorithm 3 Qhull

1: **Input:**
    a set of d dimensional data points X
2: **Initialize:**
    create a d simplex with $d+1$ points in X
3: **for each** facet f **do**
4: **for each** unassigned $x \in X$ **do**
5: **if** x is above f **then**
6: assign x to f's outside set
7: **end if**
8: **end for**
9: **end for**
10: **for each** f with a non-empty outside set **do**
11: select the furthest point x of f's outside set
12: initialize the visible facet set FV of x
13: **for each** unvisited neighbor f of facets in FV **do**
14: **if** x is above f **then**
15: add f to FV
16: **end if**
17: **end for**
18: identify the boundary of FV as a set of ridge RB
19: **for each** ridge $r \in RB$ **do**
20: create a new facet from r and x
21: **end for**
22: **for each** new facet $f^{'}$ **do**
23: **for each** point y in an outside set of a facet in FV **do**
24: **if** y is above $f^{'}$ **then**
25: assign y to $f^{'}$'s outside set

Algorithm 3 Qhull (continued)

```
            end if
        end for
    end for
    delete the facets in FV
end for
Output:
    list of facet F, list of vertices V and list of facet hyperplane
    functions including normal vectors in N and offset vector S
```