University of Alberta

Support-Vector-Machine-Based Diagnostics and Prognostics for Rotating Systems

by

Jian Qu

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Engineering Management

Department of Mechanical Engineering

Fall 2011
Edmonton, Alberta

This dissertation is dedicated with my respect to my father and my mother. Without their unconditional support and love, I would not complete my study.

This dissertation is dedicated with my love to my wife, Yang. With her love, encouragement and accompanying, I am able to defeat any difficulties and frustrations.

This dissertation is also dedicated to my son, Alex. With him around, everyday is full of surprise and happiness.

Abstract

In recent decades, condition-based maintenance has been acknowledged as a cost-effective maintenance program and widely used in many engineering systems. Diagnostics and prognostics are critical components of condition-based maintenance, responsible for providing information about present and future system health conditions. These two components are respectively an integrated process covering several essential aspects that should be cared about to ensure successful implementation of diagnostics and prognostics. This thesis focuses on certain aspects for diagnostics and prognostics, respectively.

When doing diagnostics, data processing is an important stage responsible for providing reliable data for use. Data processing usually includes feature extraction, data cleaning and feature selection. Feature extraction is responsible for extracting from the raw data characteristic features representing system conditions. Data cleaning is necessary for removing outliers caused by the noise during data collection. Feature selection is responsible for capturing and removing useless features generated during feature extraction. This thesis focuses on data cleaning and feature selection.

When doing prognostics, noise may appear in condition indicator values. The condition indicator is extracted from condition monitoring data and is able to reflect the health conditions of monitored assets. Generally, using the noisy condition indicator values may result in unreliable predictions for prognostics, so there is a demand of the method that can provide predictions without the effects of noise.

Support vector machine (SVM) is recognized an effective tool for classification and prediction that are needed by diagnostics and prognostics. SVM is a supervised-learning method and is reported to have better generalization ability and superior performance for small sample cases over other supervised learning methods such as

artificial neural network. This thesis develops SVM-based methods to solve some problems existing in diagnostics and prognostics.

The contributions of this thesis are summarized as follows:

1. An SVM-based data cleaning algorithm that removes outliers for effective diagnostics.
2. An SVM-based feature selection algorithm that removes useless features for effective diagnostics.
3. An analytical method that selects SVM model parameters for effective on-line system condition prognostics.
4. An intelligent optimization-based method that selects SVM model parameters for effective on-line system condition prognostics.

Acknowledgements

I would like to express my gratitude with sincere respect to my supervisor Dr. Ming J. Zuo for his support, care, and encouragement through all my study and research. Under his guidance, I have gained not only valuable academic training but also useful logic and methodology to deal with real world problems. It is my honor to research under this guidance.

My sincere thanks to my Ph.D. examining committee members, Dr. John Doucette, Dr. Khashayar Khorasani, Dr. Mike Lipsett, Dr. Yongsheng Ma, and Dr. Qing Zhao, for their helps in improving this thesis. Thanks to Dr. Mike Lipsett for providing valuable comments on my thesis writing. Thanks to Dr. Yongsheng Ma for his suggestions of preparing and the efforts of chairing my final oral examination. Also thanks to Dr. Khashayar Khorasani for taking efforts to have a telephone conference for my final oral examination.

I am also grateful to all the members in our Reliability Research Lab for their help and support. Working with them is always an exciting and beneficial experience. I am pleased to work with them in the past five years.

Table of Contents

List of Tables

List of Figures

Nomenclature

ADAQ	Acoustic Data Acquisition
AE	Acoustic Emission
AI	Artificial Intelligence
AIA	Artificial Immunization Algorithm
ANFIS	Adaptive Neuro-Fuzzy Inference System
ANN	Artificial Neural Network
AR	Auto-regressive
ARMA	Auto-regressive Moving Average
BMP	Band-pass Mesh Signal
BP	Back Propagation
BPNN	Back-Propagation Neural Network
BS	Backward Selection
CA	Classification Accuracy
CBM	Condition-based Maintenance
CI	Condition Indicator
CM	Condition Monitoring
CPU	Central Processing Unit

CT	Classification Tree
CUSUM	Cumulative Sum
CV	Cross-Validation
DIFF	Difference Signal
DLDA	Diagonal Linear Discriminant Analysis
FFNN	Feed Forward Neural Network
FFT	Fast Fourier Transform
FLI	Fuzzy Logic Inference
FNA	Fine Needle Aspirate
FS	Forward Selection
GA	Genetic Algorithms
GMF	Gear Mesh Frequency
GRNN	Generalized Regression Neural Network
HMM	Hidden Markov Chain
HP	Horse Power
HS	High Sensitivity
KFCV	K-Fold Cross Validation
KKT	Karsh-Kuhn-Tucker
KNN	K-Nearest Neighbor
LOOCV	Leave-One-Out Cross Validation
LR	Logistic Regression
LS	Low Sensitivity

LSSVR	Least Square Support Vector Regression
MD	Mahalanobis Distance
MR	Misclassification Rate
NBC	Naive Bays Classifier
NFIS	Neuro-Fuzzy Inference System
NL	Noise Level
NRMSE	Normalized Root Mean Square Error
OAA	One-Against-ALL
OAO	One-Against-One
PBP	Percentage of Better Performance
PCA	Principal Component Analysis
PDF	Probability Density Function
PHM	Proportional Hazard Model
PIP	Pump Inlet Pressure
POP	Pump Outlet Pressure
RAW	Raw Signal
RBF	Radial Basis Function
RBS	Recursive Backward Selection
RES	Residual Signal
RMC	Regular Mesh Components
RMS	Root Mean Square
RNL	Relative Noise Level

RPM	Revolution per Minute
RSV	Random Sub-sampling Validation
RTF	Run-to-Failure
RUL	Remaining Useful Life
SD1	Simulation Dataset 1
SD2	Simulation Dataset 2
SNR	Signal-to-Noise Ratio
SP	Separating Plane
SR	Selection Rate
SSM	State-Space Model
SVC	Support Vector Classification
SVM	Support Vector Machine
SVR	Support Vector Regression
TVCI	True Value of Condition Indicator
VDAQ	Vibration Data Acquisition
WPT	Wavelet Packet Transform
WT	Wavelet Transform

Chapter 1

Introduction

In engineering systems, one tends first to think of airplanes, cars, shovels, trucks, and so forth. These systems play an important role in many aspects of our daily life and in industry. In industrial applications, engineering systems may be used to handle difficult tasks which encompass high speeds, heavy loads, or harsh environments such as low temperatures and dusty surroundings. Such harsh working conditions accelerate the deterioration of engineering systems and accordingly increase the frequency of unexpected system breakdowns.

An example is the shovel used in oil sands excavation. Figure 1-1 shows a typical working scenario for a shovel at Syncrude Canada Limited [1]. Because of the high output torque required, this kind of shovel is usually equipped with a planetary gearbox. It is known that gear teeth experience cyclic stress when gears engage. This stress eventually contributes to gear fatigue damage such as pits and cracks. If nothing preventive is done, the gear damage will progress faster and faster until it results in failures such as broken teeth which incur unexpected downtime for the shovel [2].

When an engineering system breaks down, one has to take action to restore it to operating condition. This action is part of a maintenance program. Traditional maintenance strategies can be categorized into two classes [3]: corrective maintenance and preventive maintenance. Corrective maintenance is the maintenance that occurs when a system fails. Some researchers refer to it as repair. According to MIL-STD-721B, corrective maintenance means all actions performed as a result of failure, to restore an item to a specified condition. Preventive maintenance is

the maintenance that occurs when a system is operating. According to MIL-STD-721B, preventive maintenance means all actions performed in an attempt to retain an item in a specified condition by providing systematic inspection, detection, and prevention of incipient failures.

Figure 1-1: A shovel loading oilsand into haul truck [1]

Domestic plants in the United States spent more than $600 billion to maintain critical plant systems in 1981, and this figure doubled within 20 years [4]. An even more shocking fact is that one-third to one-half of this expenditure was wasted through ineffective maintenance [5]. This reveals the need for an advanced maintenance strategy which can implement maintenance actions in a more cost-effective way.

1.1 Condition-Based Maintenance

In recent decades, condition-based maintenance (CBM) has developed rapidly. CBM conducts maintenance based on condition monitoring (CM) data relevant to system operating states, and it is able to avoid unnecessary maintenance actions. Figure 1-2

(adapted from [6, 7]) exhibits a CBM framework where each stage performs a unique function to ensure the process of CBM can be successfully implemented.

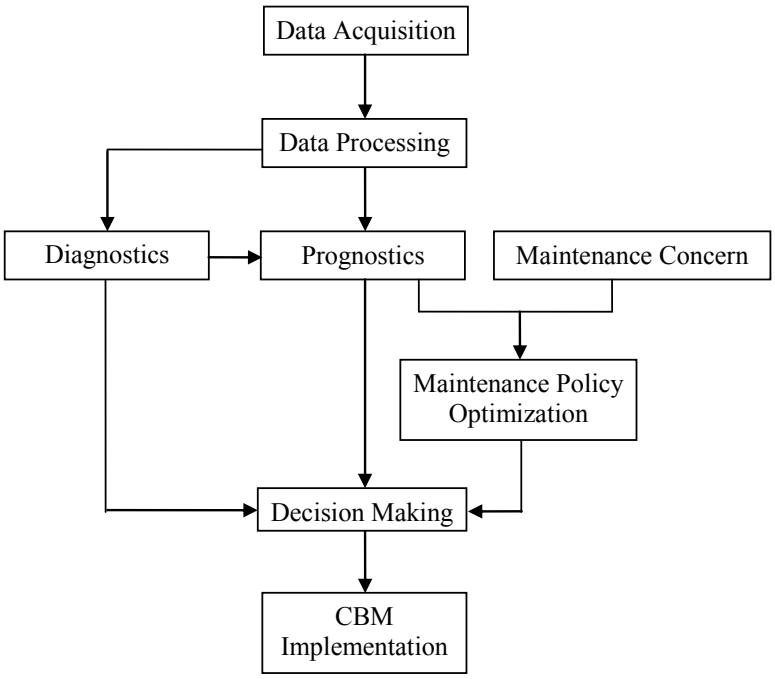

Figure 1-2: The structure of CBM

Data acquisition is a process of collecting and storing CM data for a system of interest. Monitoring instruments are used in this stage, e.g. thermometers, pressure gauges, vibration sensors, acoustic emission sensors, particle counters, etc. The CM data collected are called raw data. These data may fall into different categories. More details about data acquisition are given in Chapter 2.

Data processing is a process of providing reliable processed data for the following stages. Three sub-stages that may be included in data processing are: data cleaning, feature extraction, and feature selection. Feature refers to the characteristics that are able to reflect the conditions of the asset of interest. Data cleaning refers to identifying and eliminating outliers. It can be implemented before or after feature extraction. Feature extraction refers to calculating characteristic features from raw data for the use of diagnostics and prognostics. As the extracted features may not

all be useful, feature selection is usually used to select the most useful ones. Chapter 2 gives more details on data processing. The output of data processing can be used in both diagnostics and prognostics.

Diagnostics refers to either an off-line process of searching for the cause of a system breakdown or an online process of detecting abnormality in a system. In this thesis, diagnostics, if not otherwise stated, always refers to the online process. The outputs of diagnostics can be fault modes and fault levels which are indicative of the current state of the system. Diagnostics is one focal point of this thesis. More details on diagnostics are given in Section 1.2.

Prognostics refers to an online process of estimating remaining useful life (RUL); in other words, failure forecasting. The output of prognostics indicates future states of the system, for example the time period over which a system will normally operate. The results provided by diagnostics can be used in prognostics, because the current state of a system is useful information for predicting future ones. Prognostics is the other focal point of this thesis. More details on prognostics are given in Section 1.3.

Maintenance concerns refer to the factors that need to be considered when making maintenance decisions. A common factor is maintenance cost, that is, the cost of implementing maintenance. Other factors may include maintenance degree, availability of maintenance personnel, availability of a maintenance facility, productivity, unfilled orders and so forth [3].

Maintenance policy optimization refers to a mathematical optimization process which offers options for maintenance strategy by jointly considering system health status (the output of prognostics) and maintenance concerns. In the literature, researchers usually consider minimizing maintenance cost in terms of cost per unit time [8–10]. The output of this stage provides information about the time at which conducting maintenance will cost the least.

Decision making refers to selecting the best maintenance option for a given system. The options can be obtained from diagnostics, prognostics and maintenance policy optimization. Expert knowledge and experience is also usually incorporated

into making a decision. Once an option is finalized, maintenance scheme (e.g. schedule, degree, personnel assignment, etc.) is made based on concrete conditions of involving maintenance resources. Eventually, a maintenance action is implemented; this is the last stage of a CBM program.

From Figure 1-2, we can see that the output of diagnostics and prognostics is crucial to the success of a CBM program. In terms of diagnostics, the output depends on not only the performance of the diagnostic approach but also the quality of data transmitted from previous stages; thus it is more appropriate to treat diagnostics as an integrated process covering these relevant stages. For this reason, this thesis defines diagnostics as a process involving data acquisition, data processing and diagnostic implementation where diagnostic implementation refers to the stage of identifying fault characteristics (modes, levels, etc.) using diagnostic approaches. The above is also true for prognostics. There is a stage of prognostic implementation which refers to predicting failures using prognostic approaches. Sections 1.2 and 1.3 further introduce diagnostics and prognostics, more specifically the diagnostic and prognostic approaches.

1.2 Diagnostics

A fault is defined as an abnormal condition or a defect at the component, equipment, or subsystem level which may lead to a failure of system; however, a fault in these assets may not necessarily cause a system failure. Diagnostics is able to detect the minor fault and track its progression; as a result, the failure of system could be prevented. For better understanding of the fault and failure, the terms minor fault, major fault, and failure are used in the following context: an incipient fault can be regarded as a minor fault; a minor fault progresses to a major fault as the system operates; a major fault corresponds to a higher severity level and approaches a failure. When either a minor or a major fault occurs, the system performs at a low level, but continues to operate. When a failure occurs, the system no longer operates and must be repaired.

When a fault occurs in a system, symptoms can be detected through monitored measures, e.g. excessive vibration and noise, extremely increased temperature, oil

debris, etc. Utilized effectively and appropriately, such symptoms are helpful for detecting minor faults. Such symptoms are also known as an indicator or a feature in the scope of diagnostics. Although the ultimate failure of a system may be inevitable, diagnosing and fixing faults can postpone the failure and prolong the system's operational time.

Many approaches have been reported for diagnostics. Good reviews of them can be found in [7, 11, 12]. These approaches can be classified into three categories: signal-based approaches, knowledge-based approaches, and data-driven approaches. They will be introduced one by one below; however, data-driven approaches are the main focus of this thesis.

Signal-based approaches focus on detecting changes or variations in a signal and subsequently identifying the changes. Researchers have developed indicators that reflect system conditions, using such approaches as spectral analysis and wavelet analysis [13]. In the 1990s, tracing a single indicator for fault detection prevailed. Since 2000, the indicators obtained using the signal-based approaches are often incorporated with data-driven approaches to achieve better diagnostic results. The data-driven approaches are introduced later.

Knowledge-based approaches embody knowledge in a narrow domain related to the solutions to problems. These approaches simulate the ways that human expertise thinks and infers, creating a series of rules whereby solutions are produced. Typical examples of these approaches are the rule-based expert system and fuzzy logic inference (FLI) [14–16]. Though they have been successfully used in diagnostics, knowledge-based approaches have their shortcomings. For example, expert system can not deal with new situations, that is, ones not covered explicitly in its knowledge bases; and FLI needs good membership functions and appropriate fuzzy rules which are not always easy to obtain.

Data-driven approaches are usually conducted to classify fault modes or levels (this is discussed in Section 1.4). They further include statistical methods and artificial intelligence (AI) methods. Data-driven approaches employ a parametric model of which parameters need to be determined by training data. The parameters are input arguments required to establish the model. The training data are

known samples carrying information useful for classification. Statistical methods generalize the underlying dependency of given data through explicit mathematical models derived from certain statistical theories. AI methods achieve the same thing through a model simulating biological mechanism. Statistical methods are usually subject to certain assumptions such as data distribution and independency; while AI methods have no such constraints. Unfortunately, however, AI models are implicit and usually can not be explained to users.

One focus of this thesis is SVM-based diagnostics. SVM is an AI method which belongs among the data-driven approaches. More about SVM-based diagnostics is talked in Section 1.4. For better understanding of the state-of-the-art for data-driven approaches, some typical methods are reviewed in Chapter 2.

1.3 Prognostics

Prognostics is a process of predicting failure thereby the future life profile of system can be revealed [7]. Prognostics aims at preventing failure, but operates in a way different from that of diagnostics. Diagnostics can tell whether a system is subject to a minor fault or a major fault at present, but it can not tell how long the system will continue to operate. In contrast, prognostics can tell this — it is also known as RUL estimation. This good capability makes prognostics very attractive to maintenance decision makers. In short, diagnostics cares about detecting what is happening at present, and prognostics cares about predicting what will happen in the future; hence, some researchers consider prognostics to be superior to diagnostics [7]. Nevertheless, prognostics is technically more expensive to implement and more difficult to fulfill, so it is usually used for critical assets.

Prognostic approaches fall into four categories. Good reviews of them can be found in [5–7, 12, 17]. They are introduced one by one below; however, data-driven approaches are again my main focus.

A simple form of prognostics can be found in the experience-based approaches. These approaches gather statistical information about the amount of time that a component lasts before failure, and use these statistics to make RUL predictions. These predictions are dependent solely on the passage of time and/or measures of

usage of the system or component. For example, for a timing belt on an automobile, the manufacturer may recommend that the belt be replaced every five years or 60,000 miles.

Model-based approaches usually use mathematical models for the system being monitored. One prevailing method of this kind is the physics-based model. Physics-based models are developed by experts in the component field and validated on large sets of data to show that they are indeed accurate. They are useful in tracking realistic operating conditions of systems. Model-based prognostic approaches are only as good as the models on which they are based. For physics-based models, the understanding of failure mechanisms is often impractical and sometimes even impossible [6].

Knowledge-based approaches for prognostics are similar to those for diagnostics. These approaches mimic the ways in which human expertise thinks and infers thereby creating a series of rules through which solutions are produced. Unlike those for diagnostics, knowledge-based approaches are used for prediction in prognostics. Rule-based expert system and FLI are two typical examples [18]. The advantages and disadvantages are discussed in Section 1.2 and therefore will not be repeated here.

Unlike diagnostics, data-driven approaches for prognostics employ a parametric model to track system deterioration and predict system conditions. They also include statistical methods and AI methods. Statistical methods use explicit mathematical expressions to first calculate the probability density function (PDF) of RUL and then estimate the expectation of RUL based on the PDF obtained. AI methods otherwise obtain a real-value prediction (the point estimate) of the condition indicator based on past ones, and then estimates RUL according to a pre-specified threshold of the condition indicator. Statistical methods can give a probabilistic measure evaluating the certainty of the condition indicator being a certain value at a certain time. This is attractive in realistic cases because it allows maintenance personnel to incorporate their expertise into decision making; nevertheless, statistical methods are usually complicated and subject to assumptions that limit their applications. AI methods are unable to provide the probabilistic measure, but they

are easy to implement and free of restrictions, so they are more widely used. Some typical data-driven approaches are reviewed in Chapter 2. SVM-based prognostics, the other focus of this thesis, is discussed in the next section.

1.4 Support-Vector-Machine-Based Diagnostics and Prognostics

In recent decades, SVM has attracted more and more attention in diagnostics and prognostics because of its better generalization ability and superior performance for small sample cases [11, 19]. The mathematical explanations of these good characteristics are explained in Chapter 3 (see Sections 3.1.1 and 3.2.2). This thesis does not concentrate on analytical and mathematical development of SVM, but the applications of SVM in diagnostics and prognostics, so the formal and rigorous substantiations of these characteristics are not given. As mentioned in Sections 1.2 and 1.3, AI methods realize classification for diagnostics and prediction for prognostics. SVM is applicable for both classification and prediction, achieved by support vector classification (SVC) and support vector regression (SVR), respectively. Conventionally, researchers do not differentiate between SVC and SVR in their work, and simply call them SVM. This thesis follows this convention. Readers can easily distinguish them by following the rule that SVC is for diagnostics and SVR is for prognostics. More details of SVC and SVR are presented in Chapter 3.

As announced in Section 1.1, this thesis treats diagnostics and prognostics as an integrated process. For both SVM-based diagnostics and prognostics, the first two stages are data acquisition and data processing. The third stage is SVM-based classification for diagnostics and SVM-based prediction for prognostics. These two processes are illustrated in Figure 1-3 which is actually a part of Figure 1-2, but with output included.

For SVM-based diagnostics, outputs are usually fault modes and/or fault levels. Fault mode refers to the type of fault subject to a certain fault mechanism. Fault level refers to the severity of a certain fault. Common fault levels are binary labels, namely, normal and faulty; classifying fault levels is synonymous with fault

detection, a widely studied problem. Complex problems can include multiple labels. These labels could be different fault modes, e.g, gear crack, gear pit and gear tooth missing, and different fault levels each of which corresponds to a certain range of quantitative measures, e.g. four crack levels described as no damage, slight, moderate and severe where the "slight" level corresponds to a crack size of [0.5mm, 1mm]. Based on these outputs, one knows what type of fault the system is currently subject to and is at what level it is subject to it. Many studies have been reported in this area [11, 20–23], some of which are reviewed in Chapter 2.

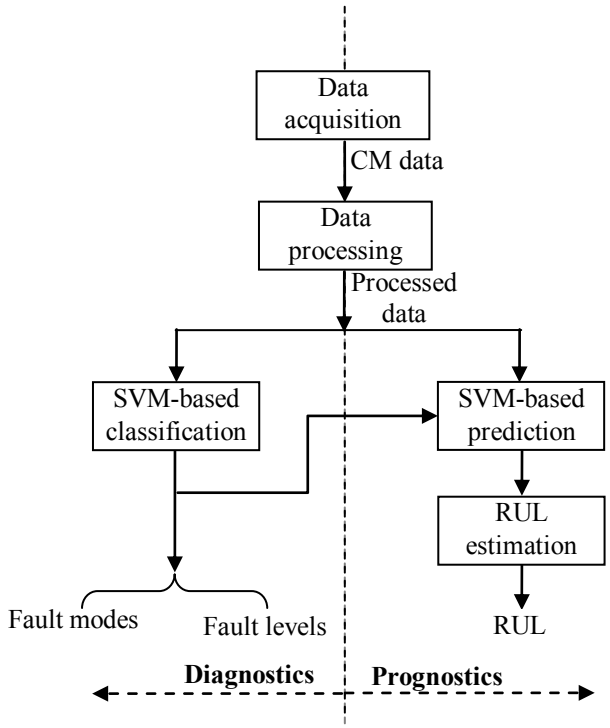

Figure 1-3: Flow chart of SVM-based diagnostics and prognostics

For SVM-based prognostics, the output is RUL based on which one can know how long a system will continue to operate. SVM is used to predict the condition indicator (CI). Then, the CI value is compared with a pre-specified threshold to determine the RUL in the RUL estimation stage. One can also trace the predicted trend of the CI to foresee system conditions; this is known as condition prognostics [19]. The output of SVM-based classification can also be used in SVM-based prediction. This, however, is not my research focus, so this thesis treats SVM-based

diagnostics and prognostics independently. SVM-based prognostics has attracted increasing attention in recent years. Some researchers have reported successful applications in this area [19, 24–27]; however, compared to SVM-based diagnostics, SVM-based prognostics has not been sufficiently studied. Some published studies are reviewed in Chapter 2.

Though SVM has been successfully used in diagnostics and prognostics, the applications are limited to the classification and prediction stages. Besides, the performance of SVM is likely to be affected by the quality of the data. These concerns have inspired the research in this thesis and they are defined in the next section.

1.5 Research Objectives

As discussed in Section 1.4, SVM has been widely used in the classification stage of diagnostics; however, its potential for use in other stages, such as data cleaning and feature selection, to improve diagnostic results is lack of study. Some work reports using SVM in feature selection [28, 29]. To the best of my knowledge, SVM for data cleaning has been rarely reported. Since these two stages are also important for the success of diagnostics, the following two topics are specifically investigated in this thesis:

- developing an SVM-based data cleaning method for diagnostics, and
- developing an SVM-based feature selection method for diagnostics.

As discussed in Section 1.4, some researchers have reported using SVM in the prediction stage of prognostics. The performance of prediction relies heavily on the behavior of the SVM model, so the selection of SVM model parameters is important. When using SVM for prediction, the random noise which contributes to over-fitting and under-fitting is another critical factor influencing prediction results. Two types of method are reported as overcoming the noise effects by means of selecting appropriate SVM parameters. One is the analytical method which employs explicit expressions for selecting the parameters [29–32]; the other is the optimization-based

method which selects the parameters via an optimization model incorporating cross-validations [24, 25]. The analytical method can address noise effects in its expressions, but it is not case-specific; the optimization-based method is case-specific, but has no explicit term for addressing noise effects and does not update the selected parameters along with the passage of time. For these reasons, the following two topics are specifically investigated in this thesis:

- modifying a reported analytical method in order to select appropriate SVM model parameters for better prognostic results, and
- developing an intelligent optimization-based method for selecting SVM model parameters to mitigate the shortcomings of common optimization-based methods.

1.6 Thesis Organization

This thesis is composed of 7 chapters. Chapter 2 reviews some published data-driven approaches to diagnostics and prognostics. Chapter 3 presents the fundamentals of SVM. Chapter 4 introduces the two experimental systems to be studied in this thesis. Chapter 5 presents an SVM-based diagnostic algorithm which contains one proposed SVM-based data cleaning algorithm and one proposed SVM-based feature selection algorithm. Chapter 6 presents an SVM-based online prognostic algorithm which uses separately two proposed methods of selecting SVM parameters for prediction. Chapter 7 summarizes my contributions and introduces the possible directions for moving forward in my future work.

Chapter 2

Review of Data-Driven Approaches for Diagnostics and Prognostics

As mentioned in Section 1.5, SVM-based diagnostics and prognostics are the focal points of this thesis. SVM belongs among the data-driven approaches; hence, this chapter gives a review of some typical data-driven approaches. As mentioned in Section 1.4, both diagnostics and prognostics employ data acquisition and data processing as their first and second stages and employ data-driven approaches as their third stage. In this chapter, I first briefly introduce data acquisition, then discuss reported data processing methods, and finally review data-driven approaches for diagnostics and prognostics, respectively.

2.1 Data Acquisition

Data acquisition is a process of collecting and storing raw data from sensors which are placed on the system to be monitored. These data usually fall into two categories, event data and condition monitoring (CM) data. Event data include historical information such as installation, breakdown, overhaul, etc., and/or what has been done such as minor repairs, preventive maintenance and lubricating the system. The CM data are versatile; they contain value-type data, waveform data, and multi-dimensional data. The value-type data are gathered over a specific time period for a condition indicator (CI). They are time series data and usually display a trend over the monitoring periods. Examples of this type of data are temperature, pressure

and humidity. The waveform data are gathered at a specific time and are also time series data. This kind of data usually displays a pattern of waveform. Typical examples are vibration signals and acoustic signals. The multi-dimensional data usually display an image over a specific time period. Ultrasonic data and visual images belong to this type.

Data acquisition supplies raw data for diagnostics and prognostics; hence, the performance of data acquisition affects the reliability of the collected data and accordingly the final results of diagnostics and prognostics. Unreliable data may result from various aspects of data acquisition, e.g. malfunction of monitoring instruments, large magnitudes of noise in the external environment, human error, etc. Although this is well known, it is often difficult to ensure the data are 100% reliable, because some factors such as ambient noise are impossible to control. This leaves a problem for the next stage, data processing, to deal with.

2.2 Data Processing

Data processing is implemented after raw data are acquired. The data processing stage can be further divided into three sub-stages: feature extraction, data cleaning, and feature selection. Figure 2-1 shows the procedure for data processing. Data cleaning follows feature extraction, which means that data cleaning is to be conducted in the feature space. It can also be conducted before feature extraction in order to eliminate outliers in the raw data; however, from the viewpoint of fault classification, this kind of methods are not classification-focused. These methods may be able to provide clean data at the raw data level, but when the feature data are extracted from the cleaned data, outliers may still be in the feature data resulting in bad classification. In this case, data cleaning needs to be conducted again to clean the feature data. For this reason, this thesis places the data cleaning after the feature extraction, so the procedure for data processing is as that shown in Figure 2-1. These three sub-stages will be introduced one by one below.

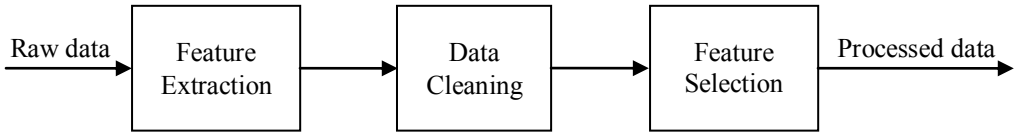

Figure 2-1: Procedure of data processing

2.2.1 Feature Extraction

In most cases, raw data can not be used directly for diagnostic and prognostic purposes. One needs a way of obtaining useful information from the raw data; this is known as feature extraction. The useful information extracted is often referred to as the CI or feature; it reflects the health status of the system. Feature extraction is a widely studied problem for which numerous models, algorithms and tools have been reported. These are reviewed in terms of domain analysis in the following.

Time-domain analysis is directly based on waveform data in the time domain. Traditional features are statistical characteristics such as mean, standard deviation, kurtosis, peak, peak-to-peak, crest factor, skewness, etc. Good reviews of time-domain features can be found in [33, 34]. Samanta et al. [19] used a kurtosis feature for the prognosis of gearbox condition subject to gear pitting wear. Time-domain features can also be extracted from some statistical models. Samuel et al. [35] compared several time-domain features for detecting faults of helicopter gearboxes. Pöyhönen et al. [36] modeled vibration signals gathered from an induction motor using an autoregressive (AR) model and used the coefficients of the AR model as features for fault detection.

Frequency-domain analysis is based on transformed signals in the frequency domain. Unlike time-domain features, frequency-domain features can detect and identify changes of a particular frequency component which may indicate a fault involving a particular system component. In addition, some fault modes may possess specific frequency spectra which make possible non-intrusive identification of system faults. Reported techniques for frequency-domain analysis include fast Fourier transform (FFT), power spectrum analysis, envelope analysis, sideband analysis, and Hilbert transform, among others. Pöyhönen et al. [36] compared the performance of the FFT-based feature selection approach in the frequency domain with

ones based on the AR model. Samanta et al. Tse et al. [37] both predicted the progression of a shaft misalignment based on the shaft's frequency spectrum.

Time domain analysis and frequency domain analysis are based on the assumption that statistical characteristics such as the mean and variance of signals do not vary over time. Such signals are known as stationary signals, but when signals are non-stationary, the above two methods are unable to capture the variations. Time-frequency domain analysis was thus developed to cope with this problem. The most typical method of this kind is wavelet transform (WT). WT has developed rapidly in the recent decades and has been widely used for feature extraction [38]. A continuous WT is defined as:

$$W(a,b) = \frac{1}{a}\int_{-\infty}^{+\infty} x(t)\Psi^*(\frac{t-b}{a})dt, \tag{2.1}$$

where $x(t)$ is waveform signal, a is scale parameter, b is time parameter, and $\Psi^*(\cdot)$ is a wavelet which is a zero average oscillatory function centered around zero with a finite energy. The asterisk means complex conjugate. Widely used WT techniques include continuous WT, discrete WT, and wavelet packet transform (WPT). He et al. [21] used wavelet packet transform (WPT) to extract time-frequency features of valve faults from the vibration signals of three-cylinder reciprocating pumps. Saravanan et al. [39, 40] developed features using various wavelets for classifying bevel gearbox fault modes. Wang et al. [41] developed a continuous wavelet transform based approach for fault detection under variant loads. Samanta et al. Wang et al. [42] used an energy feature obtained through continuous wavelet transform for the prognosis of gearbox condition subject to gear fault modes of wear, chipping and crack, respectively.

2.2.2 Data Cleaning

For some cases, the data obtained from feature extraction are not 100% reliable and need to be further processed in order to identify and eliminate outliers. This is known as data cleaning. In statistics, an outlier is an observation that is numerically distant from the rest of the data [43] or does not follow the pattern of the majority of the data [44]. In the reported literature, the terms "data cleaning", "outlier detection" and

"outlier identification" are used interchangeably. This stage is chosen to be referred to as data cleaning, because it involves two successive actions, detecting outliers and eliminating outliers. The key problem in data cleaning is how to accurately detect existing outliers.

Classical methods of detecting outliers adopt a statistical way where some statistical measures are calculated to determine the distance between a particular data point and the rest of the data points. A widely used method of detecting outliers in multivariate samples is to compute the Mahalanobis distance (MD):

$$\mathrm{MD}_i = \sqrt{(\boldsymbol{x}_i - \boldsymbol{\mu})\boldsymbol{S}^{-1}(\boldsymbol{x}_i - \boldsymbol{\mu})^{\mathrm{T}}}, \tag{2.2}$$

where $\boldsymbol{x}$ is a random vector, representing attributes or features in a N dimensional space, $\boldsymbol{\mu}$ is the mean vector of M sample vectors, $[\boldsymbol{x}_1, \ldots, \boldsymbol{x}_M]$, and $\boldsymbol{S}$ represents the covariance matrix of $\boldsymbol{x}$ of M samples. The MD considers the correlations between different features. This property makes MD superior to Euclidean distance which assumes an independency of features that does not always hold for practical cases. Some reported work on data cleaning is as follows. Rousseeuw et al. [44] proposed to compute distances based on robust estimates of locations and covariance of data points. Fun [45] proposed a method of detecting outliers based on the S-estimation robust method. Shieh et al. [46] proposed an outlier detection method for microarray data based on principal component analysis (PCA) and robust estimation of the MD.

For diagnostics, errors caused by human error, environmental factors, and malfunctioning instruments may occur during data acquisition. This causes the gathered data to contain inaccurate data, contaminated data and/or incomplete data. For this reason, data cleaning is necessary to ensure, or at least increase the chance that, clean (error-free) data are kept for further use [7]. Unfortunately, to our best knowledge, the studies reporting SVM-based data cleaning for diagnostics and prognostics of engineering systems are very limited [47, 48].

2.2.3 Feature Selection

Generally speaking, feature selection for the classification of diagnostics is distinct from that for the prediction of prognostics. For prognostics, researchers focus mainly

on univariate time series prediction where only one feature is involved, so suitable features can be directly selected by visualizing their trends, e.g. if the values for a feature monotonically increase as time passes, this feature could be a good representative of health status of the system. For diagnostics, the rationale behind feature selection is more complex for classification problems where the dimension of feature space is usually greater than one. In this thesis, if not otherwise specified, feature selection always refers to the one for classification problems.

For a certain problem, many features may be extracted according to the reported literature; however, these features may not all be useful since practical cases vary with specific assumptions, constraints, etc. For cases where all features are useful and the number of features is not too large, one can simply use all of them, but if the dimension of feature space is enormous, there will be a huge computational burden for the forthcoming stage, classification. It is, however, more usual for some features to be redundant and irrelevant. The redundant features contribute nothing to the classification results but the dimension of the feature space, and the irrelevant features actually impair the classification results; hence, such features should be selected and eliminated from the feature space.

Feature selection involves two steps. The first is feature ranking which quantitatively assesses how critical to classification an individual feature is and builds a rank of features based on the criticality values. Two models can be used for this step. One is wrapper which ranks features based on the feedback on evaluations, e.g. classification accuracy. The other is filter which ranks features by heuristically determined goodness or knowledge that are able to reflect classification performance, e.g. information entropy. These two models were originally studied in biology and medicine. Balakrishnan et al. [29] selected features for Type II diabetes using a Bays classifier based wrapper model. Guyon et al. [49] proposed an SVM-based gene wrapper model for cancer classification. In recent decades, a number of studies have reported using these two models for diagnostics. Guadrón et al. [28] proposed an SVM-based wrapper model for fault diagnosis in multi-sensor systems. Li et al. [50] used a filter model to select features for gear fault diagnosis. Baydar et al. [51] employed a PCA-based filter model for detecting localized faults in gearboxes. John et al. [52] reported that wrapper models outperform filter models with regard

to predictive power on unseen data. Since, however, wrapper models need to run classification many times in order to evaluate every feature, filter models are usually faster.

The second step is feature selecting which refers to selecting features based on their rank. Two schemes are usually used for this step: the forward selecting scheme and the backward selecting scheme. Given that a rank of features is available in which the most useful feature is ranked highest and the most useless feature is ranked lowest, the forward selecting scheme adds the top ranked features one by one to an empty feature set, and the backward selecting scheme eliminates the lowest ranked features one by one from an original feature space. A criterion, usually classification accuracy, is then applied to stop this process. Guadrón et al. [28] used the forward selecting scheme in an SVM-based feature selection for multi-sensor systems. Qu et al. [53, 54] used backward selecting scheme for classifying damage level in a slurry pump. The forward selecting scheme requires more perfect ranking, while the backward selecting scheme relies less on rank quality. The forward selecting scheme can ensure the features selected are all useful but it cannot ensure all useful features are selected, whereas the backward selecting scheme is able to retain most useful features but it may leave a few useless features in the final feature subset.

2.3 Diagnostics Using Data-Driven Approaches

As mentioned in Section 1.2, data-driven approaches encompass statistical methods and artificial intelligence (AI) methods. They both employ a parametric model the parameters of which are determined based on a set of data having inputs that correspond to their outputs. Once the model is built using the obtained parameters, the outputs can be computed according to the given inputs. The process of building the model is known as training or supervised-learning. The data used for training are called training data. A large number of both statistical and AI methods have been reported. For the sake of thesis length, only several typical ones are reviewed.

Actually, there is another broad category called unsupervised-learning methods where no training process is required before classification. A typical example is clustering analysis [55]. As opposed to the unsupervised-learning methods, the

supervised-learning methods have limitations when used in practice e.g. lack of training data, changes of the number of classes, etc. However, for the cases where historical information about system health, e.g. fault modes observed, fault levels measured, is available, the supervised-learning methods are able to provide better results than unsupervised-learning methods due to the knowledge learned from the historical information. For this reason, a tremendous number of studies are reported about using the supervised-learning methods for diagnostics (see [21–23, 56–62]). The supervised and unsupervised learning methods are two ways for solving the diagnostic problems. Which to be chosen depends on the data information available. A great number of methods have been reported for both supervised and unsupervised-learning methods. Unsupervised learning methods are not discussed in this thesis; however, it does not mean unsupervised-learning methods are not important.

2.3.1 Statistical Methods

For statistical methods, the parametric models are explicit and the parameters to be determined are usually interpretable. These parameters may be relevant to the inputs and outputs, e.g. hidden Markov model (HMM) or the features of inputs, e.g. logistic regression (LR) model. Some methods such as naive Bayes classifier (NBC) are directly derived from statistical theory. These three methods are reviewed below.

The HMM has two processes, a Markov chain with a finite number of states depicting an underlying mechanism and an observation process relying on the hidden state. A discrete-time HMM is defined by [7]:

$$X_{i+1} = AX_i + V_{i+1}, \tag{2.3}$$

$$Y_i = CX_i + W_i, \tag{2.4}$$

where X_i and Y_i represent the hidden process and the observation process, respectively, V_i and W_i are noise terms, and A and C are model parameters. Historical data are needed to train the HMM model; in other words, historical data are needed to obtain parameters which enable the model to describe the data appropriately.

Apparently, parameter A relates to the transition from one state to another, and parameter C represents the relationship between observations and hidden states. The HMM can be used to classify fault modes or levels in which X represents unknown observed features, and Y represents fault modes or levels. Nelwamondo et al. [63] classified early faults in bearings using HMM. Li et al. [64] used HMM to recognize faults of the speed-up and speed-down process in rotating machinery.

The LR model is a non-linear classification method which uses a set of samples from known classes to derive coefficients for an equation that calculates the probability of a new sample belonging to a certain class. This equation is written as [65]:

$$\text{Probability} = \frac{1}{1 + \exp[-(\beta_0 + \Sigma_i \beta_i X_i)]} \tag{2.5}$$

where β_0 is a constant term, β_i is the derived coefficient, and X_i is the value of the ith feature. Yan et al. [65] used LS model to identify normal and failure states in an elevator. Sutanto et al. [66] used LR model to analyze faults in an industrial printing process.

The NBC is based on the well-known Bayes rule. Assuming that Y is any discrete-valued class, and $X_i, i = 1, 2, \ldots, M$ are any discrete or real-valued features, NBC gives the probability that Y will take on its kth possible value, which, according to Bayes rule, is [67]

$$P(Y = y_k | X_1, \ldots, X_M) = \frac{P(Y = y_k) \prod_i^M P(X_i | Y = y_k)}{\Sigma_j^M P(Y = y_j) \prod_i^M P(X_i | Y = y_j)} \tag{2.6}$$

where the sum is taken over all possible values y_j of Y, assuming that the X_i are conditionally independent, given Y. Given a new sample $X^{\text{new}} = [X_1, \ldots, X_M]$ and the distributions $P(Y)$ and $P(X_i|Y)$ estimated from the training data, one can find the class of the new sample by:

$$Y \longleftarrow \text{argmax}_{y_k} P(Y = y_k) \prod_i P(X_i | Y = y_k) \tag{2.7}$$

where the denominator of Eq. (2.6) is removed because it does not depend on y_k. Maragoudakis et al. [68] detected faults in a gas turbine using naive Bayes classifier.

One drawback of statistical methods is that they may need assumptions before use, e.g. assumptions regarding data distribution, independency, etc. If inappropriate assumptions are made, the diagnostic results may be impaired.

2.3.2 Artificial Intelligence Methods

Unlike with statistical methods, the models of AI methods are usually implicit, and the parameters have no physical meaning. Basically, AI methods have two types of parameter, user-defined and training-data-defined. Here, the parameters are the latter. Artificial neural network (ANN) is the most widely used AI method. Its model is flexible and able to accommodate different methods to enhance classification results. Neuro-fuzzy inference system (NFIS) is a version of ANN hybridized with the knowledge-based fuzzy inference system (FIS). For ANN and NFIS, all training data are used to determine their models. SVM works in a way different from ANN and NFIS. It determines the model by relying on only a few selected training data points. For this reason, compared to other AI methods [11] SVM is regarded as possessing better generalization ability and superior for cases where only a small number of training data points are available. These three methods are reviewed below.

The ANN consists of processing elements connected in a layer structure. This structure allows the model to approximate an underlying non-linear function whose multiple inputs and outputs are known. The feed forward neural network (FFNN) in which the information moves only forward, from inputs, through hidden neurons, to outputs is the first and simplest type of ANN devised. Figure 2-2 shows the architecture of FFNN where the empty circles and the solid circles are called neurons.

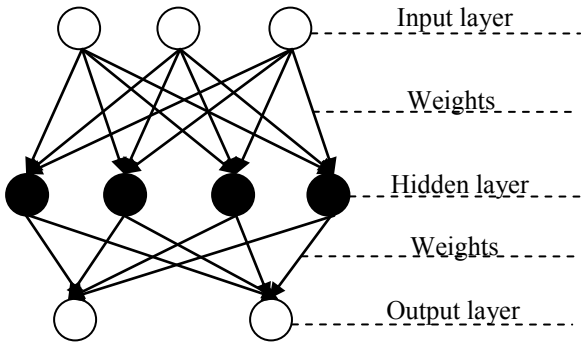

Figure 2-2: Architecture of FFNN

When the radial basis function (RBF) is used for the neurons in the hidden layer, the neural network in Figure 2-2 becomes a so-called RBF network. RBF networks typically have three layers, an input layer, a hidden layer with a non-linear RBF function and a linear output layer. The RBF function is defined as [69]:

$$\rho_i(\boldsymbol{x}) = \exp(-\beta_i \left\|\boldsymbol{x} - \boldsymbol{c}_i\right\|^2), i = 1, 2, \ldots, N, \tag{2.8}$$

where ρ_i represents the RBF function for the ith neuron, $\boldsymbol{x}$ represents a multi-dimensional input, N represents the number of neurons, $\boldsymbol{c}_i$ is the center of the ith neuron, and β_i is the weight coefficient. The output of neural network can thus be given by:

$$\varphi(\boldsymbol{x}) = \sum_{i=1}^{N} w_i \rho_i(\boldsymbol{x}), \tag{2.9}$$

where w_i is the linear weight coefficient for the ith neuron.

One widely used training method for neural network is back-propagation (BP). This method generates an error function based on the difference between predicted outputs and actual observations. During the training, the weights of each connection are continuously adjusted in order to reduce the difference. After repeating this process for a sufficiently large number of training cycles, the network will usually converge on some state where the error of the calculations is small.

Many studies report using ANN to classify fault modes or levels. Saravanan et al. [40] detected incipient faults in bevel gearboxes using FFNN. Dash et al. [56] developed an accurate fault mode classifier based on radial basis function (RBF)

neural network. Other advanced forms of ANN have also been reported. Chang et al. [57] proposed an enhanced probabilistic neural network for fault detection. Wu et al. [58] used ANN for classifying engine fault modes based on acoustic emission (AE) signals. Rafiee et al. [59] developed an algorithm for classifying gearbox fault modes using two-layer ANN. Wu et al. [60] used back-propagation neural network (BPNN) and generalized regression neural network (GRNN) for classifying fault modes in automotive generators.

The NFIS was proposed by Jang [61] where FIS was implemented in the framework of adaptive neural network. The adaptive network consists of nodes, and directional links through which nodes are connected. Moreover, part or all of the nodes are adaptive; the output depends on the parameters pertaining to these nodes, and the learning rule specifies how these parameters should be changed to minimize the prescribed measure of error. For simplicity, an adaptive neuro-fuzzy inference system (ANFIS) with two inputs, x and y, and one output, f, is considered below.

Suppose that the rule base contains two fuzzy if-then rules [61]:

Rule 1: If x is A_1 and y is B_1 then $f_1 = p_1x + q_1y + r_1$;

Rule 2: If x is A_2 and y is B_2 then $f_2 = p_2x + q_2y + r_2$;

where x and y are inputs, A_1, A_2, B_1, and B_2 are fuzzy sets to be determined by the training process, f_1 and f_2 are outputs, and p_1, q_1, r_1, p_2, q_2, and r_2 are linear parameters to be determined by the training process. The structure of ANFIS based on these two rules is shown in Figure 2-3, and the brief introduction of each layer is given below.

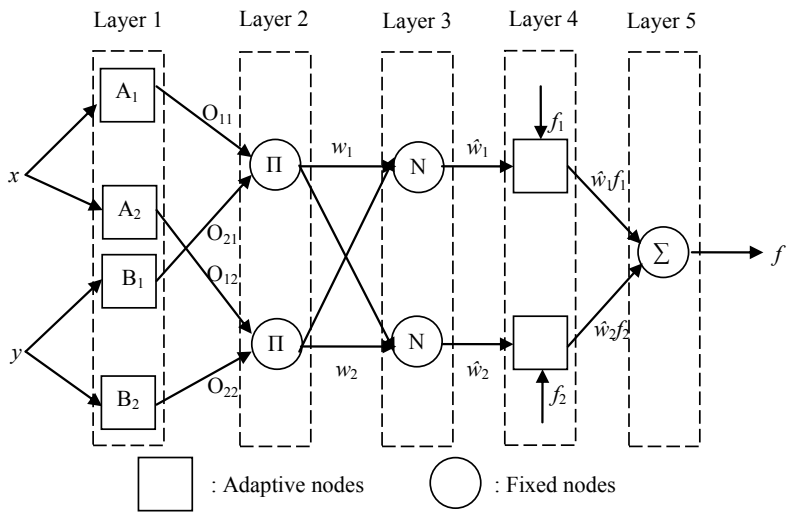

Figure 2-3: Structure of ANFIS

In layer 1, these nodes represent input nodes each of which is a linguistic label associated with the node function:

$$O_{1i} = \mu_{A_i}(x), i = 1, 2, \tag{2.10}$$

$$O_{2i} = \mu_{B_i}(y), i = 1, 2, \tag{2.11}$$

where O_{1i} and O_{2i} are membership functions with linguistic labels of A_i and B_i, respectively.

In layer 2, every node is a fixed node labeled Π which multiplies the incoming signals and sends the product out. For instance,

$$w_i = \mu_{A_i}(x)\mu_{B_i}(x), i = 1, 2, \tag{2.12}$$

where w_i is the output of the ith node and represents the firing strength of a rule.

In layer 3, every node is a fixed node labeled N. The ith node calculates the ratio of the ith rule's firing strength to the sum of all rules' firing strengths:

$$\hat{w}_i = \frac{w_i}{\sum_i w_i}, i = 1, 2, \tag{2.13}$$

where $\hat{w}_i$ is also called the normalized firing strength.

In layer 4, every node is an adaptive node with a node function:

$$\hat{w}_i f_i = \hat{w}_i(p_i x + q_i y + r_i), i = 1, 2, \tag{2.14}$$

where $\hat{w}_i$ is the output of layer 3, and p_i, q_i, and r_i are parameters defined in the two If-Then rules.

In layer 5, the single node is a fixed node labeled Σ that computes the overall output as the summation of all incoming signals. For instance,

$$\sum_i \hat{w}_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i}, i = 1, 2, \tag{2.15}$$

ANFIS then finds the optimal parameters that establish the appropriate relation between inputs and outputs.

Applications of ANFIS for diagnostics have been reported in recent years. Tran et al. [62] utilized neuro-fuzzy system for fault diagnosis of a methane compressor. Wu et al. [70] used ANFIS for gear fault mode classification using features extracted from WT. Nguyen et al. [71] used ANFIS in the fault mode classification of a transmission line.

As mentioned above, SVM has good ability in generalization and performs well for small sample cases, so it has attracted more and more attention in diagnostics in recent decades. Since SVM-based diagnostics is a focal point of this thesis, the theory of SVM for classification is presented in Chapter 3. Here are the related reported studies. Widodo et al. [11] reviewed SVM applications for classifying fault modes and levels in system diagnostics. Samanta [20] studied the performance of ANN and SVM for fault mode classification, and found that SVM outperformed ANN for all tested cases. He et al. [21] detected faults in valves from reciprocating pumps using SVM. Yuan et al. [22] developed a multi-class SVM classifier to classify fault modes in a turbo-pump rotor. Yuan et al. [23] developed a method based on SVM and artificial immunization algorithm (AIA) for classifying fault modes in a turbo-pump rotor.

AI methods can be used without any assumptions, so they are more favored for practical applications. Note, however, that they may be criticized as a black-box

approach because their models are implicit, and the parameters usually cannot be explained to users.

2.4 Prognostics Using Data-Driven Approaches

The goal of prognostics is to acquire the RUL for a system on which maintenance actions can be effectively implemented. Similar to diagnostics, data-driven approaches for prognostics also include statistical methods and AI methods. Statistical methods may obtain the point estimate of RUL in two ways, determining a system's lifetime distribution or determining the probability density function (PDF) of a system's RUL. AI methods obtain the point estimate of RUL by predicting the CI.

2.4.1 Statistical Methods

As mentioned, there are two ways of obtaining the point estimate of RUL. The first one estimates the lifetime distribution of a system, and then estimates the expectation of system life using the distribution. Once this is done, the point estimate of RUL is calculated by subtracting the current system age from the expectation. Lifetime distribution can be estimated using failure data or CM data. The methods using failure data are widely reported in reliability theory [72]. Some methods using CM data can be found in [73]. One drawback of these methods is that data from multiple samples are usually required. This may be impractical for complicated systems which are too expensive to damage.

The second way is to obtain the PDF of RUL. For this case, RUL is considered as a random variable. Let X_t represent the random variable of RUL at time t. The PDF of X_t conditional on Y_t can be represented by $f(X_t|Y_t)$ where Y_t represents the condition information up to time t. As long as the PDF of RUL is available, the expectation of RUL (a point estimate of RUL) can be easily estimated. In the following, two methods of this type are reviewed: the state-space model (SSM) and the proportional hazard model (PHM).

The SSM is one widely used statistical method in which an unobservable variable is defined as representing system operating states. The SSM model can be expressed

as [74]:

$$y_t = Z_t\alpha_t + d_t + S_t\epsilon_t, \tag{2.16}$$

$$\alpha_t = T_t\alpha_{t-1} + c_t + R_t\eta_t, \tag{2.17}$$

where ϵ_t is a serially uncorrelated disturbance with a mean of zero and a covariance matrix of H_t, and α_t is a state vector which is not observable and is assumed to be generated by the first-order Markov process as shown in Eq. (2.17). T_t is a transition matrix and η_t is a random vector of serially uncorrelated disturbances with a mean of zero and a covariance matrix of Q_t. The parameters of Z_t, d_t, S_t, T_t, c_t and R_t depend on information available at $t-1$ if normality for errors ϵ_t and η_t is assumed. Eq. (2.16) is known as the measurement equation while Eq. (2.17) is called the transition equation. The SSM model requires the following two additional assumptions:

1. the initial vector, α_0, has a mean of a_0 and a covariate matrix of Σ_0, i.e. $E[\alpha_0] = a_0$ and $Var[\alpha_0] = \Sigma_0$.

2. the disturbances, ϵ_t and η_t, are uncorrelated with each other for all time periods, and uncorrelated with the initial state variable, i.e. $E[\epsilon_t\eta_t] = 0$ and $E[\epsilon_t\alpha_0^{\mathrm{T}}] = E[\eta_t\alpha_0^{\mathrm{T}}] = 0$, for all t.

Given the information up to time s, i.e. $(y_1, \ldots, y_s)$, the conditional expectation of α_t can be denoted by $E[\alpha_t|(y_1, \ldots, y_s)] \equiv a_{t|s}$, and the conditional covariance matrix of α_t can be denoted by $Cov[\alpha_t|(y_1, \ldots, y_s)] \equiv \Sigma_{t|s}$. The evaluation of $a_{t|s}$ is known as filtering if $t = s$, smoothing if $t < s$, and predicting if $t > s$. The filtering problem can be addressed by the well-known Kalman filter [74].

Some work report using SSM for the estimation of RUL. Most of them incorporated the PDF with a cost model on which the CBM decision could be made. Christer et al. [8] adopted the SSM for predicting furnace erosion and replacement. Lu et al. [9] employed a similar state model to determine the PDF that was used in a cost model for CBM decision making. Zhou et al. [75] developed a Gamma-based SSM for predicting the life time of liquified natural gas pumps.

Another popular method is time-dependent PHM which analyzes event data and CM data together. The merit of a time-dependent PHM is its ability to relate the

probability of failure to both age and condition indicators, so that one can assess failure probability with any given system condition at any specified age [7]. The PHM assumes that the failure rate is the product of a baseline failure rate dependent only on the age of the unit and a positive function, $\varphi(\cdot)$, dependent on the values of the stochastic process Z. Z reflects the effect of the working environment on the system; therefore, the survivor function is given by [10]:

$$P(T > t|Z_s, 0 \leq s \leq t) = exp(-\int_0^t h(s, Z_s)ds), t \geq 0, \tag{2.18}$$

where T is the time to failure of the system, and $h(s, Z_s)$ is the failure rate at time t which can be expressed as:

$$h(s, Z_s) = h_0(s)\varphi(Z_s), s > 0. \tag{2.19}$$

The PHM is usually incorporated with a cost model for CBM decision making. Makis et al. [10] proposed a model for optimal replacement based on PHM. In their model, it is assumed that the failure rate of a system is a function of age but can also depend on the values for condition indicators describing the effect of the environment in which it operates. Banjevic et al. [76] developed a model for the optimization of CBM decisions using PHM with a Weibull baseline hazard function, and a time-dependent stochastic covariates are used to describe the failure rate of system.

Statistical methods can also be incorporated into AI methods to directly provide a point estimate of RUL. In this case, because the parameters of statistical methods are obtained using CM data, they can carry useful information about system health; hence, these parameters can be used as condition indicators to represent system health. This kind of problem has not been widely studied. Zhang et al. [77] used HMM to generate a health index which is the input to an adaptive prognostic component for point estimation of RUL. Yan et al. [65] adopted ARMA model to provide useful condition indicators for a logistic regression model for estimating RUL.

Statistical methods can not only provide the point estimate of RUL, but the probability of a system having a certain RUL value at a certain time. This prob-

abilistic measure can be incorporated with maintenance concerns such as costs for making maintenance decisions. Nevertheless, the statistical methods are rather complicated and are subject to some assumptions which may not be convenient to use.

2.4.2 Artificial Intelligence Methods

AI methods aim to obtain a point estimate of RUL. The point estimate of RUL is defined as the length of time between current time and system failure. Figure 2-4 exhibits the process of obtaining the point estimate of RUL where t_{c} represents current time, y_{th} represents a pre-specified threshold of the CI value indicating the failure of the system, and t_{th} represents the time corresponding to y_{th}. The threshold value is usually determined relying on expert knowledge and experience. From the viewpoint of maintenance, the threshold correlates with not only the health states of the system but also the costs of implementing maintenance.

AI methods obtain the RUL through addressing a time series prediction problem. The CI values are first observed or extracted from online CM data up to t_{c}; these values are then used to train the parametric model in AI methods. When the model is established, the CI value for a future time of interest can be predicted. When the predicted value exceeds the threshold, y_{th}, the point estimate of RUL can be computed using $t_{\mathrm{th}} - t_{\mathrm{c}}$.

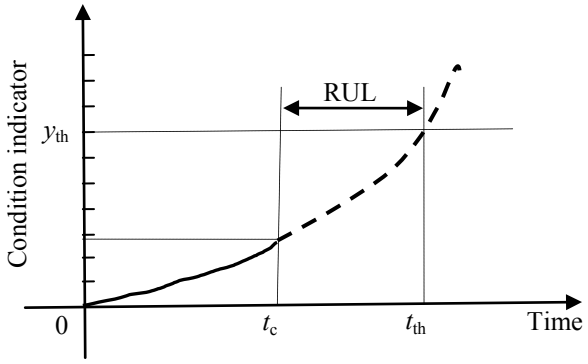

Figure 2-4: Point estimate of RUL

AI methods reported for diagnostics can also be used for prognostics. Samanta et al. [19] studied the performance of ANFIS in the prognosis of gearbox conditions. Tse et al. [37] used the amplitude of a vibration signal as a CI and predicted its value for prognostics using recurrent neural networks. Wang et al. [42] used NFIS for the prognosis of gearbox conditions. Greitzer et al. [78] proposed a model for RUL estimation of gas turbine engine health prognostics using ANN.

Some studies have been reported for SVM-based prognostics in recent years. Though the number is limited, the results are quite promising. Samanta et al. [19] adopted ANFIS and SVM for gearbox condition prognostics, and the results showed that SVM was better than NFIS for the tested problems. Chen [24] and Pai [25] proposed respectively an algorithm based on genetic algorithm and SVM for prognostics of system reliability. Kim et al. [26] proposed an SVM-based method to obtain the point estimate of RUL for bearings. Hong et al. [27] used SVM for engine condition prognostics. The theory of SVM for prediction is covered in detail in the next chapter, because SVM-based prognostics is another focal point of this thesis.

2.5 Summary

This chapter reviews some typical data-driven approaches for both diagnostics and prognostics. The diagnostics and prognostics contain the same two stages of data acquisition and data processing. The rationales for these two stages are presented, and some reported work is reviewed. For the diagnostics and prognostics stages, statistical and artificial intelligence methods of data-driven approaches are reported. The advantages and disadvantages of these two types of method are commented on, and some typical methods are reviewed. SVM, as the focus of this thesis, is also mentioned in the section on artificial intelligence methods. The relevant reported studies are reviewed, but the SVM theory is presented in the next chapter.

Chapter 3

Support Vector Machine Fundamentals

In this chapter, the fundamentals of support vector machine (SVM) are presented. SVM is a learning system that uses a hypothetical space of linear functions in a high dimensional feature space, trained with a learning algorithm from optimization theory that implements a learning bias derived from statistical learning theory. SVM can be used in the realms of classification and regression. For ease of use, this chapter uses shortened names SVC for support vector classification and SVR for support vector regression. Fundamentals of SVC and SVR are separately introduced in this chapter.

3.1 Support Vector Classification

Regular SVC is applicable only to binary classification problems. For problems involving multiple classes, several approaches can be incorporated with SVC to achieve multi-class classification problems. In this section, the theory of binary SVC is first introduced. After that, two multi-class strategies, one-against-all (OAA) and one-against-one (OAO), are introduced for multi-class SVC.

3.1.1 Binary Support Vector Classification

Consider a problem of binary classification where training data are given as $\{(\boldsymbol{x}_1, y_1); (\boldsymbol{x}_2, y_2); \ldots; (\boldsymbol{x}_M, y_M)\}, \boldsymbol{x}_i \in \mathbb{R}^m$ and $y_i \in \{1, -1\}, (i = 1, 2, \ldots, M)$. The labels of 1 and -1 are used to represent the two classes of data points. If the training data

are linearly separable, there exists a separating plane (SP) in the input space; this can be expressed as

$$f(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b = \sum_{j=1}^{m} w_j x_j + b = 0, \tag{3.1}$$

where $\boldsymbol{w} \in \mathbb{R}^m$ is a weight vector, b is a scalar, and T means the transpose operator. The parameters of $\boldsymbol{w}$ and b define the location of the SP and are determined during the training process.

In SVC theory, training data points satisfying the constraints that $f(\boldsymbol{x}_i) = 1$ if $y_i = 1$ and $f(\boldsymbol{x}_i) = -1$ if $y_i = -1$ are called support vectors. Other training data points satisfy the inequalities that $f(\boldsymbol{x}_i) > 1$ if $y_i = 1$ and $f(\boldsymbol{x}_i) < -1$ if $y_i = -1$. As a result, a complete form of constraints for all training data can be represented as

$$y_i f(\boldsymbol{x}_i) = y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b) \geq 1 \text{ for } i = 1, 2, \ldots, M. \tag{3.2}$$

Figure 3-1 illustrates a linearly separable classification problem in a two-dimensional space. In this figure, all squares are labeled -1 while all circles are labeled $+1$. The solid squares and the solid circles are called support vectors. The two dash lines represent two parallel planes where one crosses all solid squares and the other crosses all solid circles. These two parallel planes are called boundaries. The distance between the boundaries is called the margin.

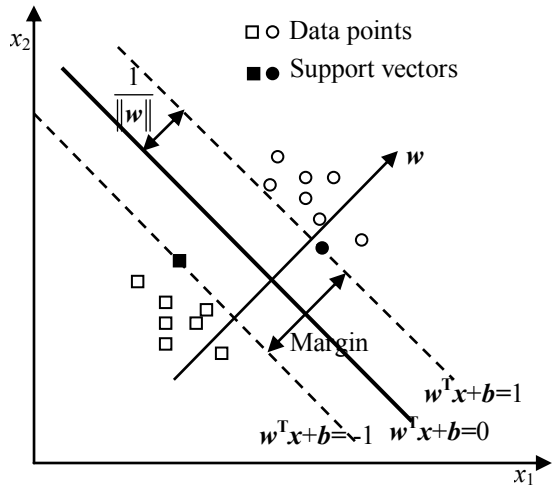

Figure 3-1: Examples of a linearly separable classification problem in $\mathbb{R}^2$

The quantity of margin can be computed by the distance between two parallel planes. Given two parallel planes, $\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + \boldsymbol{b} = +1$ and $\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + \boldsymbol{b} = -1$, the distance can be obtained by:

$$D = \left| \frac{(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b - 1) - (\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b + 1)}{\|\boldsymbol{w}\|} \right| = \frac{2}{\|\boldsymbol{w}\|}. \tag{3.3}$$

It is intuitive that many pairs of such parallel planes could exist. In SVC theory, one searches for the pair that provides the largest margin value. As long as the maximal margin is obtained, an optimal SP, the solid straight line in Figure 3-1, can be determined as the plane that is parallel to and equally distant from these two parallel planes. More details can be found in [79]. Because the margin can represent the separation of data points, its values could be used to assess feature usefulness. In Chapter 5, a feature selection method is presented based on changes in margin value .

To acquire the optimal SP, SVC adopts an optimization process that maximizes margin and minimizes noise using slack variables, ξ_i.

$$\text{Minimize} \quad \frac{1}{2}\|\boldsymbol{w}\|^2 + C\Sigma_{i=1}^{M}\xi_i, \tag{3.4}$$

$$\begin{aligned} \text{Subject to} \quad & y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b) \geq 1 - \xi_i, \quad i = 1, 2, \ldots, M, \\ & \xi_i \geq 0, \qquad\qquad\qquad\quad i = 1, 2, \ldots, M, \end{aligned} \tag{3.5}$$

where C is a positive constant and ξ_i represents the distance between a data point, $\boldsymbol{x}_i$, lying on the false class side and the plane in its virtual class side. This optimization problem can be solved by introducing the Lagrange multipliers, α_i and β_i.

$$L(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{1}{2}\|\boldsymbol{w}\|^2 + C\Sigma_{i=1}^{M}\xi_i - \Sigma_{i=1}^{M}\alpha_i(y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b) - 1 + \xi_i) - \Sigma_{i=1}^{M}\beta_i\xi_i, \tag{3.6}$$

where $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_M)^{\mathrm{T}}, \boldsymbol{\beta} = (\beta_1, \ldots, \beta_M)^{\mathrm{T}}$, and $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_M)^{\mathrm{T}}$.

For the optimal solution, the derivatives of the Lagrange function with respect to $\boldsymbol{w}$, b and $\boldsymbol{\xi}$ should vanish, that is,

$$\frac{\partial L}{\partial \boldsymbol{w}} = 0 \Longrightarrow \boldsymbol{w} = \Sigma_{i=1}^{M}\alpha_i y_i \boldsymbol{x}_i,\ i = 1, 2, \ldots, M, \tag{3.7}$$

$$\frac{\partial L}{\partial b} = 0 \Longrightarrow \Sigma_{i=1}^{M}\alpha_i y_i = 0,\ i = 1, 2, \ldots, M, \tag{3.8}$$

$$\frac{\partial L}{\partial \boldsymbol{\xi}} = 0 \Longrightarrow \alpha_i + \beta_i = C,\ i = 1, 2, \ldots, M. \tag{3.9}$$

Substituting Eqs. (3.7) and (3.8) into Eq. (3.6) yields:

$$L(\boldsymbol{\alpha}) = \Sigma_{i=1}^{M}\alpha_i - \frac{1}{2}\Sigma_{i,j=1}^{M}\alpha_i\alpha_j y_i y_j \boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j. \tag{3.10}$$

The primal minimization problem of Eqs. (3.4) and (3.5) can be transformed into the dual optimization problem of maximizing Eq. (3.10) subject to the following constraints:

$$\Sigma_{i=1}^{M} y_i\alpha_i = 0,\ i = 1, 2, \ldots, M, \tag{3.11}$$

$$C \geq \alpha_i \geq 0,\ i = 1, 2, \ldots, M. \tag{3.12}$$

By solving the dual optimization problem, one obtains the coefficients α_i which are required to express $\boldsymbol{w}$. Following the Karush-Kuhn-Tucker (KKT) condition, the products of the dual variables and the constraints should be equal to zero at the solution point:

$$\alpha_i(y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b) - 1) = 0,\ i = 1, 2, \ldots, M. \tag{3.13}$$

According to Eqs. (3.7) and (3.13), the expressions of $\boldsymbol{w}$ and b are given by:

$$\boldsymbol{w} = \Sigma_{i=1}^{M} y_i \alpha_i \boldsymbol{x}_i, \tag{3.14}$$

$$b = \frac{1}{p}\Sigma_{j=1}^{p}(y_j - \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_j), \tag{3.15}$$

where p is the number of support vectors and b has an expression only when α_i is non-zero for the support vectors. The advantage of SVM with regard to using limited amount of data for its model establishment can be seen from Eq. (3.14). This is the expression of $\boldsymbol{w}$, the coefficient of the underlying dependency. The $\boldsymbol{w}$ is dependent on the sum of the product of y_i, α_i, and $\boldsymbol{x}_i$. The α_i is non-zero for only support vectors, so it mathematically shows that SVM uses only a certain amount of data points for building its model representing the underlying dependency. Once $\boldsymbol{w}$ and b are available, the linear decision function can be given by:

$$i_f = \mathrm{sign}(\Sigma_{i=1}^{M} \alpha_i y_i (\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}) + b), \tag{3.16}$$

where if i_f is positive, a new input data point, $\boldsymbol{x}$, belongs to class 1 ($y_i = 1$) and if i_f is negative, $\boldsymbol{x}$ belongs to class 2 ($y_i = -1$).

When the given data are not linearly separable, using Eq. (3.16) is no longer appropriate. In SVC theory, this problem is addressed by introducing a mapping which projects the original input data onto a high dimensional feature space in which the input data can be linearly separated. Figure 3-2 gives an example of feature mapping from a two-dimensional input space to a two-dimensional feature space using a mapping, $\boldsymbol{\Phi}(\cdot)$. As a result, data points that can not be separated by a linear function in the input space can be linearly separated in the feature space.

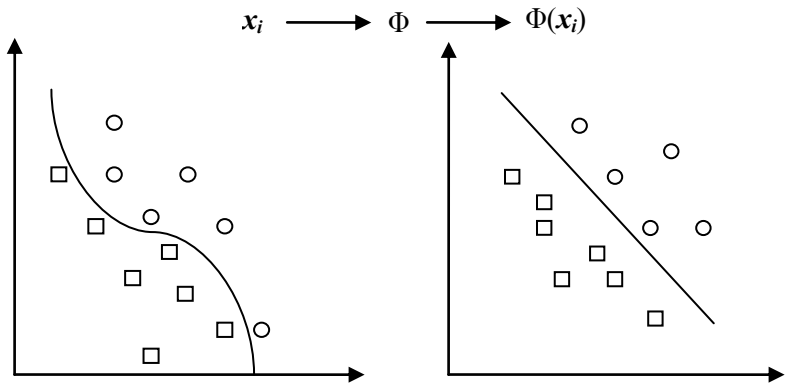

Figure 3-2: Example of feature mapping enabling linear data separation

Adopting the mapping allows the linear model of Eq. (3.16) to remain valid for non-linear cases. Accordingly, the non-linear function can be given by:

$$i_f = \text{sign}(\Sigma_{i=1}^{M} \alpha_i y_i (\mathbf{\Phi}^{\mathrm{T}}(\boldsymbol{x}_i)\mathbf{\Phi}(\boldsymbol{x})) + b), \tag{3.17}$$

where $\mathbf{\Phi} : \mathbb{R}^m \longrightarrow \chi$ is a mapping function which transforms the original input space into a feature space. The fact is, however, that this mapping function is difficult or even impossible to compute. Fortunately, one can avoid computing it by using kernel functions, since only the dot product of the mapping functions is needed in Eq. (3.17). A kernel function, represented by $K(\boldsymbol{x}_i, \boldsymbol{x}) = \mathbf{\Phi}^{\mathrm{T}}(\boldsymbol{x}_i)\mathbf{\Phi}(\boldsymbol{x})$, can return a dot product of the feature space mappings of the original inputs, which makes the explicit form of $\mathbf{\Phi}(\cdot)$ no longer necessary. The non-linear decision function can thus be given by:

$$i_f = \text{sign}(\Sigma_{i=1}^{M} \alpha_i y_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b). \tag{3.18}$$

Any function that satisfies Mercer's theorem [80] can be used as a kernel function. Many kernel functions are off the shelf. The selection of kernel function is dependent on the distribution of the data which, however, is often difficult to know if prior knowledge of the data is not available. This is difficult even for the data in a high-dimension the distribution of which can not be visualized. A commonly-used method for kernel selection is the "trials and errors" test. For example, one can assess different kernel functions in terms of their accuracy of classification. The

kernel function to be selected is the one offering the highest classification accuracy. Fortunately, this test is not computationally expensive, since Gaussian kernel and polynomial kernel are often adequate for most applications [11, 81]. In this thesis, kernel function and kernel parameter are denoted as k_f and k_p, respectively. For example, Gaussian kernel is expressed as:

$$k_f = K(\boldsymbol{x}_i, \boldsymbol{x}) = \exp(\frac{\left\|\boldsymbol{x} - \boldsymbol{x}_i\right\|^2}{2\sigma^2}) \tag{3.19}$$

where σ represents the width parameter, denoted as $k_p = \sigma$. Polynomial kernel is expressed as:

$$k_f = K(\boldsymbol{x}_i, \boldsymbol{x}) = (\boldsymbol{x}^{\mathrm{T}}\boldsymbol{x}_i + c)^d \tag{3.20}$$

where c is a constant term and d represents the polynomial degree, denoted as $k_p = [c, d]$.

3.1.2 Multi-class Support Vector Classification

3.1.2.1 One-Against-All (OAA)

The OAA strategy virtually transforms an N-class classification problem into N binary classification problems. For an SVC-based OAA approach, N SVC models are established using Eqs. (3.4) and (3.5). The j^{th} $(j = 1, 2, \ldots, N)$ SVC model is trained using all training data in which data points belonging to the j^{th} class carry the positive label and the rest carry the negative label. After a successful training of all N SVC models, a new data point, $\boldsymbol{x}$, is said to belong to the class that has the largest value for the following indicator function [11]:

$$\begin{aligned} \text{Class of } \boldsymbol{x} &\equiv \text{argmax}_{j=1,2,\ldots,N}((\boldsymbol{w}^{(j)})^{\mathrm{T}}\boldsymbol{\Phi}(\boldsymbol{x}) + b^{(j)}) \\ &= \text{argmax}_{j=1,2,\ldots,N}(\Sigma_{i=1}^{M}\alpha_i^{(j)}y_i^{(j)}K^{(j)}(\boldsymbol{x}_i, \boldsymbol{x}) + b^{(j)}) \end{aligned} \tag{3.21}$$

where $\boldsymbol{w}^{(j)}$ and $b^{(j)}$ represent the parameters of the j^{th} SVC model.

3.1.2.2 One-Against-One (OAO)

The OAO strategy consists of constructing one SVC for each pair of classes. Thus, for a problem with N classes, the number of $N(N-1)/2$ SVCs are trained to distinguish data points of one class from those of another class. An unknown data point is classified according to the maximal number of votes, where each SVC votes for one class [11].

3.2 Support Vector Regression

An SVR model can be obtained by minimizing the following expression:

$$R_{\mathrm{SVR}} = R_{\mathrm{STR}} + R_{\mathrm{EMR}}, \tag{3.22}$$

where R_{SVR} represents the regression risk of SVR, R_{STR} represents the structural risk (model complexity), and R_{EMR} represents the empirical risk (estimation error) assessed by loss functions.

Simultaneously minimizing empirical risk and model complexity enables SVR to have good generalization ability and makes it less prone to over-fitting. The type of SVR depends on the loss functions employed. For example, standard SVR employs the ϵ-insensitive loss function; another commonly used SVR model called least square SVR (LSSVR) employs the squared loss function. This section introduces the loss function as well as the two mentioned SVRs because they are used in following chapters.

3.2.1 Loss Function

Under the assumption that a given set of samples, $\{(\boldsymbol{x}_1, y_1); (\boldsymbol{x}_2, y_2); \ldots; (\boldsymbol{x}_M, y_M)\} \subset \mathbb{R}^m \times \chi$, can be generated by an underlying functional dependency plus additive noise, $y_i = f(\boldsymbol{x}_i) + \eta_i$, for the density model of $p(\eta_i)$. The optimal loss function for the samples can be obtained using the maximum likelihood method:

$$\begin{aligned} \hat{f} &= \mathrm{argmax}_f(p(\eta_1, \ldots, \eta_M)), \\ &= \mathrm{argmax}_f(p(y_1 - f(\boldsymbol{x}_1), \ldots, y_M - f(\boldsymbol{x}_M))), \end{aligned} \tag{3.23}$$

where $\hat{f}$ represents the optimal underlying functional dependency.

If it is assumed that the additive noise is independently and identically distributed, Eq. (3.23) can be equally represented as:

$$\hat{f} = \mathrm{argmax}_f(\prod_{i=1}^{M} p(y_i - f(\boldsymbol{x}_i))). \tag{3.24}$$

Since maximizing a product is equivalent to minimizing a negative logarithm of the product, Eq. (3.24) can be expressed as:

$$\hat{f} = \text{argmin}_f(-\sum_{i=1}^{M} \log[p(y_i - f(\boldsymbol{x}_i))] = \text{argmin}_f(\sum_{i=1}^{M} c(y_i - f(\boldsymbol{x}_i)), \tag{3.25}$$

where c represents an optimal loss function or cost function.

Based on the density models reported in [82], the commonly used ϵ-insensitive and squared loss functions can be derived from:

$$p_\epsilon(\eta) = \frac{1}{2(1+\epsilon)} e^{|\eta|_\epsilon} \overset{\text{-log}}{\Longrightarrow} c_\epsilon(\eta) = |\eta|_\epsilon, \tag{3.26}$$

$$p_s(\eta) = \frac{1}{\sqrt{2\pi}} e^{\frac{1}{2}\eta^2} \overset{\text{-log}}{\Longrightarrow} c_s(\eta) = \frac{1}{2}\eta^2, \tag{3.27}$$

where p_ϵ and p_s represent the density models corresponding to the ϵ-insensitive function, c_ϵ, and the squared loss function, c_s, respectively. It is noted that p_s is virtually a Gaussian density function; hence, the squared loss function is optimal for data with Gaussian additive noise.

3.2.2 Standard SVR

Let us consider the samples in Section 3.2.1. If it is assumed that they can be linearly described, the following expression exists:

$$f(\boldsymbol{w}, \boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b, \boldsymbol{w} \in \mathbb{R}^m, b \in \chi. \tag{3.28}$$

In SVR theory, the R_{STR} term in Eq. (3.22) is represented by the squared norm of $\boldsymbol{w}$, namely $\|\boldsymbol{w}\|^2 = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{w}$. It is believed that a smaller such value corresponds to a lower complexity, in other words a flatter shape for $f(\boldsymbol{w}, \boldsymbol{x})$.

Standard SVR adopts the ϵ-insensitive loss function to determine tolerated errors between observations and predictions. This indicates that one is not concerned about the difference between the observations and the predictions that are smaller than ϵ but care about those larger than ϵ. In other words, SVR concerns only data

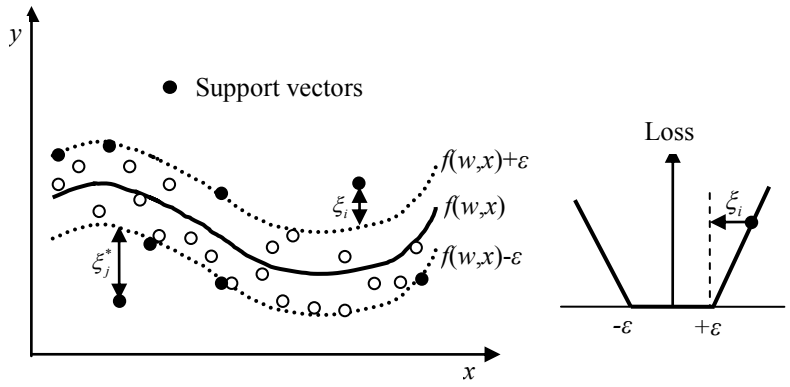

Figure 3-3: SVR with ϵ-insensitive loss function for one-dimensional data

points that are outside the ϵ-insensitive zone. Figure 3-3 shows an example of SVR with ϵ-insensitive loss function for one-dimensional data.

The ϵ-insensitive loss function is formulated as:

$$|y - f(\boldsymbol{w}, \boldsymbol{x})|_\epsilon = \begin{cases} 0 & \text{if } |y - f(\boldsymbol{w}, \boldsymbol{x})| \leq \epsilon, \\ |y - f(\boldsymbol{w}, \boldsymbol{x})| - \epsilon & \text{otherwise.} \end{cases} \tag{3.29}$$

The standard SVR model can thus be formulated as:

$$\text{Minimize} \quad \frac{1}{2}\|\boldsymbol{w}\|^2 + C\Sigma_{i=1}^{M}(\xi_i + \xi_i^*) \tag{3.30}$$

$$\text{Subject to} \quad \begin{cases} y_i - \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i - b & \leq \epsilon + \xi_i, \\ \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b - y_i & \leq \epsilon + \xi_i^*, \\ \xi_i, \xi_i^* & \geq 0, \end{cases} \tag{3.31}$$

where two non-negative slack variables, ξ_i and $\xi_i^*, i = 1, 2, \ldots, M$, are used to measure deviations outside the ϵ-insensitive zone, and parameter C is a positive constant called a regularization parameter which determines a compromise between the complexity and the amount up to which deviations larger than ϵ are tolerated. The parameters C and ϵ need to be pre-specified. Some studies reporting the selection of the parameters can be found in [24, 30, 83, 84].

The problem given above is a constrained optimization problem which can be solved using the Lagrange multiplier method; this is given by:

$$\begin{aligned}\text{Maximize}\quad L(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*) = \frac{1}{2}\|\boldsymbol{w}\|^2 + C\Sigma_{i=1}^{M}(\xi_i + \xi_i^*) \\ - \Sigma_{i=1}^{M}\alpha_i(\epsilon + \xi_i - y_i + \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b) \\ - \Sigma_{i=1}^{M}\alpha_i^*(\epsilon + \xi_i^* + y_i - \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i - b) \\ - \Sigma_{i=1}^{M}(\lambda_i\xi_i + \lambda_i^*\xi_i^*),\end{aligned} \tag{3.32}$$

where $\alpha_i, \alpha_i^*, \lambda_i$, and λ_i^* are Lagrange multipliers and are subject to non-negativity requirements. The conditions for optimality are given by:

$$\frac{\partial L}{\partial \boldsymbol{w}} = 0 \Longrightarrow \boldsymbol{w} = \Sigma_{i=1}^{M}(\alpha_i - \alpha_i^*)\boldsymbol{x}_i,\ i = 1, 2, \ldots, M, \tag{3.33}$$

$$\frac{\partial L}{\partial b} = 0 \Longrightarrow \Sigma_{i=1}^{M}(\alpha_i - \alpha_i^*) = 0,\ i = 1, 2, \ldots, M, \tag{3.34}$$

$$\frac{\partial L}{\partial \xi_i} = 0 \Longrightarrow C = \alpha_i - \lambda_i,\ i = 1, 2, \ldots, M, \tag{3.35}$$

$$\frac{\partial L}{\partial \xi_i^*} = 0 \Longrightarrow C = \alpha_i^* - \lambda_i^*,\ i = 1, 2, \ldots, M. \tag{3.36}$$

From Eq. (3.33), $\boldsymbol{w}$ can be expressed in terms of variables α_i and α_i^*. Substituting Eq. (3.34) into Eq. (3.32) makes b terms vanish. Substituting Eqs. (3.35) and (3.36) into Eq. (3.32) makes λ_i and λ_i^* vanish and results in only one term containing ξ_i and ξ_i^*, expressed as $C\Sigma_{i=1}^{M}(\xi_i + \xi_i^*)$. This term is a constant for a given set of data which does not affect maximization results, so it can be removed from the resultant objective function. As a result, the optimization problem becomes:

$$\begin{aligned}\text{Maximize}\quad & -\frac{1}{2}\Sigma_{i,j=1}^{M}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j - \epsilon\Sigma_{i=1}^{M}(\alpha_i + \alpha_i^*) \\ & + \Sigma_{j=1}^{M}y_j(\alpha_j - \alpha_j^*)\end{aligned} \tag{3.37}$$

$$\text{Subject to}\quad \begin{cases}\Sigma_{i=1}^{M}(\alpha_i - \alpha_i^*) = 0, \\ \alpha_i, \alpha_i^* \in [0, C].\end{cases} \tag{3.38}$$

The term b can be obtained according to the KKT condition on which the product of dual variables and constraints at the point of the optimal solution has to vanish:

$$\alpha_i(\epsilon + \xi_i - y_i + \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b) = 0, \tag{3.39}$$

$$\alpha_i^*(\epsilon + \xi_i^* + y_i - \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i - b) = 0, \tag{3.40}$$

$$(C - \alpha_i)\xi_i = 0, \tag{3.41}$$

$$(C - \alpha_i^*)\xi_i^* = 0. \tag{3.42}$$

Only the data points corresponding to $\alpha_i = C$ or $\alpha_i^* = C$ lie outside the ϵ-insensitive zone and yield a non-zero coefficient, $\boldsymbol{w}$. Such data points are referred to as support vectors. This determines the advantage of SVM about using limited number of data points for establishing its model, which is similar to that discussed for SVR above. As well, $\alpha_i\alpha_i^* = 0$ states that no pairs of α_i and α_i^* can be non-zero at the same time; this yields:

$$b = y_i - \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i - \epsilon, \text{ for } 0 \leq \alpha_i \leq C, \tag{3.43}$$

$$b = y_i - \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i - \epsilon, \text{ for } 0 \leq \alpha_i^* \leq C. \tag{3.44}$$

Therefore, the optimal regression function can be represented as:

$$f(\boldsymbol{x}) = \sum_{i=1}^{M}(\alpha_i - \alpha_i^*)\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x} + b. \tag{3.45}$$

When the given samples can not be linearly represented, Eq. (3.45) can still be made valid by introducing a mapping function, $\boldsymbol{\Phi}(\cdot)$, (the same as that of SVC) on which the primal input data are projected onto a high dimensional feature space in which they can be linearly represented. Hence, the regression model can be formulated as:

$$f(\boldsymbol{x}) = \sum_{i=1}^{M}(\alpha_i - \alpha_i^*)\boldsymbol{\Phi}^{\mathrm{T}}(\boldsymbol{x}_i)\boldsymbol{\Phi}(\boldsymbol{x}) + b. \tag{3.46}$$

Kernel function, $K(\boldsymbol{x}_k, \boldsymbol{x}_j) = \boldsymbol{\Phi}^{\mathrm{T}}(\boldsymbol{x}_k)\boldsymbol{\Phi}(\boldsymbol{x}_j), k, j = 1, \ldots, M$, is then used, and the following regression model is obtained:

$$f(\boldsymbol{x}) = \sum_{i=1}^{M}(\alpha_i - \alpha_i^*)K(\boldsymbol{x}_i, \boldsymbol{x}) + b, \tag{3.47}$$

where the term b can be computed based on Eqs. (3.43) and (3.44).

3.2.3 Least Square SVR (LSSVR)

Unlike standard SVR, LSSVR is based on a squared loss function and is optimal for predicting data containing Gaussian additive noise. The LSSVR model is formulated as:

$$\text{Minimize} \quad J(\boldsymbol{w}, \boldsymbol{e}) = \frac{1}{2} \|\boldsymbol{w}\|^2 + \frac{1}{2} C \sum_{i=1}^{M} e_i^2 \tag{3.48}$$

$$\text{Subject to} \quad y_i = \boldsymbol{w}^{\mathrm{T}} \boldsymbol{\Phi}(\boldsymbol{x}_i) + b + e_i, \ i = 1, 2, \ldots, M, \tag{3.49}$$

where a mapping of $\boldsymbol{\Phi}(\cdot)$ is directly employed to allow LSSVR to work for non-linear cases.

The constraints in Eq. (3.49) are all equality constraints which are easy to deal with. Introducing the Lagrange multiplier method into Eqs. (3.48) and (3.49) yields an unconstrained optimization problem:

$$\text{Maximize } L(\boldsymbol{w}, b, \boldsymbol{e}, \boldsymbol{\alpha}) = J(\boldsymbol{w}, \boldsymbol{e}) - \sum_{i=1}^{M} \alpha_i [\boldsymbol{w}^{\mathrm{T}} \boldsymbol{\Phi}(\boldsymbol{x}_i) + b + e_i - y_i]. \tag{3.50}$$

The conditions for optimality are given by:

$$\frac{\partial L}{\partial \boldsymbol{w}} = 0 \rightarrow \boldsymbol{w} = \sum_{i=1}^{M} \alpha_i \boldsymbol{\Phi}(\boldsymbol{x}_i), \ i = 1, 2, \ldots, M \tag{3.51}$$

$$\frac{\partial L}{\partial b} = 0 \rightarrow \sum_{i=1}^{M} \alpha_i = 0, \ i = 1, 2, \ldots, M, \tag{3.52}$$

$$\frac{\partial L}{\partial e_i} = 0 \rightarrow \alpha_i = C e_i, \ i = 1, 2, \ldots, M, \tag{3.53}$$

$$\frac{\partial L}{\partial \alpha_i} = 0 \rightarrow \boldsymbol{w}^{\mathrm{T}} \boldsymbol{\Phi}(\boldsymbol{x}_i) + b + e_i - y_i = 0, \ i = 1, 2, \ldots, M. \tag{3.54}$$

These conditions are similar to the optimality conditions of standard SVR except for the condition $\alpha_i = C e_i$. It is seen that every e_i is used for computing α_i which implies that all the data points are used for constructing the regression model. This, as a result, makes LSSVR lose the sparseness property.

By eliminating $\boldsymbol{w}$ and $\boldsymbol{e}$, one can obtain a matrix equation that contains only $\boldsymbol{\alpha}$ and b:

$$\begin{bmatrix} \mathbf{0} & \mathbf{1}^{\mathrm{T}} \\ \mathbf{1} & \boldsymbol{\Omega} + C^{-1}\mathbf{I} \end{bmatrix} \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \boldsymbol{y} \end{bmatrix}, \tag{3.55}$$

where $\boldsymbol{\Omega}_{k,j} = \boldsymbol{\Phi}^{\mathrm{T}}(\boldsymbol{x}_k)\boldsymbol{\Phi}(\boldsymbol{x}_j), k, j = 1, \ldots, M, \boldsymbol{y} = [y_1; \ldots; y_M], \boldsymbol{\alpha} = [\alpha_1; \ldots; \alpha_M]$, and $\mathbf{1} = [1; \ldots; 1]$. When $\boldsymbol{\Omega} + C^{-1}\mathbf{I}$ is symmetric and positive definite, its inverse exists. Hence, the solution of Eq. (3.55) can be represented by:

$$\boldsymbol{\alpha} = (\boldsymbol{\Omega} + C^{-1}\mathbf{I})^{-1}(\boldsymbol{y} - b\mathbf{1}), \tag{3.56}$$

$$b = \frac{\mathbf{1}^{\mathrm{T}}(\boldsymbol{\Omega} + C^{-1}\mathbf{I})^{-1}\boldsymbol{y}}{\mathbf{1}^{\mathrm{T}}(\boldsymbol{\Omega} + C^{-1}\mathbf{I})^{-1}\mathbf{1}}. \tag{3.57}$$

Using kernel function, the LSSVR model can be represented as:

$$f(\boldsymbol{x}) = \sum_{i=1}^{M} \alpha_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b. \tag{3.58}$$

where $K(\boldsymbol{x}_i, \boldsymbol{x}) = \boldsymbol{\Phi}^{\mathrm{T}}(\boldsymbol{x}_i)\boldsymbol{\Phi}(\boldsymbol{x}), i = 1, \ldots, M$.

3.3 Summary

This chapter presents the theory of SVC. SVC can be used only for binary classification problems; this means only two classes of data points can be classified. Fortunately, one can extend SVC to multi-class SVC using OAA or OAO in order to deal with multiple classes. Chapter 5 presents a diagnostic algorithm using both SVC and OAA-based multi-class SVC. The theories of SVR and LSSVR are also presented in this chapter. These two can be used for prediction, however, LSSVR is superior when Gaussian noise are involved in the data. Chapter 6 presents an online prognostic algorithm where SVR and LSSVR are used for prediction.

Chapter 4

Experimental Systems

This thesis proposes SVM-based algorithms for diagnostics and prognostics. In order to examine their effectiveness in industrial applications, two test rigs are employed to collect data . The two test rigs include one slurry pump and a planetary gearbox which were designed and established for collaborative research projects between Syncrude Research Center and Reliability Research Lab (Department of Mechanical Engineering, University of Alberta). This thesis uses some data, tables, and pictures of these two projects. The author would like to thank all the people who have made efforts on gathering and creating them. The two test rigs are used, because they can mitigate the possible inadequacy of the data collected from a single test rig. In addition, many features have been reported for gearbox diagnosis, so it allows to test the proposed feature selection algorithm which needs a large number of features to begin with. Slurry pump data do not have such an advantage. This chapter introduces the two test rigs, the details of experiments conducted, and the data collected for evaluating the proposed algorithms.

4.1 Planetary Gearbox Test Rig

Planetary gearboxes are widely used in Canada's oil sands industry and other heavy-duty industry such as helicopters, heavy-duty trucks, and other large-scale machinery. Planetary gearboxes are able to undertake heavy-duty tasks (high torque output) because they have multiple planet gears to share and carry loads. Unscheduled outages of planetary gearboxes have major economic consequences. Online assessment of the health condition of planetary gearboxes could generate significant cost

savings for these industries.

4.1.1 System Descriptions

A planetary gearbox test rig is designed to provide full capabilities of performing controlled experiments suitable for developing a reliable diagnostic system for planetary gearboxes. Figure 4-1 illustrates the test rig established the main components of which include one 20 HP drive motor, one stage of bevel gearbox, two stages of planetary gearbox, two stages of speed-up gearbox, and one 40 HP load motor. The drive motor is on the first foundation, the bevel gearbox and the planetary gearboxes are on the second foundation, and the two speed-up gearboxes and the load motor are on the third foundation. My research objective is the two stages of planetary gearboxes which have an over-hung floating configuration that mimic the support found in field planetary gearboxes at Syncrude's mining operations.

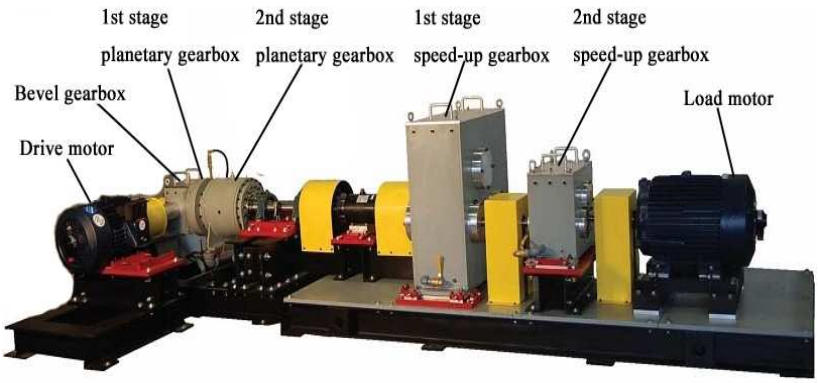

Figure 4-1: View of planetary gearbox test rig

Figure 4-2 shows a diagram of the structure of the two stages of planetary gearboxes. The 1st stage sun gear is mounted on the right side of shaft #1 with the driven bevel gear mounted on the left side. The 1st stage planet gears are located on the 1st stage carrier which is connected to shaft #2. The 2nd stage sun gear is mounted on the other side of shaft #2. The 2nd stage planet gears are mounted on the 2nd stage carrier located on output shaft #3. The ring gears of the 1st and 2nd stage are mounted on the housing of the corresponding stage. The two stages of planetary gearbox have different parameters for their gears. Table 4-1 lists the

number of teeth and the speed ratio achieved by each stage of the gearbox. For the speed-up ratio, the ratio value is calculated by the output speed divided by the input speed. For the speed reduction ratio, the ratio value is calculated by the input speed divided by the output speed.

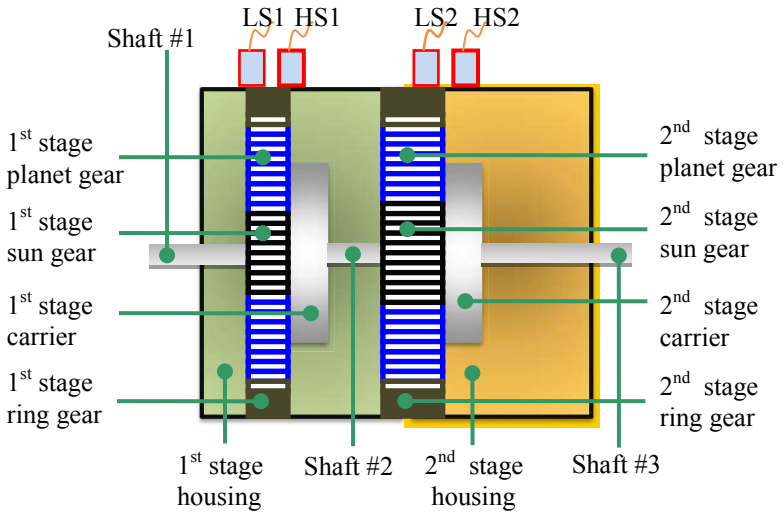

Figure 4-2: Diagram of two-stage planetary gearbox

Table 4-1: Number of teeth for two-stage planetary gearbox

	Bevel		1st planetary			2nd planetary			1st speed-up				2nd speed-up			
Gears	IB	OB	S	P	R	S	P	R	GI	SM	LM	GO	GI	SM	LM	GO
No.	18	72	28	62(3)	152	19	31(4)	81	72	32	80	24	48	18	64	24
Ratio	4↓		6.429↓			5.263↓			3.75↑				7.111↑			

Note: No.: number of gear teeth, IB: input bevel gear, OB: output bevel gear, S: sun gear, P: planet gear, R: ring gear, GI: gear on input shaft, SM: small gear on middle shaft, LM: large gear on middle shaft, GO: gear on output shaft, ↓: speed reduction ratio, and ↑: speed-up ratio. The number of planet gears is indicated in the parentheses.

Four accelerometers are located on the housing of the two-stage planetary gearbox; these consist of two identical low sensitivity accelerometers (LS1 and LS2) and two identical high sensitivity accelerometers (HS1 and HS2) as shown in Figure 4-2. The LS is miniature DeltaTron accelerometer, the manufacturer is Brüel & Kjaer, the model number is 4508B, the frequency range is 0.3 - 8,000 Hz, and the adhesive mounting clip is attached to the housing for securing the body of accelerometer. The HS is SEISMIC ICP accelerometer, the manufacturer is PCB Piezotronics, and the model number is 393B32, the frequency range is 0.2 - 200 Hz, and the adhesive

mounting is attached to the housing and the body of accelerometer is secured by screw. Figure 4-3 gives a view of the location of the four accelerometers and three speed-reduction gearboxes. Acoustic emission (AE) and particle counter data are also collected. This thesis focuses only on the analysis using vibration signals.

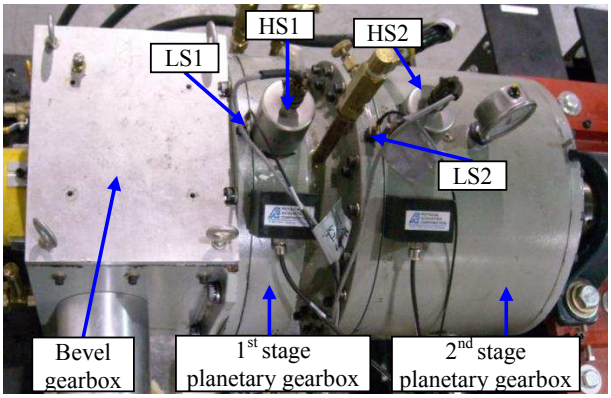

Figure 4-3: View of accelerometers and three reduction gearboxes locations

4.1.2 Manual Pitting Damage Experiments

Manual pitting damage experiments were designed and implemented to provide vibration data corresponding to different levels of gear pitting damage. The vibration data are the basis on which methods of classifying fault levels can be examined.

4.1.2.1 Damage Creation

Based on stress calculations [85], it has been found that planet gears on the 2nd stage planetary gearbox are more likely to suffer pitting damage during operations, so our focus was the pitting damage on these planet gears. Since it is difficult to govern the natural progression of pitting damage so as to obtain desired discrete damage levels, we chose to create pitting damage artificially. To mimic the pits observed on actual pitted gears, the following rationales were applied to manually creating various levels of pitting damage.

Slight level: Three holes (simulated pits) on one tooth and one hole on each of its two nearest neighboring teeth. The holes on the teeth are on the same mesh side. The percentage of pitted area, denoted as $A_{\text{Pit}}/A_{\text{Face}} \times 100\%$, is used to assess created holes where A_{Pit} represents the area of all created holes

and A_{Face} represnts the area of the tooth surface. We define the diameter of a hole as being 3 mm which gives $A_{\text{Pit}} = 7.07$ mm^2. The area of tooth surface is obtained from gear specifications, $A_{\text{Face}} = 267$ mm^2. In this way, the percentages of simulated pitted area are 2.65%, 7.95%, and 2.65% for the three teeth. This choice is in accordance with the definition of Level 2 given in the ASM Handbook which specifies a pitted area of 3-10% of the tooth area. This level of damage is used to mimic a minimal detectable level of pitting damage.

Moderate level: A total of 10 holes on one tooth, and 3 holes on each of its two nearest neighboring teeth, with one hole on each of its second nearest neighboring teeth. The most pitted tooth corresponds to ASM's level 3 pitting (15% - 40% of tooth area). The pitted areas of the 5 teeth are 2.65%, 7.95%, 26.5%, 7.95% and 2.65%.

Severe level: A total of 24 holes on one tooth, with 10 holes on each of its two nearest neighboring teeth, and 3 holes on each of its second nearest neighboring teeth. The most pitted tooth corresponds to ASM level 4 pitting (50% - 100% of tooth area). The pitted areas of the 5 teeth are 7.95%, 26.5%, 63.6%, 26.5% and 7.95%. This level of damage is used to mimic the last tolerable level of damage before major shutdown of the system occurs.

Figure 4-4 illustrates the schematic of the holes on a tooth surface where n represents a certain gear tooth number, $n+1$ and $n-1$ represent its nearest gear tooth numbers, and the numbers in parentheses represent the number of holes on the teeth from $n-2$ to $n+2$. Figure 4-5 gives views of the planet gears with four damage levels, baseline (top left), slight (top right), moderate (bottom left), and severe (bottom right). More details on creating pitting damage can be found in [86].

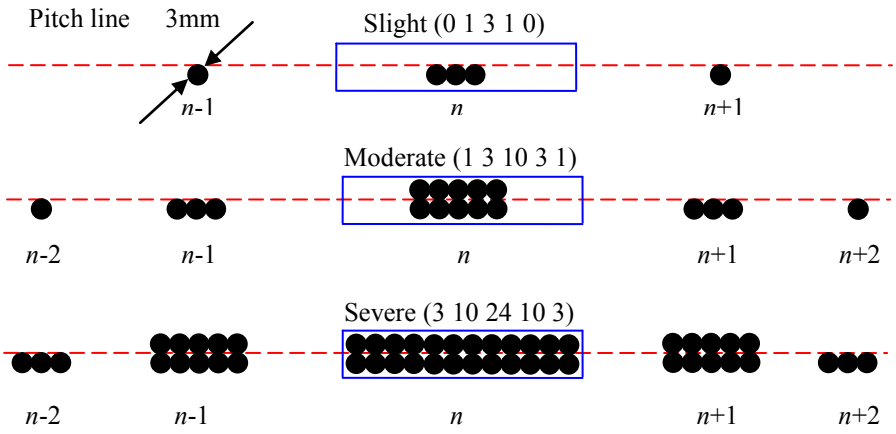

Figure 4-4: Simulation of pitting levels on planet gears

Figure 4-5: View of four artificially pitted planet gears

4.1.2.2 Implementations

There are four planet gears in the 2nd stage planetary gearbox (see Table 4-1). We installed three normal and one artificially damaged planet gears in the test rig and conducted the experiments. Upon finishing one experiment, we replaced the artificially damaged gear with one having another level of damage. This procedure was repeated until all four damage levels were tested.

For each of damage level, we conducted experiments on two separate days. For

each day, the load and the drive motor speed were varied. The two load conditions involved were "noload" and "load". For the load condition, constant torque of 10,000 lb·in was applied to the output shaft of the 2nd stage planetary gearbox. The four drive motor speeds applied were: 300, 600, 900, and 1200 revolutions per minute (rpm). Under each load condition, the drive motor speed was applied constantly one after another.

4.1.2.3 Data Collection

For each combination of loads and speeds, vibration data were collected from each of the four accelerometers with a sampling frequency of 10,000 Hz and a time span of 5 minutes. Each of these time-span data were further split into 10 segments of equal length. The segment length was determined to ensure that the lowest frequency component of interest, the carrier frequency from the 2nd stage planetary gearbox at 300 rpm, was covered. As a result, there are 80 segments (2 days × 4 damage levels × 10 segments) for each combination of loads and speeds. Table 4-2 lists the number of segments available for different conditions.

Table 4-2: Availability of data segments for manual pitting damage experiments

Exp. day	Load cond.	Speed	Baseline	Slight	Moderate	Severe
Day 1	No-Load	300 rpm	10	10	10	10
		600 rpm	10	10	10	10
		900 rpm	10	10	10	10
		1200 rpm	10	10	10	10
	Load	300 rpm	10	10	10	10
		600 rpm	10	10	10	10
		900 rpm	10	10	10	10
		1200 rpm	10	10	10	10
Day 2	No-Load	300 rpm	10	10	10	10
		600 rpm	10	10	10	10
		900 rpm	10	10	10	10
		1200 rpm	10	10	10	10
	Load	300 rpm	10	10	10	10
		600 rpm	10	10	10	10
		900 rpm	10	10	10	10
		1200 rpm	10	10	10	10

4.1.3 Run-to-Failure Experiments

Run-to-failure (RTF) experiments were conducted in order to study the wear progression of planetary gears. Because we were most interested in the 2^{nd} stage planetary gearbox, a softer gear set was chosen for this stage. This allowed the 2^{nd} stage planetary gears to wear out before obvious wear appeared on the gears of other stages. To track the wear progression of gears, sun, planet and ring gears of the 2^{nd} stage planetary gearbox were all uniquely labeled; moreover, ten teeth of each of these gears were also labeled.

The RTF experiment was operated for 762 hours. Interim, we suspended the experiments many times for inspections. During each suspension, we took pictures of each gear. A microscope was also used to capture details of the wear pattern on each labeled tooth. Figure 4-6 compares tooth profiles for the 1^{st}, 11^{th}, and 19^{th} runs. Figure 4-7 shows changes in the wear pattern on the sun gear of tooth number 10 for these three runs. In the 11^{th} run, we observed a large pit on the top right tooth surface above the pitch line, which might have been caused by adhesive wear. Under the pitch line, few pits were observed (see circled area in Figure 4-7). At the tooth root, significant squeezing wear was observed. In the 19^{th} run, the large pit became smaller due to polishing. Under the pitch line, some old pits were polished away, some became larger, and new pits also appeared. Squeezing wear was still obvious. On the basis of our calculations of tooth profile, we estimated that by the 19^{th} run, 60% of the material had been removed from the sun gear teeth; therefore, we terminated the RTF experiment at this run.

The parameter settings for the RTF experiment are as follows. The motor speed was kept constant at 1200 rpm, the load applied was 20,000 lb·in, and the sampling frequency was 10,000 Hz for runs 1 - 4 and 5,000 Hz for the remaining runs.

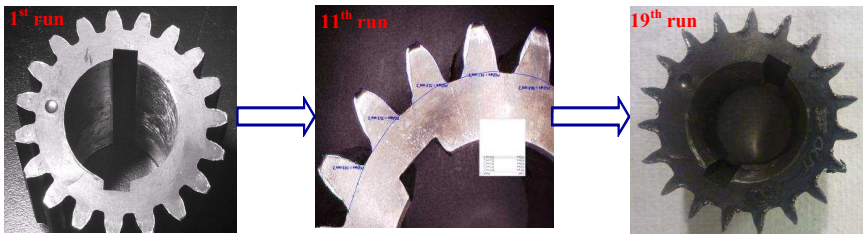

Figure 4-6: Profile change on sun gear teeth in 2nd stage planetary gearbox

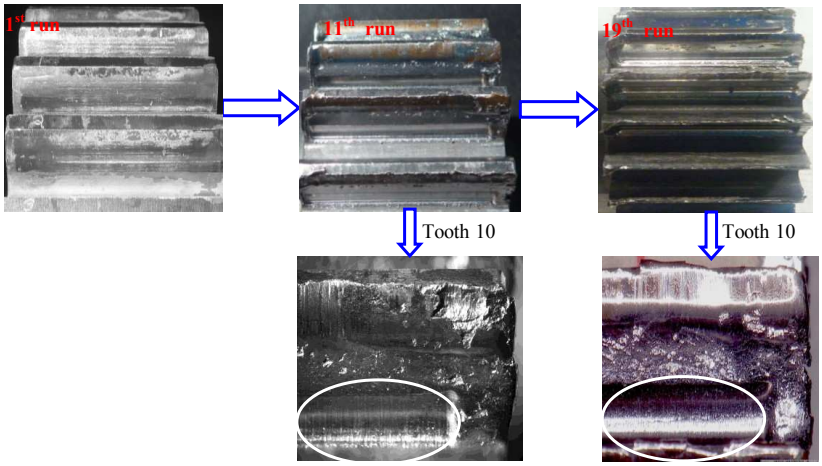

Figure 4-7: Wear pattern change on sun gear teeth in 2nd stage planetary gearbox

4.1.3.1 Data Collection

For the RTF experiment, vibration data were collected every one hour for runs 1-3 and every two hours for the remaining runs. Similar to the manual pitting experiments, every 5-minute time span data were recorded and stored in a data file. The data from each file were further split into 10 segments of equal length. All data segments were then connected in a time series in order to construct a series of degradation data. Table 4-3 shows the duration, the data files available and some experimental parameters for each run.

Table 4-3: Availability of data files for RTF experiment

Run No.	Hours operated	No. of data files	Freq. of collection	Sampling freq. (Hz)
1	8	10	Every 1 hour	10,000
2	8	9	Every 1 hour	10,000
3	16	17	Every 1 hour	10,000
4	32	11	Every 2 hours	10,000
5	32	17	Every 2 hours	5,000
6	64	41	Every 2 hours	5,000
7	32	19	Every 2 hours	5,000
8	32	19	Every 2 hours	5,000
9	48	41	Every 2 hours	5,000
10	48	32	Every 2 hours	5,000
11	54	32	Every 2 hours	5,000
12	48	30	Every 2 hours	5,000
13	48	30	Every 2 hours	5,000
14	48	30	Every 2 hours	5,000
15	48	36	Every 2 hours	5,000
16	56	26	Every 2 hours	5,000
17	58	37	Every 2 hours	5,000
18	51	25	Every 2 hours	5,000
19	31	18	Every 2 hours	5,000

4.2 Slurry Pump Test Rig

Slurry pumps used at Syncrude play the critical role of providing and maintaining a flow of slurries for bitumen separation. Their reliable operation is essential for the running of the whole production process. Slurries contain abrasive and erosive solid particles, which cause severe wear of wetted components in the pumps. The wear of wetted components is a main cause of reduced pump performance and eventual pump failure. Monitoring the wear condition of the wetted components in slurry pumps provides useful information for effective pump operation and maintenance, enhancement of pump availability, and reduction of operation costs [87].

4.2.1 System Description

The slurry pump test rig was designed to run the pump at controlled speeds, flow rates, slurry densities, and inlet pressures using wetted components with controlled levels of damage. Figure 4-8 shows a diagram of the slurry pump test rig.

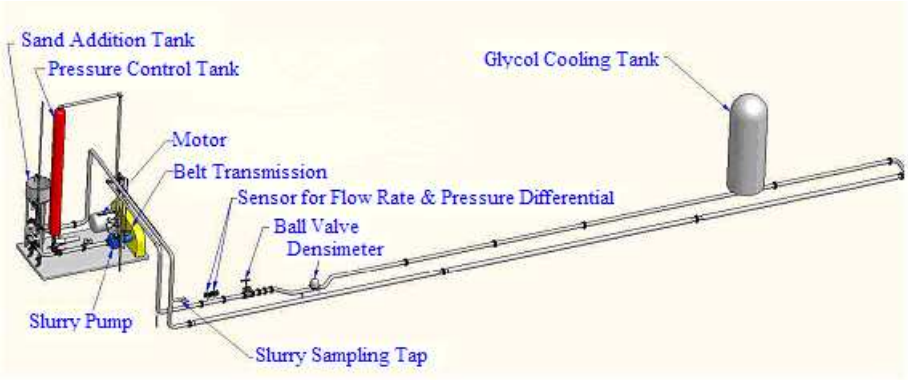

Figure 4-8: Diagram of slurry pump test rig

The test rig contains a Weir/Warman 3/2 CAH slurry pump (40 HP) with impeller C2147 (8.4") which has a radially split casing, a 2-piece liner, a gland packing shaft seal arrangement, a closed impeller design, and a maximum pump casing pressure of 300 psi. The inlet and outlet diameters are 3 inches and 2 inches, respectively. The drive motor rating is 40 HP, 1200 rpm, and 326 T frame. Three tri-axial vibration accelerometers were used, one with high sensitivity and low frequency located on top of the casing (A2) and the other two with low sensitivity and high frequency located at the side of the outlet (A1) and at the top of the pump bearing (A3). There are also pressure gauges located for control purposes at the suction and the discharge sides of the pump. A gate valve is in-line to control the flow rate, and a venturi meter provides flow rate information on the control panel. Figure 4-9 shows local views of the test rig where ADAQ represents acoustic data acquisition, VDAQ represents vibration data acquisition, PIP represents pump inlet pressure, and POP represents pump outlet pressure. More details on this test rig can be found in [87].

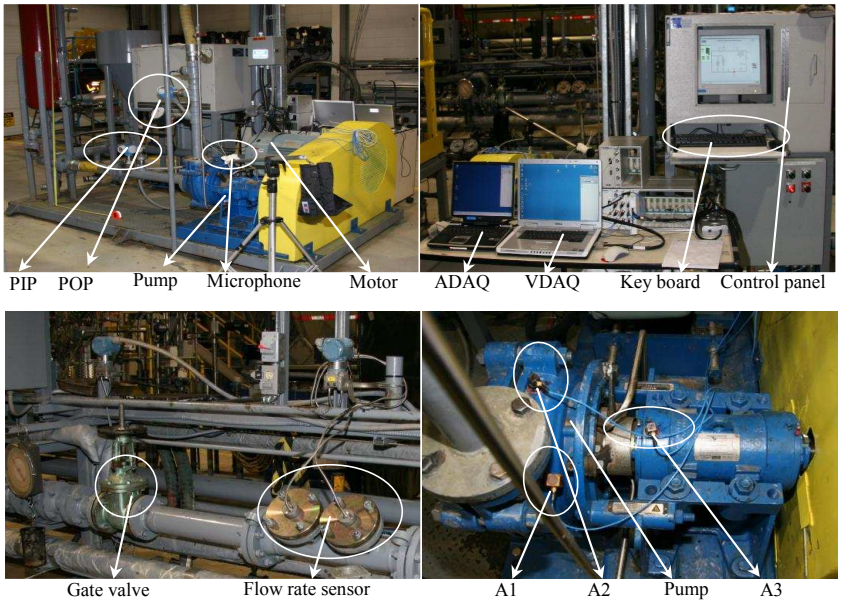

Figure 4-9: Views of slurry pump test rig

4.2.2 Impeller Damage Experiments

Upon examination of the wear patterns of worn impellers taken off from field slurry pumps, four damage modes were considered: impeller hole-through damage, impeller vane trailing edge damage, impeller vane leading edge damage, and impeller expeller vane damage. Figure 4-10 shows views of the studied trailing edge and the leading edge damage from typical industrial pumps. The damage profiles produced on lab impellers mimic the observed wear patterns of worn field impellers.

Figure 4-10: Trailing edge damage (left) and leading edge damage (right)

For each damage mode, four wear levels were used with regard to the impeller: baseline (no wear), slight, moderate, and severe. Detailed information on the profiles of different levels for each of the damage modes machined can be found in [88]. Experiments were conducted at pump speeds of 1800, 2200, and 2600 rpm.

4.2.2.1 Data Collection

The slurry pump experiments were conducted according to the various conditions of medium, damage level and speed which are shown in Table 4-4. Vibration data were collected under these conditions through three accelerometers mounted at different places on the pump casing. For each of these conditions, 5-minute time span data were recorded. The sampling frequency was 9,000 Hz. One can refer to [87] for more details regarding data availability.

Table 4-4: Availability of data files for slurry pump experiment

Medium	Damage mode	Damage degree	Speed (rpm)
Water	Trailing edge damage	Baseline	1400,1600,1800,2200,2600
		Slight	1800,2200,2600
		Moderate	1600,1800,2200,2600
		Severe	1600,1800,2200,2600
	Leading edge damage	Baseline	1400,1600,1800,2200,2600
		Slight	1800,2200,2600
		Moderate	1600,1800,2200,2600
		Severe	1600,1800,2200,2600
Slurry	Trailing edge damage	Baseline	1400,1600,1800,2200,2600
		Slight	1800,2200,2600
		Moderate	1600,1800,2200,2600
		Severe	1600,1800,2200,2600
	Leading edge damage	Baseline	1400,1600,1800,2200,2600
		Slight	1800,2200,2600
		Moderate	1600,1800,2200,2600
		Severe	1600,1800,2200,2600

4.3 Summary

This chapter presents a planetary gearbox test rig and a slurry pump test rig which mimic the field machines employed in the oil sands industry. We collected vibration data from these two test rigs under various experimental conditions. These data are used to examine the performance of our proposed methods for diagnostics and prognostics of engineering systems. Data from the slurry pump and planetary gear-

box are used for damage level classification in Chapter 5. Condition prognostics for the slurry pump and the planetary gearbox are presented in Chapter 6.

Chapter 5

Support-Vector-Machine-Based Diagnostics

As mentioned in Section 1.4, SVM-based classification has been successfully used in diagnostics; however, this is true only for the classification stage. For other stages such as data processing (see Figure 1-3), reported studies are very limited. As mentioned in Section 1.5, two sub-stages of data processing, data cleaning and feature selection, can both be classification-based. This chapter presents a diagnostic algorithm composed of an SVM-based data cleaning [53, 54], an SVM-based feature selection algorithm [89, 90], and the SVM-based classification. This algorithm is basically an off-line process, but the results of this process can be used for online purpose, e.g. the data after using the proposed data cleaning and feature selection can be used to build a SVM classifier for online diagnostics. However, this thesis focuses only on the off-line and the online diagnostic framework needs to be developed in future. The performance of the proposed diagnostic algorithm is studied using various benchmark datasets, and application results are given for the two experimental systems of interest.

5.1 Preliminaries

5.1.1 Terminology

For ease of reading, the terms that are to be used in this chapter are listed. Some of them may be defined in a more specific way than is commonly done.

Feature space: A multi-dimensional space with extracted features as its dimensions, e.g. a feature space has 21 types of feature as its dimensions.

Data point: A multi-dimensional feature space with an output label, e.g. a data point has 21 features and a label of -1 to represent one of two classes.

Dataset: A set containing a certain number of data points.

Training process: A process of determining SVM model parameters where cross-validation methods are usually incorporated.

Testing process: A process of evaluating a built SVM model in terms of a measure for evaluating classification performance (hereafter called classification measure).

Classification accuracy (CA): A classification measure denoted as $N_c/(N_c + N_f) \times 100\%$ where N_c denotes the number of data points that are correctly classified and N_f denotes the number of data points that are falsely classified.

Misclassification rate (MR): A classification measure which is given by: MR=1-CA.

Selection rate (SR): A fraction that specifies the percentage of data points to be chosen for training data in each sampling of random sub-sampling validation.

Classification accuracy and misclassification rate are used to measure the performance of the proposed algorithms. In the field of artificial intelligence, a confusion matrix [91] is also commonly used, in which each column of the matrix represents the instances in a predicted class and each row represents the instances in an actual class. One benefit of the confusion matrix is that it is easy to see if the system is confusing two classes (i.e. mislabeling one as another). According to the above descriptions, the confusion matrix is used to assess the performance of the supervised-learning method, namely, a classifier such as SVM, so that whether the classifier is confused by classes can be revealed. Nevertheless, the proposed algorithms are developed for data cleaning and feature selection, which focus on data processing rather than classification. We will not use the confusion matrix, because whether the classifier is confused by the classes is not our concern. We are more interested in whether the

overall classification results are improved after using the proposed algorithms. This can be sufficiently given by the regular classification accuracy or misclassification rate.

5.1.2 Data Partition

For SVM-based classification, one needs various datasets for purposes of training, validating and testing. In this chapter, we arrange a given dataset according to Figure 5-1. First, we partition the given dataset into an original training dataset and a test dataset; then the original training dataset is further partitioned into a training dataset and a validation dataset. The training dataset is used to train an SVM model and the validation dataset is used to determine appropriate parameters for building an SVM model.

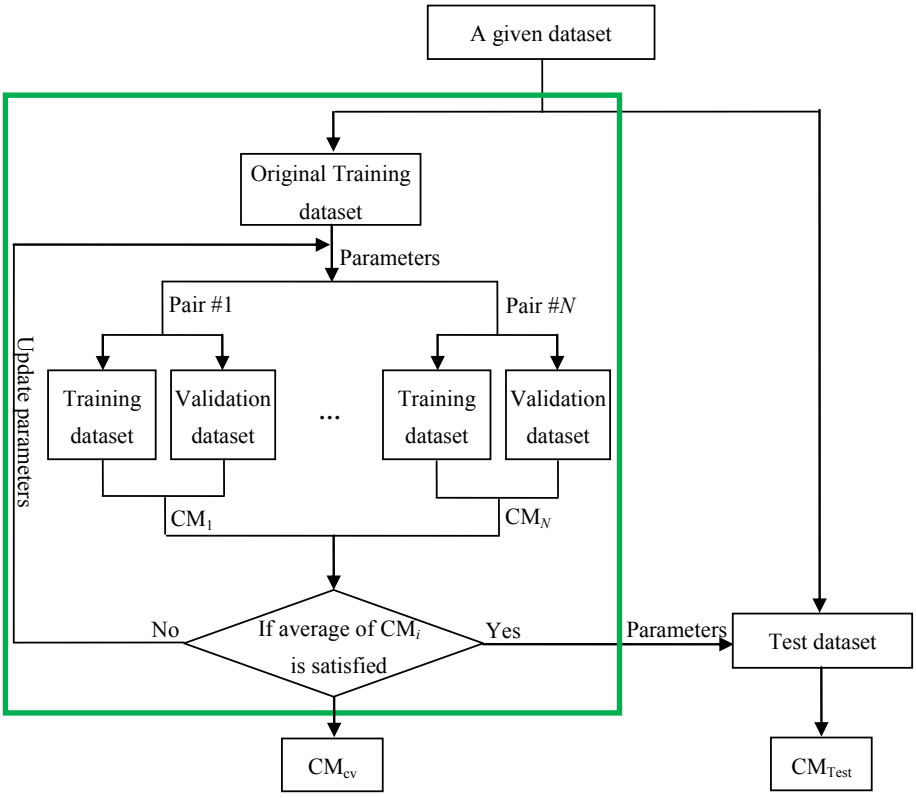

Figure 5-1: Data partition and procedure of cross-validation

Usually, cross-validation methods are incorporated into the training process to help determine model parameters (see the solid square of Figure 5-1). These meth-

ods need multiple pairs of training datasets and validation datasets to calculate the values for the classification measure, denoted as $\mathrm{CM}_i, i = 1, 2, \cdots, N$; the average, denoted as $\mathrm{CM}_{\mathrm{cv}}$, is output for determining the SVM model. For some cases where the number of data points is limited, this average value can also be used directly to represent classification performance. Nevertheless, it is intuitive that some data points may be used as both training and validation data, so the $\mathrm{CM}_{\mathrm{cv}}$ may not reflect the real classification performance. For this reason, the classification measure (computed using the test dataset, $\mathrm{CM}_{\mathrm{Test}}$) is more appropriate, because it has nothing to do with the training process and this allows it to reflect the truth. In this chapter, $\mathrm{CM}_{\mathrm{cv}}$ and $\mathrm{CM}_{\mathrm{Test}}$ are both used but for different cases, depending on the number of data points available.

5.1.3 Cross-Validation

Cross-validation (CV) is a technique for assessing how the results of a statistical analysis will generalize to an independent dataset. It is used mainly for two objectives: model selection and performance estimation, both of which have exactly the same process of implementation. This section introduces three cross-validation methods that are used in the proposed diagnostic algorithm. They provide an average value for classification measure to either determine SVM model parameters or evaluate classification performance. In the following, it is assumed that sufficient data are available, so the CV methods are working with the original training dataset.

K-fold cross validation (KFCV): KFCV splits original training data into K disjoint subsets (folds). In each iteration, $K - 1$ folds are used in the training dataset and the remaining one in the validation dataset. Next, the SVM model is trained and its classification measure is computed. This process is repeated until all folds have been used once for the validation dataset. The average value for the classification measure for K iterations is then computed for output. The output from different runs of KFCV is usually different, because samples are randomly selected for different folds in each run.

Leave-one-out cross validation (LOOCV): LOOCV is actually a degenerate case of KFCV which involves using a single sample of the original training dataset for the validation dataset, and the remaining for the training dataset.

In each iteration, the value for the classification measure is computed. This process is repeated so that each sample in the original training dataset is used once for the validation dataset. LOOCV also returns the average value for the classification measure as output. Unlike the KFCV, the output of LOOCV remains constant over runs because the training and validation datasets are not varied over different runs.

Random sub-sampling validation (RSV): RSV trains the SVM model for a specified number of samplings (iterations) in each of which a fixed number of data points are randomly chosen for the training dataset and the rest for the validation dataset. Like the previous two validation methods, RSV returns as output the average value for the classification measure obtained over the samplings. For a given original training dataset, RSV provides varied average values over different runs, since in each sampling it randomly selects the training and validation datasets.

5.1.4 Benchmark Datasets for Demonstration

The data cleaning and feature selection algorithms proposed in this thesis aim to improve diagnostic results. To demonstrate the performances of the proposed algorithms, the datasets should be widely used by other researchers. Five benchmark datasets are employed for demonstrating the performance of the proposed algorithms in this chapter. They all have been tested and studied by other researchers, so if the results using the proposed algorithms are good, they can be attributable to the proposed algorithms rather than attributable to the uniqueness of the datasets. The first dataset was obtained from MATLAB data library, the others from [92].

1. **The modified Fisher's Iris dataset:** a modification of the original Fisher's Iris dataset provided by MATLAB's data collection. The original dataset is comprised of three classes: "Setosa", "Versilcolor" and "Viginica". There are 4 attributes for a data point: sepal length, sepal width, petal length, and petal width. We modified the original dataset into a binary class dataset which contains data points from only two of the classes, Setosa and Versilcolor.

2. **The colon cancer dataset:** 62 samples of which 22 are from normal tissue and 40 are from tumor tissue [93]. Original expression levels (features) involves

more than 6500 genes which were measured using Affymetrix oligonucleotide arrays. The dataset was filtered down to the 2000 genes with the highest minimal intensity across all samples.

3. **The sonar dataset:** 208 observations on 61 features. The first 60 represent the energy within a particular frequency band, integrated over a certain period of time. The last column provides the class labels. There are two classes: "R" if the object is a rock and "M" if the object is a mine (metal cylinder).

4. **The breast cancer dataset:** 569 samples of which 357 belong to benign, "B", and 212 samples belong to malignant, "M". The dataset includes 32 attributes with their ID number and outcome, benign or malignant. There are 30 real-value features which were computed from a digitized image of a fine needle aspirate (FNA) from a breast mass. They describe the characteristics of the cell nuclei present in the image.

5. **The Parkinson dataset:** a range of biomedical voice measurements from 31 people, 23 of whom have Parkinson's disease (PD). Each attribute is a particular voice measurement, and there are 195 voice recordings from these individuals.

5.2 SVM-Based Diagnostic Algorithm

This section presents the SVM-based diagnostic algorithm that is an integration of an SVM-based data cleaning algorithm, an SVM-based feature selection algorithm, and an SVM-based classification. Figure 5-2 illustrates the relation between each stage and its output. Descriptions of each stage are also given below for better understanding.

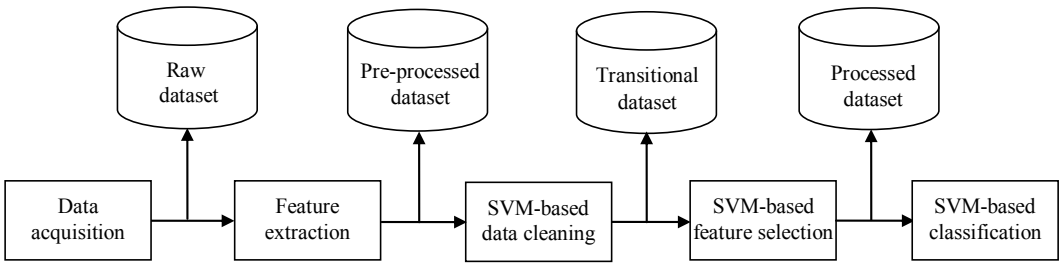

Figure 5-2: Diagram of the proposed diagnostic algorithm

First, data acquisition is conducted to gather condition monitoring data, out of which a raw dataset is built. Next, feature extraction methods are used to process the raw data and return a pre-processed dataset containing a certain number of data points, each of which has an equal dimension of feature space. The pre-processed dataset can be arranged as shown in Table 5-1 where empty cells are assigned feature values; these could be linguistic labels, real-values, boolean numbers, etc. The feature values to be assigned should be able to reflect the characteristics of the corresponding class label. This thesis uses real-value feature only. Next, the pre-processed dataset is input to an SVM-based data cleaning algorithm which provides a transitional dataset, possibly with a reduced number of data points, i.e. M is reduced, because outliers are removed during data cleaning. The transitional dataset is a cleaned pre-processed dataset. It is subsequently imported to an SVM-based feature selection algorithm, the output of which is stored in a processed dataset. The processed dataset may have a reduced number of features for feature space, i.e., L is reduced, because redundant and irrelevant features are removed during feature selection. Such processed data are eventually used in the SVM-based classification in order to identify fault modes or fault levels in the monitored system.

Table 5-1: Structure of pre-processed dataset

Feature # / Data point #	1	2	…	L	Label
1					1
2					-1
…	…	…	…	…	…
M					-1

In the following sections, the proposed SVM-based data cleaning algorithm and the proposed SVM-based feature selection algorithm are separately presented. They serve in the corresponding blocks of Figure 5-2.

5.3 SVM-Based Data Cleaning

Classical methods of outlier identification use statistical indices such as sample mean and covariance matrix to create a measure to determine the distance of a certain data point from others in a defined space; this is done so data points away from the majority of data can be detected and identified as outliers. One drawback of these methods is that one needs to create an appropriate decision criterion for separating good data points from outliers, which is usually difficult. Another drawback is that these methods do not work when outliers lie in between classes, especially when different classes are near each other. Additionally, outliers detected by these methods may be trivial to SVM-based classification because they are far from the boundaries, thus are unlikely to be used as support vectors for determining a separating plane (SP). For these reasons, one needs a method that is different from the classical ones, one that can detect outliers for SVM-based classification. Before introducing the proposed method, the outlier effects on SVM-based classification are illustrated. This will assist in understanding the proposed method.

5.3.1 Outlier Effects on SVM Separating Plane

As presented in Section 3.1, an SP obtained using SVM depends only on support vectors. Theoretically, an outlier located in between classes or even in the opposite class has a high chance of being selected as a support vector, because it is odd to the others. If such outlier data are used for classification, the resulting SP tends to be inappropriate and unacceptable. Figures 5-3 and 5-4 show examples of the influence of an outlier on SP. The symbol "+" represents the data of class one and the symbol solid dot represents the data of class two. A solid white curve represents the SP. The two dotted curves represent the boundaries. The contours in yellow and blue colors represent the areas for the two classes and as data points are close to the SP, the color becomes darker. For both figures, the SP for a dataset without outliers is shown in the left panel, while the one with an outlier is shown in the right

panel. The outlier does not overlap the opposite class in Figure 5-3, but it overlaps the opposite class in Figure 5-4. In the captions, k_f represents kernel function and k_p represents the parameters of kernel function.

In Figure 5-3, the SP moves to the right due to the outlier in class one. Consequently, test data which should belong to class two is classified as class one (left of the SP) as shown in the right panel. In Figure 5-4, not only the SP moves to the left due to the outlier of class one, but a small area of class two data is classified as being class one. A test data of class two in this area is thus incorrectly classified as class one. In accordance with previous remarks, the outlier in the right panel is selected as a support vector (indicated by a circle) for both cases. Based on Figures 5-3 and 5-4, we can conclude that the existence of outliers impairs classification performance, so data cleaning is necessary for removing such undesired data points. The plots in this sub-section were drawn using Steve Gunn's SVM toolbox [94].

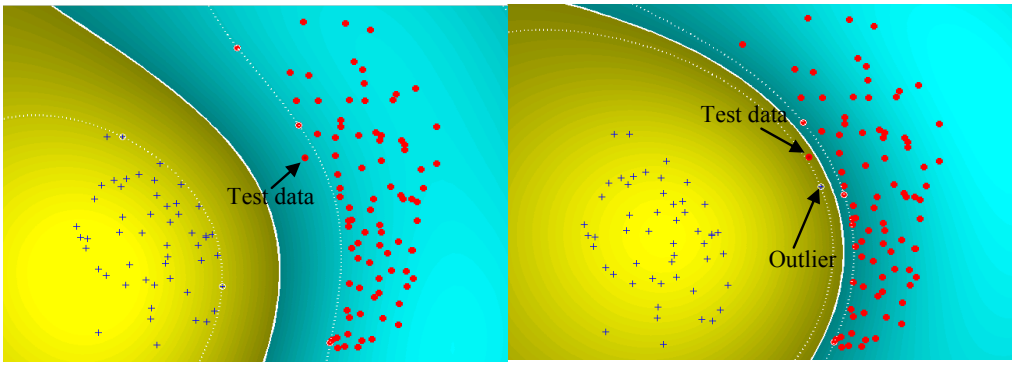

Figure 5-3: Illustration of outlier effects on SP (k_f is Gaussian kernel, $k_p = 1$, and $C = +\infty$)

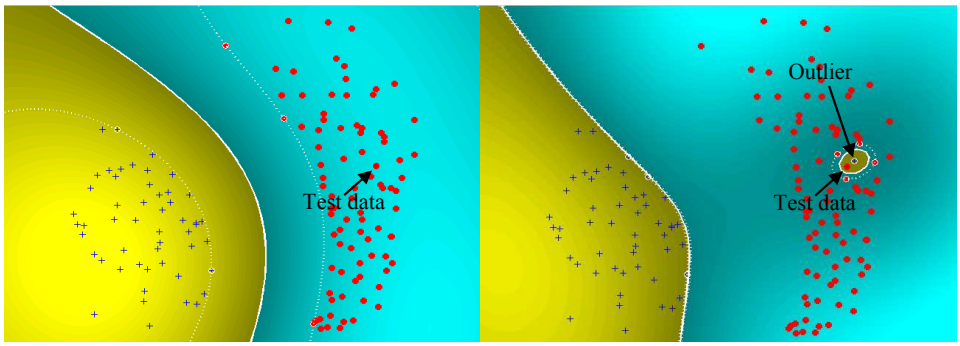

Figure 5-4: Illustration of outlier effects on SP (k_f is Gaussian kernel, $k_p = 1$, and $C = +\infty$)

5.3.2 Outlier Effects on SVM Parameter Selection

The parameters of the SVM model are very critical to classification performance, since improper selection could lead to under-fitting or over-fitting. There are three parameters in the SVM model: kernel function, k_f; kernel parameter, k_p; and parameter C. Parameter C, as shown in Eq. (3.4), is very important for controlling such "fitting" problems. In terms of separable data, if there is no existing outlier, the selection of parameter C is somewhat trivial. One can simply set $C = +\infty$ to obtain a desirable SP. Figure 5-5 shows an example of this, where parameter C is set at 100 for the left panel and positive infinity for the right panel. We see that the SPs are no differences in these two panels. When outliers exist and are overlapped with the opposite class, the selection of parameter C turns out to be complicated. Figure 5-6 gives an example where two outliers exist in two different classes. In the left panel, the classification is acceptable because the outliers are tolerated by the SVM with a parameter C of 100; but over-fitting occurs in the right panel due to the setting of $C = +\infty$, even though it gives no misclassified data. Compared to $C = 100$, $C = +\infty$ creates an over-fitted SP for classification.

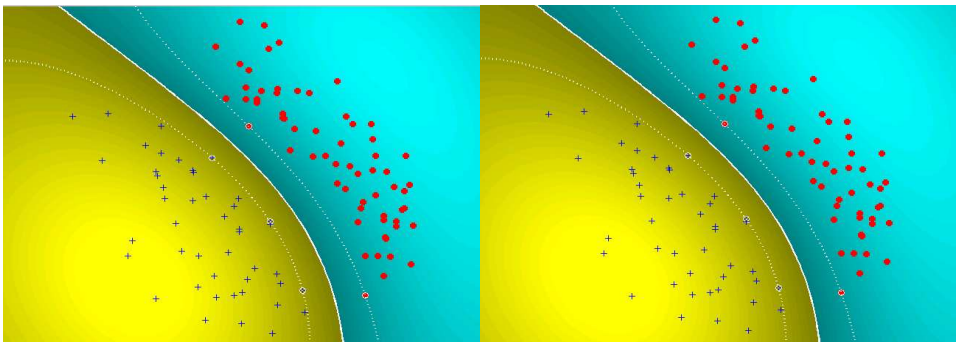

Figure 5-5: Classification without outliers using different settings of parameter C (for both panels, k_f is Gaussian kernel and $k_p = 1$; for parameter C, the left panel is $C = 100$ and the right one is $C = +\infty$)

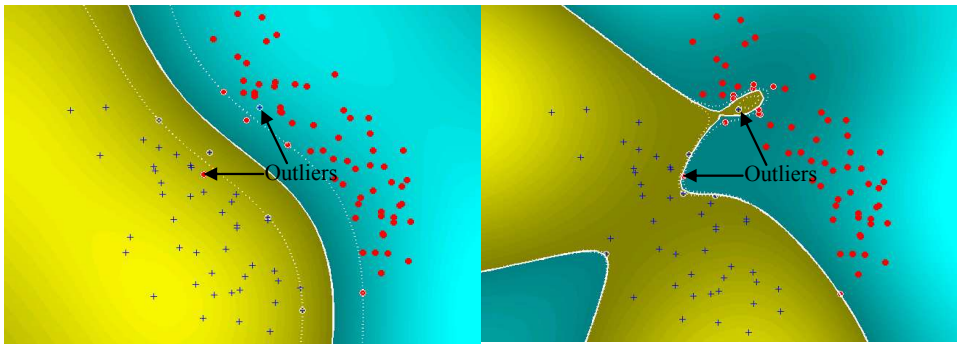

Figure 5-6: Classification with outliers using different settings of parameter C (for both panels, k_f is Gaussian kernel and $k_p = 2$; for parameter C, the left panel is $C = 100$ and the right one is $C = +\infty$)

One way of avoiding such over-fitting is to find an appropriate parameter C as shown in the left panel of Figure 5-6. As a matter of fact, it is difficult for a specific case, since classification is usually implemented in a high dimensional space in which the SP can not be visualized. The other solution is to remove outliers from the dataset; this makes the SP insensitive to the selection of parameter C as shown in Figure 5-5. This is possible if the outliers can be located. Then, the problem turns out to be how to locate the outliers. This is given in the next section.

5.3.3 Principle of Outlier Identification Using SVM

When using SVM for classification, misclassified data points could result from outliers in either the training or test data; however, outliers in training data affect classification results in a different way from those in test data. The former may cause the SP to be falsely determined, leading to misclassifying test data (as shown in Figures 5-3 and 5-4 where one test data point from class two is misclassified). On the other hand, outliers in test data may be misclassified, because they are located in the opposite class according to the SP determined by the training data. If an SP can be reasonably determined, misclassified test data stand a high chance of being outliers. The key point is how to find a reasonable SP, one that is not affected by the outliers in the training data.

As shown in Eq. (3.4), the SVM model introduces a slack variable, ξ_i, tolerant of noise and outliers in the training data; this allows for taking into account more training data points than merely those closest to the boundaries. Thanks to ξ_i, outliers in training data are allowed to be misclassified as long as the parameters of the SVM model have been appropriately selected. For example, two outliers of different classes have been left misclassified in the left panel of Figure 5-6.

With such good characteristics, test data can be used to identify outliers. The following four scenarios may take place during training and testing (see Figure 5-7 for schematics in $\mathbb{R}^2$):

Scenario 1: No outliers are in the training data or the test data;

Scenario 2: Outliers are in the training data only;

Scenario 3: Outliers are in the test data only;

Scenario 4: Outliers are in both the training and the test data.

The corresponding results obtained by using SVM with appropriate parameters are:

Result 1: No outliers are in the training data, so the SP can be determined properly. No outliers are in the test data, so no data points are misclassified;

Result 2: Outliers in the training data are tolerated by the SVM model, so the SP can be determined properly. No outliers are in the test data, so no data points are misclassified;

Result 3: No outliers are in the training data, so the SP can be properly determined. Outliers in the test data are misclassified;

Result 4: Outliers in the training data are tolerated by the SVM model, so the SP can be determined properly. Outliers in the test data are misclassified.

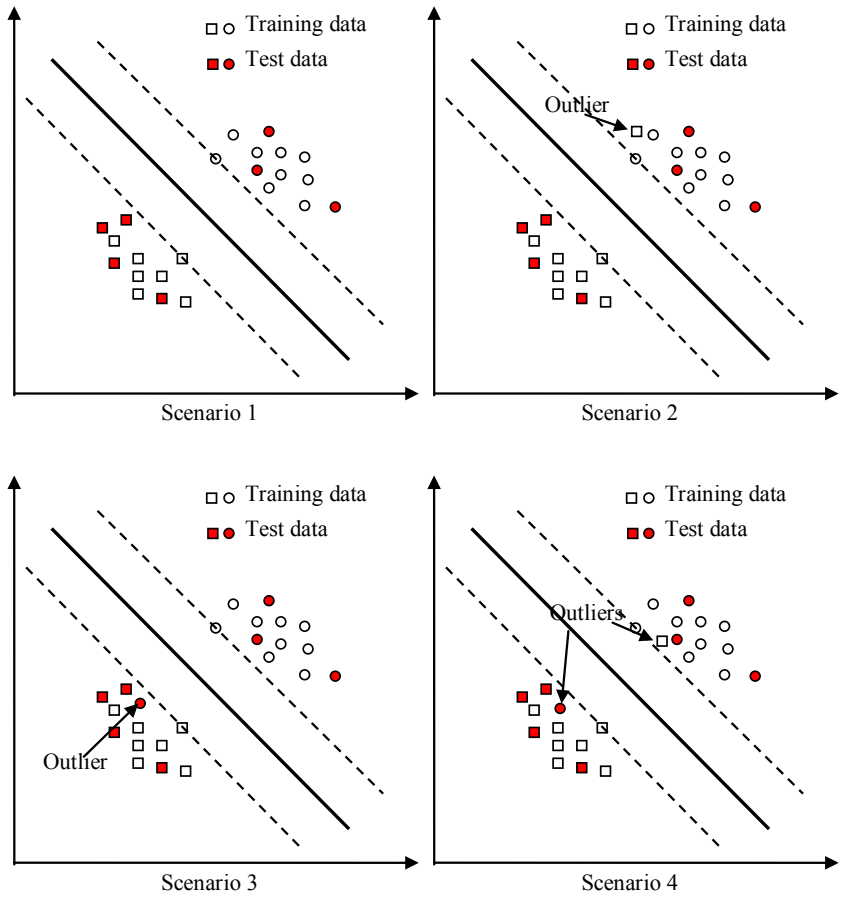

Figure 5-7: Illustration of outlier identification using test data

It can be concluded that if the SP is appropriately determined, outliers in the test data will be misclassified. Hence, if we conduct training and testing for a particular number of runs and label misclassified data points for each run, the final outliers

will be among those that are the most frequently misclassified. The proposed data cleaning algorithm is inspired by this idea.

Nevertheless, one may argue that this idea is quite dependent on setting appropriate SVM parameters, which is not easy. Actually, this is not true and the reasons are given below. Suppose that the settings of SVM parameters are not appropriate. A particular normal data point will be misclassified when real outliers are in the training dataset, leading to an inappropriate SP, and the same data point will be correctly classified when no outliers are in the training dataset. It can be concluded that a normal data point can be correctly classified for at least a certain number of runs. In contrast, an outlier, because it is actually located in the opposite class, will be misclassified for most of the runs whether the SP is appropriate or not. By this way, outliers can be identified.

In the following, a supervised SVM-based data cleaning algorithm is proposed. In the literature, unsupervised SVM-based data cleaning (anomaly detection) approaches are also reported [48]. The major difference between the proposed and their methods is that the proposed algorithm needs training data with known labels. This makes the proposed algorithm able to provide more accurate results than the unsupervised one, but the drawback is that the proposed algorithm is unable to operate when there are no training data with known labels, which are common in practice.

5.3.4 The Proposed Data Cleaning Algorithm

The proposed data cleaning algorithm uses the RSV to conduct the training process many times. The RSV constructs different validation and training datasets over samplings. Given a sufficient number of samplings, it can be assumed that every data point is selected as the validation data or the training data an equal number of times; hence, the chance of a data point being misclassified over samplings can easily be represented by a fraction. Since real outliers should have the greatest chance of being misclassified, the fraction values can thus be used to identify them.

It should be noted that a large fraction may also represent real support vectors which can be used to determine the SP; it is, therefore, inappropriate to determine

outliers by relying solely on the fraction values. To address this problem, the misclassification rate (MR) can be used to determine final outliers, since it is believed that the removal of true outliers lowers the MR value. The value for the fraction is thus computed for each data point, based on which all data points are ranked. Data points corresponding to larger fraction values can thus be treated as candidate outliers. By removing the candidate outliers one by one according to their rank and calculating the corresponding MR values, real outliers can be finally determined.

Figure 5-8 illustrates the procedure of the proposed data cleaning algorithm. As mentioned in Chapter 2, this thesis focuses on a case where data cleaning is implemented after feature extraction, so the proposed algorithm works in a feature space, and the input is the pre-processed dataset. The proposed algorithm is implemented as follows.

First, training and validation data are randomly chosen from the pre-processed dataset based on an SR. Next, data points misclassified in current sampling are recorded and an array is established to store the number of times each data point is misclassified. This process is repeated until a pre-specified number of samplings, N_s, is reached. Then the number of times that the i^{th} data point is misclassified can be obtained, and denoted as $N_f^i, i = 1, 2, \ldots, M$. A fraction value for the i^{th} data point can thus be denoted by $N_f^i/N_s, i = 1, 2, \ldots, M$. Data points are then ranked according to their fraction values from large to small, and the ones with larger fraction values are considered as candidate outliers.

The final outliers are determined based on the impact removing candidate outliers has on MR values. MR values are computed using LOOCV. To begin, the first candidate outlier in the rank is removed from the pre-processed dataset. The removal takes the scheme of "without replacement", which means once a data point is removed it will not be returned to the dataset. The MR values for the pre-processed dataset and the resultant dataset are then computed and compared. If the resultant dataset has a reduced MR value, the second candidate outlier in the rank is subsequently removed, and the MR values for the datasets before and after the removal are compared. The above procedure is repeated until the MR value does not decrease further. As a result, data points whose removal results in a reduction of MR

values are identified as final outliers. These are then permanently removed from the pre-processed dataset, giving a clean dataset, namely a transitional dataset.

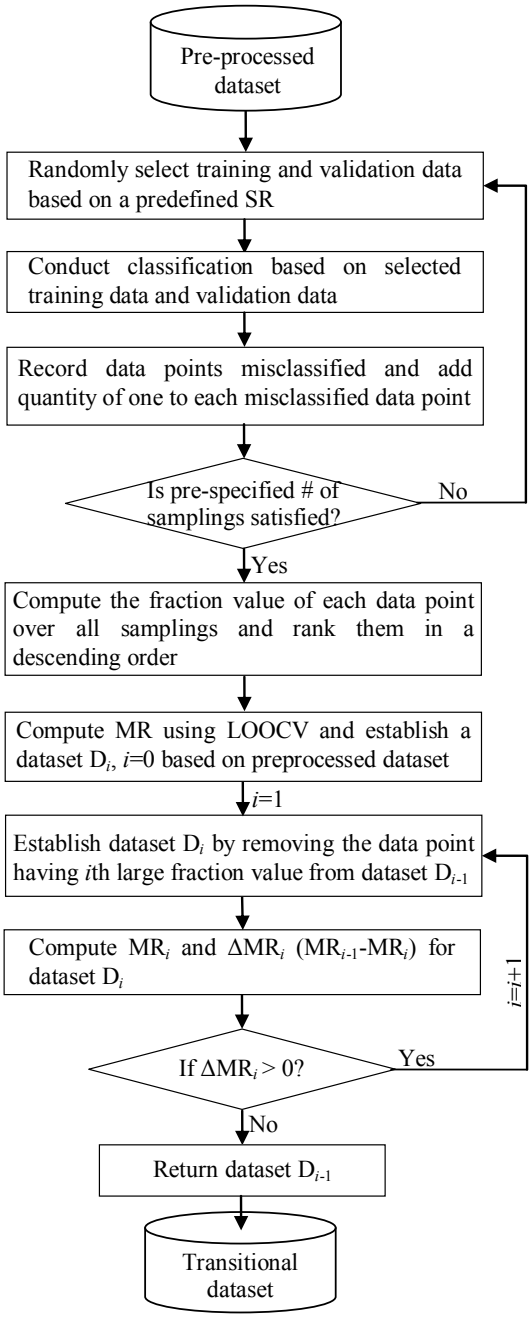

Figure 5-8: Flow chart of the proposed data cleaning algorithm

In the next section, the effectiveness of the proposed algorithm is verified using

various benchmark datasets. The results of these datasets are compared with those reported in the literature.

5.3.5 Evaluation Using Benchmark Datasets

In this section, the performance of the proposed data cleaning algorithm is demonstrated. Three benchmark datasets are used for the demonstrations: the modified Iris dataset, the colon cancer dataset, and the sonar dataset. These datasets are presented in Section 5.1.4. The proposed algorithm is programmed using MATLAB. This chapter uses SVM for classification. Kernel function is an important parameter for SVM. There are a large number of kernel functions available in the literature [82]. However, according to our experience, polynomial and Gaussian kernels are generally capable of addressing most problems. For this reason, this chapter adopts the "trials and errors" tests to select the kernel function from these two. The one to be selected is expected to provide higher classification accuracy.

5.3.5.1 Evaluation Using Modified Iris Dataset

The modified Iris dataset has two classes, and a data point from class Versilcolor is intentionally mislabeled as class Setosa to simulate an outlier. The outlier is data point #70. All data points are plotted in terms of petal length and petal width in Figure 5-9. We see that a Versilcolor data point that should be a blue "+" is mislabeled by a red "*" representing a Setosa data point.

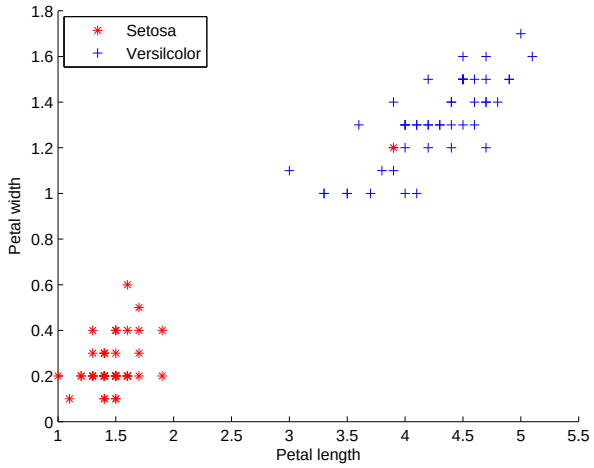

Figure 5-9: Modified Iris dataset

The proposed data cleaning algorithm is applied to the modified Iris dataset and the results are shown in Figure 5-10. The parameters for SVM are as follows: $C = 100$, k_f is polynomial kernel, and $k_p = 1$. It is seen that data point #70 corresponds to a fraction value of one. This means that this data point has been misclassified for every sampling; it can thus be identified as the only candidate outlier. All other data points are correctly classified and correspond to a fraction value of zero. The MR value computed is 1% for the modified Iris dataset. When data point #70 is removed, the MR value becomes 0; hence, data point #70 is identified as a final outlier.

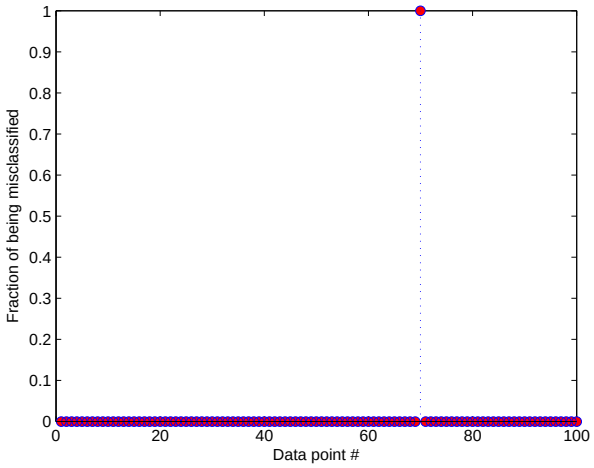

Figure 5-10: Fractions of misclassified data points for the modified Iris dataset

5.3.5.2 Evaluation Using Colon Cancer Dataset

The colon cancer dataset is a widely used benchmark dataset in outlier identification. The proposed data cleaning algorithm suggests that the outliers are samples from N36, N34, T36, T33 T30, T2, and N8 where "N" represents the normal samples and "T" represents the tumor samples. The proposed data cleaning algorithm is compared with three reported methods. Shieh's method [46] used principal component analysis (PCA) and robust estimation of Mahalanobis distances (MD). Alon's method [93] used a two-way clustering algorithm based on deterministic annealing. Li's method [95] used a genetic algorithm with a K-nearest neighbor (KNN) classifier. The outliers detected for the colon caner dataset were explicitly given in their papers. This allows for a direct comparison with the outliers detected by our

algorithm.

The MR values for the clean datasets resulting from the four methods were computed using LOOCV. SVM is used as the classifier and the parameters for SVM are $C = 100$, k_f is Gaussian kernel, and $k_p = 1$. The results are listed in Table 5-2. The MR values shown in the last column are based on the clean datasets, that is, those without outliers. The MR value for the original dataset is 17.74%. We find that every algorithm provides a reduced MR value and the proposed algorithm gives the smallest. Because the CPU time consumptions were not reported for the three existing methods and we did not program their methods, the comparison of CPU time consumption is not conducted.

Table 5-2: Performance comparisons using colon cancer dataset

Algorithms	Outliers identified	MR
Shieh's method	T20, T30, T33, T36, T37, N8, N12, and N34	3.7%
Alon's method	T30,T33,T36, N8, N34, and N36	3.6%
Li's method	T2, T30, Y33, T36, T37, N8, N34, and N36	1.8%
The proposed algorithm	N36, N34, T36, T33 T30, T2, and N8	0%

The proposed data cleaning algorithm is an SVM-based algorithm, so it is likely to give the smallest MR value when SVM is adopted for classification; such good performance should be consistent when other classifiers are used. Because of this concern, the outliers identified by the proposed algorithm were also tested using four other classifiers, including k-nearest neighbor (KNN) classifier, diagonal linear discriminant analysis (DLDA), classification trees (CT), and feed-forward back-propagation neural network (FBNN). We used MATLAB-provided commands with default settings for the first three classifiers and 10 hidden layers and 1 output layer for FBNN.

Table 5-3 shows the results of our comparisons. The MR values in the "Original" column are for the original dataset. The MR values in the "Random" column are for the dataset in which six random samples are removed from the original dataset. For each classifier, the six random samples were drawn uniformly from the original dataset. Such datasets were included to demonstrate that simply removing samples does not reduce MR values. The MR values in the "clean" column are for

the datasets without the outliers identified by the proposed algorithm. Comparing results show that the clean dataset achieves the best performance for every classifier. This shows that the proposed algorithm gives consistent good performance for classification irrespective of the classifiers used for this dataset. It also shows that the removal of random samples is unable to consistently improve classification performance, and in some cases (such as KNN, CT and FBNN) it has even impaired classification performance.

Table 5-3: Results of classification for colon cancer dataset using different classifiers

	Original	Random	Clean
SVM	17.74%	14.55%	0%
KNN	22.58%	25.45%	14.55%
DLDA	37.09%	35.71%	18.18%
CT	17.74%	21.05%	5.45%
FBNN	17.74%	26.78%	5.45%

5.3.5.3 Evaluation Using Sonar Dataset

Li's method [96] combines estimating the overall probability density and sequential ranking of the data according to observed changes in performance on validation datasets. The validation dataset is discussed in Section 5.1.2. This method is used for the sonar dataset and six outlier samples: #3, #101, #23, #172, #134, and #131 are found. The proposed algorithm also detected six outlier samples: #45, #18, #74, #151, #36, and #8. Table 5-4 compares the results. The parameters for SVM classification are $C = 100$, k_f is Gaussian kernel, and $k_p = 1$. We find that the outliers identified by the two algorithms are entirely different. Because the sonar dataset used in this experiment was obtained from the UCI repository as that used in [96], this tremendous difference is strange.

Table 5-4: Performance comparisons using Sonar dataset

Algorithms	Outliers identified	MR
Li's method	#3, #101, #23, #172, #134, and #131	25.25%
The proposed algorithm	#45, #18, #74, #151, #36, and #8	16.83%

We tested the outliers identified by the proposed algorithm using different classi-

fiers; the results are shown in Table 5-5. It is seen that the cleaned dataset performs best for every classifier and removing random samples does not improve classification performance consistently.

Table 5-5: Results of classification for Sonar dataset using different classifiers

	Original	Random	Clean
SVM	25.48%	25.24%	16.83%
KNN	12.50%	13.36%	11.88%
CT	33.65%	27.72%	18.81%
DLDA	30.77%	30.20%	29.21%
FBNN	20.19%	21.37%	18.81%

5.3.6 Short Summary

Based on the test results of benchmark datasets, the performance of the proposed data cleaning algorithm is demonstrated. Because the three benchmark datasets adopted are also used by many other researchers for testing their methods, the effectiveness of the proposed algorithm using the same benchmark datasets is convincing. This effectiveness is due to that the proposed algorithm is classification-focused. Compared to the classical data cleaning methods which focus on identifying the outliers distantly away from the majority, the proposed algorithm focuses on identifying the outliers in-between opposite classes. This enables the outliers impairing classification results to be identified and removed rather than those that do not impair classification results. As the classification-focused outlier identification is not well studied, we conclude the proposed algorithm has important contributions to this scope.

5.4 SVM-Based Feature Selection

As mentioned in Section 3.1, SVM seeks an SP that provides the largest margin for two classes of data points; this is because the larger the margin, the better the separation of data points. With this unique property, margin value can be used to quantify the importance of a certain feature subset to classification performance. This observation is the theoretical support for the SVM-based feature selection algorithm that is presented below.

5.4.1 Feature Evaluation using SVM Measure

According to Eq. (3.3), the margin is proportional to the reciprocal of the norm of the weight vector $\|\boldsymbol{w}\|$; hence, the norm value can be used to assess feature sets in terms of classification performance. In the following, the feature set of the transitional dataset is referred to as the original feature space. When a feature is removed from the original feature space, the resultant norm value may or may not be changed. This property allows us to evaluate the feature. The explanations are given below using an example.

Suppose that the original feature space has $L, (L > 2)$ features. We remove features C and D from the original feature space, obtaining the resultant $\|\boldsymbol{w}_{\mathrm{C}}\|$ and $\|\boldsymbol{w}_{\mathrm{D}}\|$, respectively. The $\|\boldsymbol{w}_0\|$ is obtained using the original feature space. Now, we consider four scenarios of which true observations are listed in Table 5-6. For the sake of brevity, we explain in detail only one of the four scenarios, but the other scenarios can be explained following the same rationale.

Table 5-6: Feature importance comparisons

Scenarios \ Conditions		Observations		
		$\lvert\delta_{\mathrm{C}}\rvert > \lvert\delta_{\mathrm{D}}\rvert$	$\lvert\delta_{\mathrm{C}}\rvert = \lvert\delta_{\mathrm{D}}\rvert$	$\lvert\delta_{\mathrm{C}}\rvert < \lvert\delta_{\mathrm{D}}\rvert$
$\|\boldsymbol{w}_{\mathrm{C}}\|=\|\boldsymbol{w}_0\|+\delta_{\mathrm{C}}$ & $\|\boldsymbol{w}_{\mathrm{D}}\|=\|\boldsymbol{w}_0\|+\delta_{\mathrm{D}}$	$\delta_{\mathrm{C}}>0$, $\delta_{\mathrm{D}}>0$	$I_{\mathrm{C}}>I_{\mathrm{D}}$	$I_{\mathrm{C}}=I_{\mathrm{D}}$	$I_{\mathrm{C}}<I_{\mathrm{D}}$
	$\delta_{\mathrm{C}}>0$, $\delta_{\mathrm{D}}<0$	$I_{\mathrm{C}}>I_{\mathrm{D}}$	$I_{\mathrm{C}}>I_{\mathrm{D}}$	$I_{\mathrm{C}}>I_{\mathrm{D}}$
	$\delta_{\mathrm{C}}<0$, $\delta_{\mathrm{D}}>0$	$I_{\mathrm{C}}<I_{\mathrm{D}}$	$I_{\mathrm{C}}<I_{\mathrm{D}}$	$I_{\mathrm{C}}<I_{\mathrm{D}}$
	$\delta_{\mathrm{C}}<0$, $\delta_{\mathrm{D}}<0$	$I_{\mathrm{C}}<I_{\mathrm{D}}$	$I_{\mathrm{C}}=I_{\mathrm{D}}$	$I_{\mathrm{C}}>I_{\mathrm{D}}$

Scenario: $\|\boldsymbol{w}_{\mathrm{C}}\| = \|\boldsymbol{w}_0\| + \delta_{\mathrm{C}}$ and $\|\boldsymbol{w}_{\mathrm{D}}\| = \|\boldsymbol{w}_0\| + \delta_{\mathrm{D}}$, $(\delta_{\mathrm{C}} > 0$ and $\delta_{\mathrm{D}} < 0)$

Observation: The equality $\|\boldsymbol{w}_{\mathrm{C}}\| = \|\boldsymbol{w}_0\| + \delta_{\mathrm{C}}, (\delta_{\mathrm{C}} > 0)$ indicates that removing feature C increases the $\|\boldsymbol{w}\|$ value (reduces the margin); this means that the removal of feature C impairs the classification; therefore, feature C is important to the classification. Inversely, the equality $\|\boldsymbol{w}_{\mathrm{D}}\| = \|\boldsymbol{w}_0\| + \delta_{\mathrm{D}}, (\delta_{\mathrm{D}} < 0)$ indicates that removing feature D reduces the $\|\boldsymbol{w}\|$ value (increases the margin); because its removal enhances the classification performance, feature D is harmful to the classification. We can thus conclude that, for this scenario, no matter what the absolute values are for δ_{C} and δ_{D}, feature C is more important than feature D (see the 2^{nd} scenario of Table 5-6); this is denoted

by, $I_{\mathrm{C}} > I_{\mathrm{D}}$, where I represents the importance of a particular feature to the classification. Any criterion satisfying Table 5-6 can be used to evaluate feature importance for the purpose of classification.

5.4.2 The Proposed Feature Selection Algorithm

5.4.2.1 Model for Binary Classification

As mentioned in Chapter 2, a feature selection method includes feature ranking and feature selecting. In accordance with Table 5-6, we propose a measure for feature ranking; this is given as:

$$\delta_i = \|\boldsymbol{w}_i\| - \|\boldsymbol{w}_0\|, i = 1, 2, \ldots, L. \tag{5.1}$$

where L represents the total number of features.

Based on Eq. (5.1), we can easily obtain the observation results given in Table 5-6, since the following causal relations always hold: if $\delta_{\mathrm{C}} > \delta_{\mathrm{D}}$, $I_{\mathrm{C}} > I_{\mathrm{D}}$; if $\delta_{\mathrm{C}} = \delta_{\mathrm{D}}$, $I_{\mathrm{C}} = I_{\mathrm{D}}$; and if $\delta_{\mathrm{C}} < \delta_{\mathrm{D}}$, $I_{\mathrm{C}} < I_{\mathrm{D}}$. The features can be ranked for feature selection in accordance with the δ values.

Forward selection (FS) and backward selection (BS) are two commonly-used feature selection schemes. FS adds useful features to an empty feature set. BS eliminates useless features from the original feature set. As discussed in Section 2.2.3, FS relies more on a perfect rank, whereas, BS relies less on rank quality. Though BS may leave some useless features in the final feature subset, most useful features are reserved. BS is more robust than FS in terms of obtaining good classification results, because it is usually difficult to ensure a perfect rank.

We adopt the BS scheme in the proposed algorithm. The CA value obtained from KFCV is employed to determine whether a particular feature should be removed. Unlike regular BS where feature ranking is conducted only once, the proposed algorithm re-ranks the features when no more features can be removed from the current feature space. As a result, regular BS is recursively implemented until removing the first top-ranked feature decreases the CA. This recursive backward selection (RBS) allows irrelevant and redundant features remaining in the reduced feature space to have multiple chances of being removed. There is, however, a side effect: it may

take more computational time. Figure 5-11 shows the flow chart of the proposed algorithm where the measures of U and R represent CA and $\|\boldsymbol{w}\|$, respectively. The flow chart is depicted in detail in the next section.

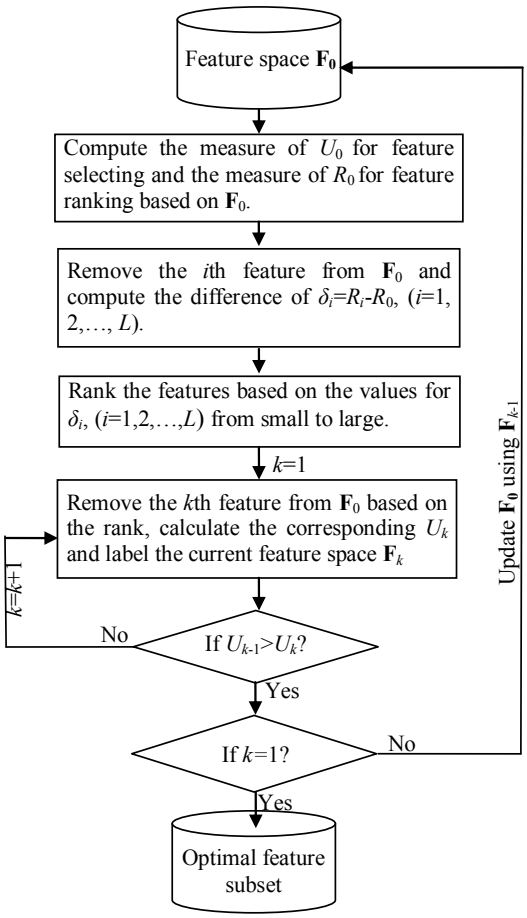

Figure 5-11: Flow chart of the proposed feature selection algorithm

5.4.2.2 Model for Multi-class Classification

The proposed algorithm for binary classification is also extended for multi-class classification problems. The One-against-all (OAA) approach is adopted for multi-class SVM classification. Figure 5-11 is still applicable to this case where the measure of U is still defined as the CA, but the measure of R is re-defined to adapt to the multi-class cases given below.

First, the U_0 (CA) is calculated based on the feature space, $\mathbf{F}_0$. $\mathbf{F}_0$ is the original feature space and is used in the first iteration. The value for $R_{0,j}(\|\boldsymbol{w}_{0,j}\|), j =$

$1, 2, \ldots, N$ is also calculated; it is returned by the j^{th} SVM model when all features in the $\mathbf{F}_0$ have been used. Next, the impact of removing the i^{th} $(i = 1, 2, \ldots, L)$ feature on the j^{th} $(j = 1, 2, \ldots, N)$ SVM model is calculated, represented by $\delta_{i,j} = \|\boldsymbol{w}_{i,j}\| - \|\boldsymbol{w}_{0,j}\|, i = 1, 2, \ldots, L$ and $j = 1, 2, \ldots, N$. This yields an impact matrix:

$$\begin{bmatrix} \delta_{1,1} & \delta_{2,1} & \cdots & \delta_{L,1} \\ \delta_{1,2} & \delta_{2,2} & \cdots & \delta_{L,2} \\ \vdots & \vdots & \vdots & \vdots \\ \delta_{1,N} & \delta_{2,N} & \cdots & \delta_{L,N} \end{bmatrix} \tag{5.2}$$

where the impact of removing each feature on a certain binary SVM model is given in the row, and the impact of removing a certain feature on each binary SVM model is given in the column. Since there is often no prior knowledge regarding which SVM model should be preferred in classifying a particular data point, we use the equal weighting technique to evaluate the overall impact of removing a certain feature. For the i^{th} feature, this is given by $\Delta_i = \frac{1}{N}\Sigma_{j=1}^{N}\delta_{i,j}, i = 1, 2, \ldots, L$ and $j = 1, 2, \ldots, N$. The features are then ranked according to these Δ values. The feature corresponding to the smallest Δ value has the top rank and the one corresponding to the largest Δ value has the lowest rank. Next, the features are removed one at a time, starting from the top-ranked feature, until removing a feature decreases the CA. The feature space, $\mathbf{F}_0$, is then updated by eliminating the removed features. The above procedure is repeated until the CA is decreased when removing the first top-ranked feature (the least useful feature). The optimal feature subset is then obtained using the original feature space without all the removed features.

5.4.2.3 Additional Details

At the feature ranking step of the proposed algorithm, several features may have the same δ value for binary class cases and the same Δ value for multi-class cases. For binary class cases, these features are arbitrarily ranked. For multi-class cases, the following strategy is adopted. Suppose that features H and G have the same Δ value. We count the number so that $\delta_{H,j} > \delta_{G,j}, j = 1, 2, \ldots, N$ is satisfied and denote it as n_{com}. If $n_{\text{com}} > N/2$, feature G will be placed ahead of feature H in the rank. If $n_{\text{com}} < N/2$, feature H will be placed ahead of feature G. Otherwise, the two features are ranked arbitrarily. When more than two features have the same

Δ value, this strategy is applied to any pair of features until every such feature is ranked.

5.4.3 Evaluation Using Benchmark Datasets

This section demonstrates the performance of the proposed feature selection algorithm. Three benchmark datasets are used: the sonar dataset, the breast cancer dataset, and the Parkinson dataset. These three datasets have been introduced in Section subsec:5BMDUD, and pre-arranged identically as described in Section 5.1.2. Feature selection is conducted using training and validation datasets. Once the optimal feature subset is determined, the whole dataset is updated using the selected features. Then the original training dataset is used to train the SVM model, and the test dataset is used to evaluate its performance.

The proposed algorithm is compared with Gualdrón's SVM-based algorithm [28] in terms of feature ranking and selecting. The Gualdrón's algorithm used the following feature ranking measure:

$$\delta_i = | \, \|\boldsymbol{w}_0\| - \|\boldsymbol{w}_i\| \, |, i = 1, 2, \ldots, L, \tag{5.3}$$

where all the parameters are defined in the same way as those in Eq. (5.1). FS was used as its feature selecting scheme. To fully demonstrate our algorithm's performance, we examine both feature ranking and feature selecting schemes. This is done by comparing the following three algorithms, the Gualdrón's algorithm, the proposed feature ranking + Gualdrón's feature selecting (PFR+GFS) and the proposed feature ranking + the proposed feature selecting (PFR+PFS). Comparing the Gualdrón's algorithm and PFR+GFS reveals how performs the proposed feature ranking versus Gualdrón's feature ranking. Comparing PFR+GFS and PFR+PFS reveals how performs the proposed RBS versus FS. In addition, SVM using all features is used as a baseline in relation to which the capabilities of the three methods of improving classification performance can be assessed. We conducted 30 independent trials for each of the three benchmark datasets.

5.4.3.1 Evaluation Using Sonar Dataset

Table 5-7 shows the results using the sonar dataset. With regard to the CA, we found that Gualdrón's algorithm provided a value even smaller than the baseline.

In contrast, PFR+GFS increased the baseline value by about 3%. This value was further increased by 9%, when PFR+PFS was used. We observed similar results from the column of "Number of features" where the mean values were all rounded off. Though with the identical value of three, PFR+GFS provided a CA value about 6% greater than that of Gualdrón's method. This shows that the proposed feature ranking is better able to detect useful features.

Table 5-7: Results of classification for sonar dataset

Methods		CA (%)			Number of features		CPU time (s)
		Mean	Std.	PBP	Mean	Std.	Mean
SVM using all features		77.55	2.26	0	60	0	-
Gualdrón's method		74.63	7.46	40	3	1.36	8.4729
The proposed algorithm	PFR+GFS	80.44	9.15	53.3	3	1.38	8.4328
	PFR+PFS	89.58	4.95	100	20	7.76	20.819

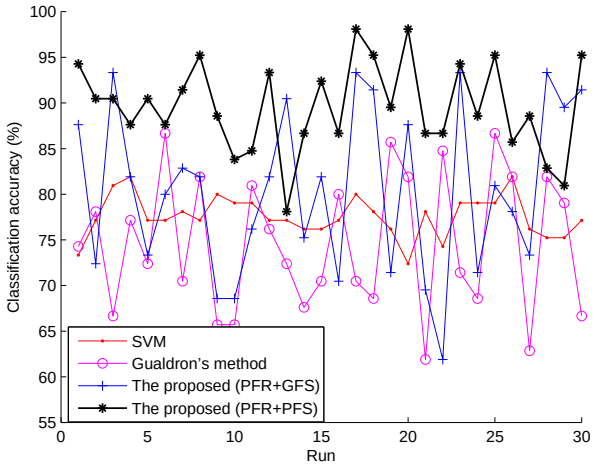

Figure 5-12: Results of CA for sonar dataset over 30 trials

An index called the percentage of better performance (PBP) was also used to evaluate the robustness of these feature selection methods. The PBP is the ratio between the number of trials where the CA value is improved by a feature selection method and the total number of trials. It is seen that PFR+PFS improved the CA for every trial (PBP=100%); in contrast, Gualdrón's method had a PBP of 40% and PFR+GFS had a PBP of 53.33%, indicating a much lower robustness. Figure 5-12 shows the CA value for each method for all 30 trials. It is seen that PFR+PFS provided almost the largest CA values for every single trial.

In terms of CPU time, Gualdrón's method and PFR+GFS use comparable amount of time. It indicates that the PFR and Gualdrón's feature ranking basically consume the same amount of CPU time, as they both adopt the forward selecting scheme. In contrast, the PFR+PFS consumes twice the CPU time of the other two methods. This is a drawback of the RBS — the greater the gain in CA, the more expensive the computations. For the given sonar dataset, we can see that double CPU time consumptions contribute to 10% increase of CA.

5.4.3.2 Evaluation Using Breast Cancer Dataset

Table 5-8 shows the results using the breast cancer dataset. Basically, the three methods performed much as they did for the sonar dataset. Figure 5-13 shows the performance of each method over 30 trials, giving four curves that were basically separated from each other. The PFR+PFS curve is above all the others, revealing the superior robustness of this method. The curve for Gualdrón's method is at the bottom, lying even below the baseline curve.

Table 5-8: Results of classification for breast cancer dataset

Methods		CA (%)			Number of features		CPU time (s)
		Mean	Std.	PBP	Mean	Std.	Mean
SVM using all features		88.96	1.79	0	30	0	-
Gualdrón's method		79.72	6.98	3.33	2	0.76	182.96
The proposed algorithm	PFR+GFS	93.85	2.64	96.67	4	1.43	178.15
	PFR+PFS	99.32	0.43	100	22	4.34	364.36

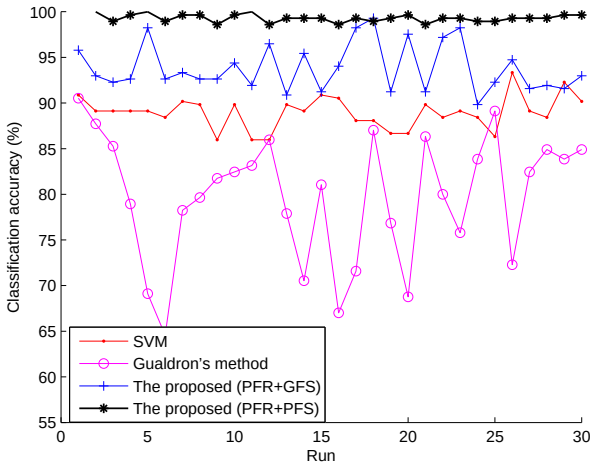

Figure 5-13: Results of CA for the breast cancer dataset over 30 trials

Unlike the sonar dataset, PFR+GFS provided a CA of 93.85% and a good PBP of 96.67%. Moreover, it selected fewer features in the final feature subset and used the least CPU time. If the CA requirement is not exceptionally high, PFR+GFS may be a good choice for the breast cancer dataset indicates, because it balances CA, the number of features selected, and computational time.

As mentioned, PFR+GFS uses the least amount of CPU time, 178.15s. Gualdrón's method uses a slightly higher CPU time, 182.96s. However, for the CA value, PFR+GFS is 14% higher than that of Gualdrón's method. Hence, it is no doubt that PFR+GFS is much better than Gualdrón's method in terms of CA. PFR+PFS uses the CPU time of 364.36s which is approximately twice the other two methods. This is similar to the sonar dataset. For the given breast cancer dataset, we can see that double CPU time consumptions contribute to merely 5% increase of CA.

5.4.3.3 Evaluation Using Parkinson Dataset

Table 5-9 lists the results using the Parkinson dataset. It is seen that Gualdrón's algorithm and PFR+GFS provided almost the same results; their curves almost merged together as Figure 5-14 shows. Studying the two feature ranking schemes, we found that the δ values for Eq. (5.1) are positive for most features, with the exception of five that have nearly zero δ values. Taking the absolute values of these δs, we obtained similar results as found using Eq. (5.3), so this explains why

Gualdrón's algorithm and PFR+GFS gave the same results. These obtained δ values satisfy the scenarios of both the first row ($\delta_C > 0$ and $\delta_D > 0$) and conditions of the first column ($|\delta_C| > |\delta_D|$) and the second one ($|\delta_C| = |\delta_D|$) in Table 5-6. From this it can be concluded that most of the features in the Parkinson dataset are useful and the rest are redundant.

Table 5-9: Results of classification for Parkinson dataset

Methods		CA (%)			Number of features		CPU time (s)
		Mean	Std.	PBP	Mean	Std.	Mean
SVM using all features		94.43	2.05	0	23	0	-
Gualdrón's method		84.33	7.42	13.33	3	1.66	3.25
The proposed algorithm	PFR+GFS	84.47	7.44	13.33	3	1.92	3.27
	PFR+PFS	96.15	1.65	76.67	11	4.65	7.12

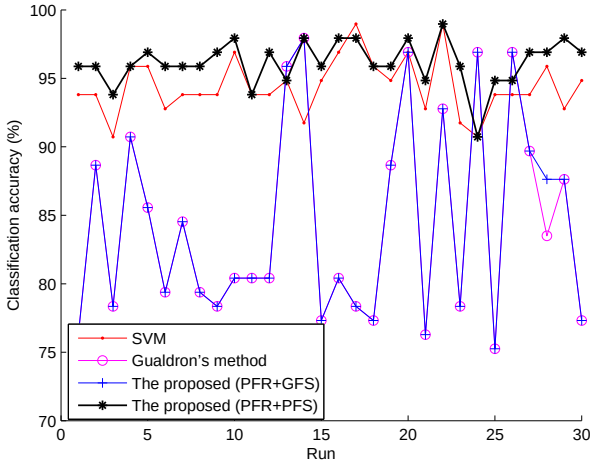

Figure 5-14: Results of CA for Parkinson dataset over 30 trials

This conclusion accords well with the results of the row labeled "SVM using all features" where a high CA (94.43%) is obtained without using feature selection. Although a reasonable rank is obtained, FS fails to select all possible useful features; whereas, the proposed RBS selects a subset containing 11 features and achieves an even higher CA (96.15%).

In terms of CPU time, it is generally consistent with the other two datasets. The CPU time for Gualdrón's method and PFR+GFS are almost the same. PFR+PFS

uses twice the CPU time of the other two methods. For the given Parkinson dataset, we can see that double CPU time consumptions contribute to 11% increase of CA.

It is interesting that for all three datasets PFR+PFS always uses twice the CPU time of the other two methods. Basically, the CPU time consumption is dependent on the number of iteration conducted for removing useless features, the greater the number, the larger the CPU time. The proposed method uses RBS that sacrifices CPU times for high CA values. For this reason, when using the proposed method, CPU time should be considered, for sometimes it may go to a high number that one does not desire.

5.4.4 Short Summary

Based on the test results of benchmark datasets, the performance of the proposed feature selection algorithm is demonstrated. Because the three benchmark datasets adopted are also used by many other researchers for testing their methods, the effectiveness of the proposed algorithm using the same benchmark datasets is convincing. This effectiveness stems from the deep understanding of the SVM theorem for classification. Compared to a reported SVM-based feature selection method, the proposed algorithm addresses the drawbacks of the reported method in both feature ranking and feature selecting aspects. Therefore, we conclude that the proposed algorithm enhances the theorem of SVM-based feature selection and provides a good choice for feature selection.

5.5 Applications

This section presents the application results using the proposed algorithms. First, the proposed data cleaning algorithm is used in classifying the damage level in the given slurry pump system; next, the proposed feature selection algorithm is used in classifying the level of pitting damage in the given planetary gearbox system. The proposed diagnostic algorithm is applied to the slurry pump system to show how well integrating data cleaning and feature selection worked.

5.5.1 Data Cleaning for Slurry Pump System

5.5.1.1 Feature Extraction and Database Establishment

As mentioned in Section 4.2.2, four damage levels involving impellers were studied: baseline, slight, moderate, and severe. We labeled them classes 1, 2, 3, and 4, respectively. Experiments were conducted for each of the four damage modes. For each damage mode and each damage level, pump speed was varied using the control panel. At each specified pump speed, several process parameters were recorded. Table 5-10 lists those (F1 – F7) to be used as features for classifying damage levels. The number of data points belonging to classes 1, 2, 3, and 4 are 11, 97, 88, and 88, respectively. These numbers were determined by the number of experiments conducted.

Table 5-10: List of extracted features for impeller damage level classification

Features	Description	Features	Description
F1	Pump rpm	F12	Peak to Peak - A1
F2	Motor Speed (%)	F13	Peak to Peak - A2
F3	Horse Power	F14	Kurtosis - A1
F4	Flow Rate (maximum)	F15	Kurtosis - A2
F5	Flow Rate (minimum)	F16	1X - A1
F6	Outlet Pressure (maximum)	F17	1X - A2
F7	Outlet Pressure (minimum)	F18	2X - A1
F8	RMS - A1	F19	2X - A2
F9	RMS - A2	F20	5X - A1
F10	Peak - A1	F21	5X - A2
F11	Peak - A2		

Vibration signals with a time span of 5 minutes were collected for each combination of damage mode, pump medium, pump speed, and damage level within specified ranges. We used the signals of two channels, A1 and A2 (see Figure 4-9 for locations), to extract features following the suggestions in [87].

Possible useful features were also extracted using statistical and signal processing techniques. First, we selected four time-domain features which are commonly used for monitoring the condition of rotating machinery.

Root mean square (RMS): a time domain feature which measures the power content in a vibration signature. It is a good idea to track the overall noise standard deviation in order to be able to detect a major out-of-balance in

rotating systems. The RMS value is given by [97]:

$$\text{RMS} = \sqrt{\frac{1}{T}\Sigma_{t=1}^{T} x^2(t)} \tag{5.4}$$

where $x(t), t = 1, 2, \ldots, T$ represents a data series, and T represents the length of the data series.

Peak: the maximum amplitude of the signal regardless of the sign [98] .

Peak-to-peak: the difference between the maximum and minimum of the signal [98].

Kurtosis: the fourth moment of the distribution; it measures the relative peakedness or flatness of the distribution and is used as an indicator of major peaks in a set of data. Kurtosis is given by [97]:

$$\text{Kurtosis} = \frac{\Sigma_{t=1}^{T}[x(t) - \mu]^4}{T(\sigma^2)^2} \tag{5.5}$$

where $x(t), t = 1, 2, \ldots, T$ represents a data series, T represents the length of the data series, μ represents the mean of the data series given by $\mu = \frac{1}{T}\Sigma_{t=1}^{T} x(t)$, and σ^2 represents the variance of the data series given by $\sigma^2 = \frac{1}{T-1}\Sigma_{t=1}^{T}(x(t) - \mu)^2$.

Spectral analysis in the frequency domain is also employed to extract features. Three harmonics of shaft rotating speed are considered:

First harmonic (1X): presents in the vibration spectrum due to several faults mainly caused by imbalance, even for a good pump [99].

Second harmonic (2X): usually appears in the vibration spectrum of rotating machinery. It is the two times of the 1X in the frequency spectrum.

Vane passage frequency (5X): is equal to the number of blades times the shaft speed. It is often a dominant component of pump's vibration and is of special interest for pump diagnostics [100].

The above seven features are extracted from each of the two channels of vibration signals which provide 14 more features for feature space. They are F8 – F21 as listed in Table 5-10. The number of data points belonging to classes 1, 2, 3, and 4 are

8, 26, 32, and 32, respectively. Note that these numbers are different from those of the process data, that is why we retained only the data points having values for all 21 features on which the pre-processed dataset was established with a dimension of 94×21. As a result, the number of data points the pre-processed dataset has is 4, 26, 32, and 32 for classes 1, 2, 3, and 4, respectively. It is seen that class 1 has only 4 data points which is a small number. First, 4 data points are not enough for building SVM model because the information that the data can convey is very limited. Second, from the viewpoint of the classification accuracy, suppose that 2 data points are used for training, and then 2 data points are used for testing. If one testing data point is misclassified, the classification accuracy is 50%. This is not likely to reflect the capability of the metohd. For this reason, we consider only classes 2, 3, and 4. As a result, the pre-processed dataset ultimately has a dimension of 90×21.

5.5.1.2 Results

The slurry pump data are analyzed using OAA-based multi-class SVM classification. At first, genetic algorithms (GA) combined with 8FCV were used to determine the parameters of the multi-class SVM model based on the pre-processed dataset. Once obtained, the optimal parameters were used in the SVM model to establish relations between features and damage levels. One may argue that because of using GA, the computational time of the proposed data cleaning will be significantly increased. Some discussions about this argument is given below. The GA used here is not for the proposed data cleaning algorithm but for determining the model parameters of SVM, the classifier. This is a common step for data-driven-approach-based classification [11] such as determining the optimal number of layers, neurons, training epochs and so forth for artificial intelligence networks. For this reason, the computational time should be attributed to the classification phase, namely SVM. SVM is used to do classification based on which the classification accuracy can be calculated and the data cleaning algorithms can be evaluated. This phase is needed by most data cleaning algorithms not specifically for the proposed data cleaning, so it is not reasonable to say the proposed data cleaning algorithm needs more computational time due to adopting GA.

The parameters obtained for SVM were: $C = 19000$, k_f is the polynomial kernel, and $k_p = 1$. The parameters used for the proposed algorithm were: $N_s = 5000$ and SR $= 4/5$. The SR value selected resulted in $72(4/5 \times 90)$ data points being selected as training data and $18(1/5 \times 90)$ data points being used as validation data for each sampling.

Figure 5-15 shows the fraction values for misclassified data points. It is found that many data points have non-zero fraction values of which data point #65, with a value of one, is the largest. The fraction values for data points #83, #66, and #90 are also large (all more than 90%) which are regarded as candidate outliers. The MR value is 14.4% for the pre-processed dataset. After removing data point #65, the MR value became 11.23%, but when data point #83 which has the second largest fraction value was sequentially removed, the MR value increased to 11.36%. As a result, only data point #65 was identified as a final outlier.

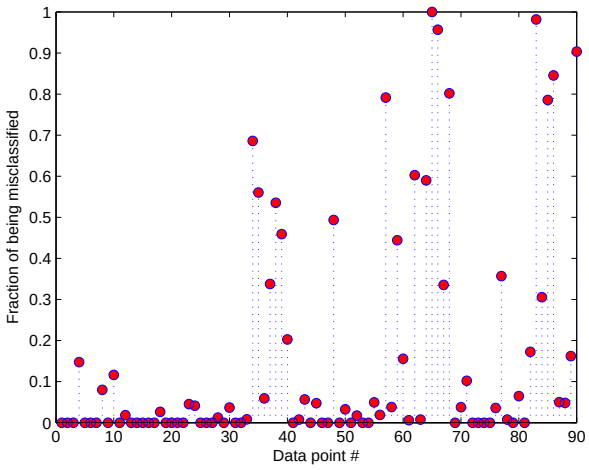

Figure 5-15: Fractions of misclassified data points for slurry pump dataset

5.5.1.3 Discussions

The data point #65 was determined to be an outlier. We investigated the whole preprocessed dataset and found that the data point #65 had a pump speed (the first feature) of 1400 rpm. This value appeared only once in the preprocessed dataset and other speeds were 1600, 1800, 2200, and 2600 rpm. Because the rpm values were recorded manually, we suspected that human errors occurred here. Investigations

on the values of other features for the outlier did not show any anomaly.

To visualize the outlier from other data, we used PCA to extract three principal components for the use of plotting. Figure 5-16 shows the 3D-graph plotted based on the preprocessed dataset. Basically, we could not see a clear classification from the plot. This may be due to the fact that the three principal components do not contain all useful information needed for classifying the data. However, the plot still provides us some clues in distinguishing the outlier. In Figure 5-16, the identified outlier (data point #65) is belonging to the slight wear class represented by "+". We differentiated the outlier using "∗". It can be found that the "∗" is away from the majority of "+" but is close to the severe wear class represented by "•". Figure 5-16 provides a visual evidence of identifying the data point #65 as an outlier.

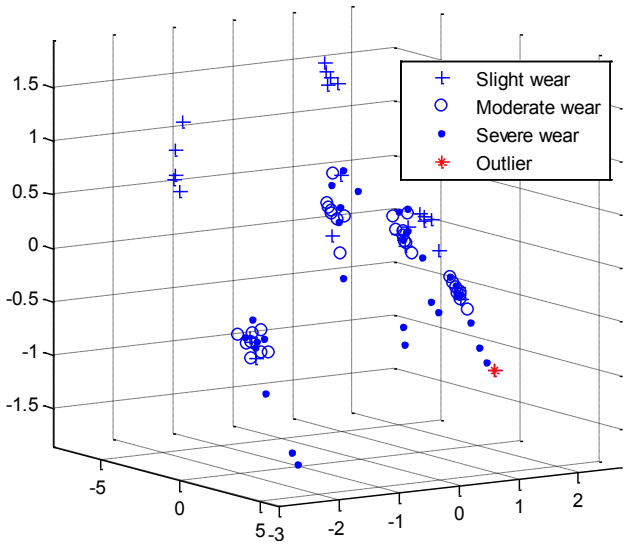

Figure 5-16: 3D plot of pump data distribution

5.5.2 Feature Selection for Planetary Gearbox System

5.5.2.1 Feature Extraction and Database Establishment

The features to be extracted for classifying the level of manual pitting damage in the planetary gearbox include those features reported for fault detection and fault mode classification of fixed shaft gearboxes [33, 87, 97, 101]. Due to the unique behavior of planetary gearboxes, their sidebands are different from those of fixed shaft gearboxes

[102]. Hence, our modification of the features which require information regarding the sidebands of the planetary gearbox.

Figure 5-17 shows the preprocessing of vibration signals for feature extraction. We define the regular mesh components (RMCs) as the fundamental shaft frequency, its second harmonic, gear mesh frequency (GMF), its harmonics and its 1st order sidebands. It is reported in [102] that unlike the sidebands of the fixed shaft gearbox, those of the planetary gearbox appear at the integer multiples of planet passing frequency (the number of planets multiplied by carrier frequency), with the largest sideband being found at the frequency closest to the GMF. Figure 5-18 shows the spectrum of the manual pitting data for our 2^{nd} stage planetary gearbox. All gears are in a normal condition, the motor speed is 1200 rpm, and the load applied is 10,000 lb·in. The horizontal axis depicts the carrier order which is the ratio of frequency value to carrier frequency. The red-dashed vertical line corresponds to the GMF. We see that the amplitude of GMF is not sizable, and that the maximum amplitude appears at the carrier order of 80 which is twenty multiples of four, the number of planet gears. Figure 5-18 indicates that the sidebands of a planetary gearbox may show up even if there is no damage on its gears.

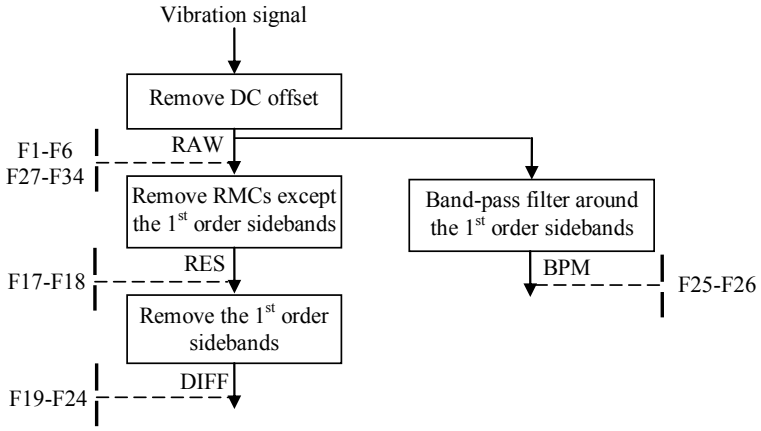

Figure 5-17: Processing flow for feature extraction

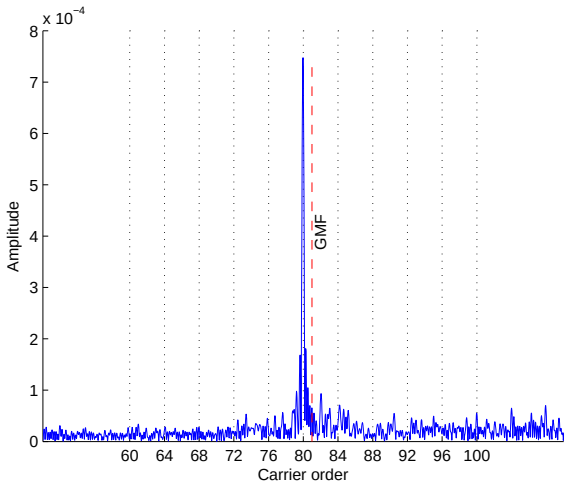

Figure 5-18: An example of planetary gearbox sidebands

Table 5-11: List of extracted features for manual pitting damage level classification

	Features	Definition		Features	Definition
F1	Maximal value	$\max(x(t))$	F2	Minimal value	$\min(x(t))$
F3	Average absolute value	$\frac{1}{T}\sum_{t=1}^{T}\lvert x(t)\rvert$	F4	Peak to peak	F1-F2
F5	Variance	$\frac{1}{T}\sum_{t=1}^{T}\left(x(t)-\bar{x}\right)^2$	F6	Standard deviation	$\sqrt{\frac{1}{T}\sum_{t=1}^{T}\left(x(t)-\bar{x}\right)^2}$
F7	Skewness	$\frac{\frac{1}{T}\sum_{t=1}^{T}\left(x(t)-\bar{x}\right)^3}{\mathrm{F6}^3}$	F8	Kurtosis	$\frac{\frac{1}{T}\sum_{t=1}^{T}\left(x(t)-\bar{x}\right)^4}{\mathrm{F5}^2}$
F9	Root mean square (RMS)	$\sqrt{\frac{1}{T}\sum_{t=1}^{T}x(t)^2}$	F10	Crest factor	$\frac{\mathrm{F1}}{\mathrm{F9}}$
F11	Clearance factor	$\frac{\mathrm{F1}}{\frac{1}{T}\sum_{t=1}^{T}x(t)^2}$	F12	Impulse factor	$\frac{\mathrm{F1}}{\mathrm{F3}}$
F13	Shape factor	$\frac{\mathrm{F9}}{\mathrm{F3}}$	F14	Delta RMS	$\sqrt{\frac{1}{T}\sum_{t=1}^{T}x_m(t)^2}-\sqrt{\frac{1}{T}\sum_{t=1}^{T}x_{m-1}(t)^2}$
F15	Energy ratio	$\frac{\sqrt{\frac{1}{T}\sum_{t=1}^{T}\left(d(t)-\bar{d}\right)^2}}{\sqrt{\frac{1}{T}\sum_{t=1}^{T}\left(x(t)-\bar{x}\right)^2}}$	F16	Energy operator	$\frac{\frac{1}{T}\sum_{t=1}^{T}\left(\Delta x(t)-\Delta\bar{x}\right)^4}{\left(\frac{1}{T}\sum_{t=1}^{T}\left(\Delta x(t)-\Delta\bar{x}\right)^2\right)^2}$
F17	NA4	$\frac{\frac{1}{T}\sum_{t=1}^{T}\left(r(t)-\bar{r}\right)^4}{\left(\frac{1}{M_n}\sum_{m=1}^{M_n}\left(\frac{1}{T}\sum_{t=1}^{T}\left(r_m(t)-\bar{r}_m\right)^2\right)\right)^2}$	F18	NA4*	$\frac{\frac{1}{T}\sum_{t=1}^{T}\left(r(t)-\bar{r}\right)^4}{\left(\frac{1}{M_h}\sum_{m=1}^{M_h}\frac{1}{T}\sum_{t=1}^{T}\left(r_m(t)-\bar{r}_m\right)^2\right)^2}$
F19	FM4	$\frac{\frac{1}{T}\sum_{t=1}^{T}\left(d(t)-\bar{d}\right)^4}{\left(\frac{1}{T}\sum_{t=1}^{T}\left(d(t)-\bar{d}\right)^2\right)^2}$	F20	FM4*	$\frac{\frac{1}{T}\sum_{t=1}^{T}\left(d(t)-\bar{d}\right)^4}{\left(\frac{1}{M_h}\sum_{m=1}^{M_h}\left(\frac{1}{T}\sum_{t=1}^{T}\left(d_m(t)-\bar{d}_m\right)^2\right)\right)^2}$
F21	M6A	$\frac{\frac{1}{T}\sum_{t=1}^{T}\left(d(t)-\bar{d}\right)^6}{\left(\frac{1}{T}\sum_{t=1}^{T}\left(d(t)-\bar{d}\right)^2\right)^3}$	F22	M6A*	$\frac{\frac{1}{T}\sum_{t=1}^{T}\left(d(t)-\bar{d}\right)^6}{\left(\frac{1}{M_h}\sum_{m=1}^{M_h}\left(\frac{1}{T}\sum_{t=1}^{T}\left(d_m(t)-\bar{d}_m\right)^2\right)\right)^3}$
F23	M8A	$\frac{\frac{1}{T}\sum_{t=1}^{T}\left(d(t)-\bar{d}\right)^8}{\left(\frac{1}{T}\sum_{t=1}^{T}\left(d(t)-\bar{d}\right)^2\right)^4}$	F24	M8A*	$\frac{\frac{1}{T}\sum_{t=1}^{T}\left(d(t)-\bar{d}\right)^8}{\left(\frac{1}{M_h}\sum_{m=1}^{M_h}\left(\frac{1}{T}\sum_{t=1}^{T}\left(d_m(t)-\bar{d}_m\right)^2\right)\right)^4}$
F25	NB4	$\frac{\frac{1}{T}\sum_{t=1}^{T}\left(e(t)-\bar{e}\right)^4}{\left(\frac{1}{M_n}\sum_{m=1}^{M_n}\left(\frac{1}{T}\sum_{t=1}^{T}\left(e_m(t)-\bar{e}_m\right)^2\right)\right)^2}$	F26	NB4*	$\frac{\frac{1}{T}\sum_{t=1}^{T}\left(e(t)-\bar{e}\right)^4}{\left(\frac{1}{M_h}\sum_{m=1}^{M_h}\left(\frac{1}{T}\sum_{t=1}^{T}\left(e_m(t)-\bar{e}_m\right)^2\right)\right)^2}$
F27	Mean Frequency	$\frac{1}{K}\sum_{k=1}^{K}X(k)$	F28	Frequency Center	$\frac{\sum_{k=1}^{K}\left(f(k)\cdot X(k)\right)}{\sum_{k=1}^{K}X(k)}$
F29	Root Mean Square Frequency	$\sqrt{\frac{\sum_{k=1}^{K}\left(f(k)^2\cdot X(k)\right)}{\sum_{k=1}^{K}X(k)}}$	F30	Standard Deviation Frequency	$\sqrt{\frac{\sum_{k=1}^{K}\left(\left(f(k)-\mathrm{F28}\right)^2\cdot X(k)\right)}{\sum_{k=1}^{K}X(k)}}$
F31	Largest sideband amplitude	$\max(X(k^*))$	F32	FM0	$\frac{\mathrm{F4}}{\sum X(k^*)}$
F33	Sideband index	$\frac{\sum X(k^*)}{2}$	F34	Sideband level factor	$\frac{\sum X(k^*)}{\mathrm{F6}}$

We define the 1st order sidebands for planetary gearboxes as the lower and the upper sidebands closest to the GMF. Four types of signal are used to calculate features: raw signals (RAW), residual signals (RES), difference signals (DIFF) and band-pass mesh signals (BPM). RAW denotes the vibration signal subtracted by its mean, DIFF denotes the RAW excluding the RMCs, RES is similar to DIFF but has the 1st order sidebands included, and BPM denotes the band-pass mesh signal which is obtained using a band-pass filter filtering around the 1st order sidebands. The RAW, DIFF, RES, and BPM are represented by $x(t), d(t), r(t)$, and $b(t), t = 1, 2, \ldots, T$, respectively, where T is the number of data points in the data series.

Table 5-11 lists 34 extracted features including 26 time domain features and 8 frequency domain features. The time domain features include 16 commonly-used ones (F1 – F16) for fault diagnosis of generic systems and 10 advanced ones (F17 – F26) exclusively proposed for gear fault detection. Frequency domain features are calculated based on sidebands including 4 (F27 – F30) proposed for gear fault detection and 4 (F31 – F34) exclusively developed for planetary gearboxes. Details on these features can be found in [33, 34, 97, 101–103].

The notations in Table 5-11 are defined as follows:

1. $\Delta x(t)$ is obtained piecewise. For the non-endpoints, it is obtained by the squared $x(t)$ subtracted by the product of the data points of $x(t-1)$ and $x(t+1)$. For the endpoints, the data point of $x(t)$ is looped around.

2. $X(k), k = 1, 2, \ldots, K$ represents the k^{th} measurement of the frequency spectrum of $x(t)$. $f(k)$ represents the frequency value for the k^{th} spectrum line.

3. $x_m(t), r_m(t)$ and $d_m(t)$ represent the RAW, RES and DIFF of the m^{th} time record, respectively. The bar notation represents the mean, e.g. $\bar{x}$ represents the mean of $x(t)$. M_n represents the total number of time records up to the present. M_h represents the total number of time records corresponding to healthy conditions of the gearbox. See [87] for details regarding estimating the variance for a gearbox in good condition.

4. $e(t)$ represents the envelope of the current time record expressed as $e(t) = |b(t) + j \cdot H(b(t))|$ where $H(b(t))$ represents the Hilbert transform of $b(t)$ and $e_m(t)$ represents the envelope of the m^{th} time record.

5. k^* represents the index of the 1^{st} order sidebands.

In Table 5-11, features F17 - F26 are exclusively reported for gearbox diagnosis. Their definitions are specially described in the following. For illustration purpose, they may not be introduced following the order of their labels.

FM4 is defined to be the kurtosis of the DIFF over each time record where a complete data series collected (called a run ensemble) is divided into M time records each including T data points [104]. According to [105], FM4 reacted well to the damage on one or two isolated teeth, but lost its sensitivity significantly as the damage spreads to other teeth. Thus, FM4 is good for detecting initial faults, but not suitable for measuring the progression of faults.

FM4* is equal to the ratio between the fourth moment of a DIFF time record and the squared value of the average variance of the "healthy" DIFF time records [34]. FM4* is designed for the purpose of monitoring the progression of a fault instead detecting of the initial fault. At the point of using FM4*, we assume that the fault has already appeared. Among the M time records in the run ensemble, the first M_h time records correspond to the gearbox condition of no fault yet. The remaining records would then contain information on the growth of the fault.

M6A is defined to be the ratio between the sixth moment of a DIFF time record and the third power of the variance of the DIFF time records. It is called M6A in [106] and M6 in [107]. As we know, kurtosis uses the fourth moment in the numerator and the second power in the denominator, while M6A uses the sixth moment in the numerator and the third power in the denominator. The underlying theory of M6A is the same as that of FM4. However, M6A is expected to be more sensitive to peaks in the DIFF due to the use of the sixth moment in the numerator [108].

M6A* is based on the M6A with the exception that it differentiates the "healthy" records from the "faulty" records [106]. It uses the average variance of the "healthy" DIFF time records in the denominator. Based on its definition, M6A* is expected to improve the performance of M6A in tracking progression

of gear faults. In addition, M6A* is more sensitive than FM4* due to the use of the sixth moment [106].

M8A is exactly the same as M6A except that the sixth moment of a DIFF time record becomes the eighth moment and the third power of the variance of the DIFF time record becomes the fourth power. M8A is expected to be more sensitive to peaks than M6A by definition [108] .

M8A* is exactly the same as M6A* except that the sixth moment of a DIFF time record becomes the eighth moment and the third power of the variance of the DIFF time record becomes the fourth power. M8A* is expected to be more sensitive to the fault progression than M6A* by definition [108].

NA4 is defined to be the ratio between the fourth moment of an RES time record and the squared value of the average variance of the RES time records in the run ensemble where the run ensemble is divided into M time records each with T data points. NA4 is designed to overcome the shortcoming of FM4 which is less sensitive to the progression of faults. NA4 can not only detect the onset of damage, as FM4 does, but also continue to react as the damage spreads and increases in magnitude [105].

NA4* is defined as the kurtosis of the run ensemble normalized by the squared average variance of the RES time records collected when the gearbox is in the "healthy" condition [109]. This is similar to FM4*. Ref. [109] reports that NA4* is robust for indicating not only initial damage but also progressive damage.

NB4 is equal to the ratio between the fourth moment of a BPM time record and the squared value of the average variance of the BPM time records within the run ensemble of the envelope signal [104]. The theory behind NB4 is that the damage on gear teeth will cause transient load fluctuation that is different from that caused by healthy teeth, and this can be seen from the envelope of the signal.

NB4* aims to improve the performance of NB4 in tracking damage progression. The concept of "healthy" time records is used here [106]. Ref.[104] reports

that NB4 and NB4* present trends similar to those of NA4 and NA4*, with a more robust indication to the severity of damage.

In accordance with Section 4.1.2.3, we built a dataset with 80 data points, each of which has 136 (34 features $\times$ 4 accelerometers) input features and an output damage level. The features were numbered in the following way: #1 – #34 are from LS1, #35 – #68 are from LS2, #69 – #102 are from HS1, and #103 – #136 are from HS2. The features for each accelerometer are in the same order as shown in Table 5-11, e.g. features #1, #35, #69, and #103 all correspond to maximal value (F1). There are eight conditions of load and speed, hence there are eight classification problems to be addressed.

5.5.2.2 Results

Since four damage levels are involved, each of the eight problems is a 4-class classification problem. We used the multi-class model of the proposed algorithm for feature selection. The parameters used for classification are: k_f is the Gaussian kernel, $k_p = 1$, and $C = 50$. The 3FCV is used to calculate the CA. Table 5-12 shows the results averaged over 30 trials. The definition of CA(%) is given in Section 5.1.1. The column labeled "All features" indicates the results of SVM classification using all the features. We see that the CA values are quite low for all conditions. Some are even as low as 50%, which is unacceptable. Relatively large standard deviations are also observed; these suggest that the data points may not be distributed evenly in the original feature space. The CA is significantly improved by using the proposed feature selection algorithm for all conditions. We see that the CA values are all greater than 95%, and the corresponding standard deviations are reduced. Moreover, all these good results are provided by feature sets with less than 15 features, five of which contain fewer features than 5. From these observations, we can conclude that the proposed algorithm is capable of significantly enhancing classification performance for the given damage level classification problem.

Table 5-12: Results of classification using the proposed feature selection algorithm

Conditions	All features		The proposed algorithm				
	CA (%)		CA (%)			Number of features	
	Mean	Std.	Mean	Std.	PBP	Mean	Std.
300rpm & noload	63.50	5.82	99.75	1.01	100	2	0.60
300rpm & load	50.67	9.91	96.58	5.34	100	4	3.93
600rpm & noload	72.42	8.39	99.58	1.33	100	2	3.38
600rpm & load	54.67	8.11	97.67	2.86	100	10	8.87
900rpm & noload	65.83	8.16	100	0	100	1	0.18
900rpm & load	68.67	9.28	96.25	4.34	100	14	6.78
1200rpm & noload	59.00	11.83	99.25	1.99	100	2	1.87
1200rpm & load	64.42	7.90	100	0	100	7	4.89

5.5.2.3 Discussions

From Table 5-12, it is seen that, given the same speed, the degree of improvement in CA is greater for noload conditions than for load conditions. In addition, the dimensions of resultant feature subsets are larger for load conditions than for noload conditions; the same is true of their standard deviations. These observations indicate that the classification problem may become more difficult when a load is applied. For this reason, we examined the noload condition and the load condition separately in this section.

(1) Analysis of the resultant feature subsets for noload conditions

The feature subsets of 900 rpm were analyzed first. It was seen that about one feature on average was selected in this condition. We examined the composition of these subsets over the 30 trials and found the results were quite consistent. Feature #98 was selected for 23 trials, feature #96 was selected for 3 trials, feature #28 was selected for 2 trials, and feature #132 and a combination of features #86 and #90 were both selected for 1 trial. Based on Table 5-11, it was seen that apart from features #86 and #90, most were either standard deviation frequency (F30) or frequency center (F28). Because feature #98 was selected for most of the trials, out of interest, we plotted the classification results using this feature in Figure 5-19. It exhibited a good separation of data points for different damage degrees.

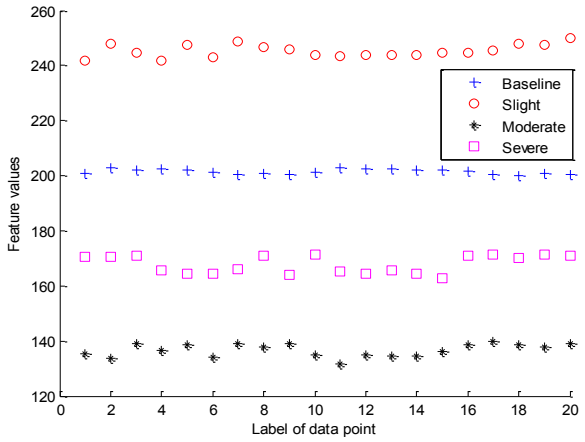

Figure 5-19: Classification results using standard deviation frequency of HS1 under 900 rpm & noload condition

For the 300 rpm, the resultant feature subsets showed two compositions. One contained only feature #96 for 11 trials and the other contained both features #92 and #126 for 17 trials. For the 600 rpm, the resultant feature subsets were a bit various. We found that features #28 and #96 were selected for the most trials, 14 and 10, respectively. These two were both frequency center (F28) but from different accelerometers. For the 1200 rpm, the most frequently selected features were #96 and #98 which were selected for 11 and 10 trials respectively. Based on the above observations, we concluded that the frequency center (F28) and the standard deviation frequency (F30) were the most useful features for noload conditions. To test this conclusion, classification using features #28 (F28 from LS1), #96 (F28 from HS1), and #98 (F30 from HS1) was conducted for the same 30 sets of training and testing data. The resulting classification accuracies were comparable to those obtained using the proposed method.

(2) Analysis of the resultant feature subsets for load conditions

For each speed, the resultant feature subsets over 30 trials were not as consistent as those from noload conditions. The resultant subsets usually contained more than one feature and displayed a relatively high variety. We still, however, found some features appearing in the subsets with high frequencies for all speeds; these included features #29 (F29 from LS1), #30 (F30 from LS1), #96 (F28 from HS1), #129 (F27

from HS2) and #133 (F31 from HS2). It was apparent that these features were all frequency domain features. Similarly, studying the classification results from using these features under load conditions, we obtained classification accuracies that were not as good as those using the proposed method; they were, however, all over 90%, which was acceptable.

5.5.3 Diagnostics of Slurry Pump System

This section presents the results of classifying impeller damage level in a slurry pump using the SVM-based diagnostic algorithm. Since one outlier has been identified from the pre-processed slurry pump dataset (see Section 5.5.1), this section directly conducts feature selection for the cleaned dataset. Accordingly, the transitional dataset can be built by removing data point #65 from the pre-processed dataset. The proposed feature selection algorithm is then applied to the transitional dataset. The selected features are F1, F2, F10, F19, F16, F20, F18, F21, F15, F5, F6, F3, and F4, which reduce the dimension of the original feature space by 8. Table 5-13 shows the results of the given classification problem.

The diagnostic results are given in the row labeled "Combination", which signifies that the proposed data cleaning and feature selection methods were used together for diagnostics. It is seen that using both proposed algorithms reduces the MR value by more than 5%, and reduces the dimension of the feature space by 11, compared to the values given by the original SVM. The results from using only data cleaning are given in the row labeled "Data cleaning (only)", and those from using only feature selection are given in the row labeled "Feature selection (only)". We see that data cleaning reduced the MR value by 3%. Feature selection did not reduce the MR value, but the dimension of the feature space was reduced by 5. Note also that the selected useful features for "Feature selection (only)" and "Combination" are not consistent. This is straightforward, because they were determined based on a different database, the pre-processed dataset and the transitional dataset, respectively. Such a difference implies that outliers should not be ignored and that data shoud be cleaned before feature selection.

Table 5-13: Results of classification using the proposed data processing algorithm

	Outliers removed	Features selected	MR
Original SVM	N/A	All	14.44%
The proposed data cleaning (only)	#65	All	11.23%
The proposed feature selection (only)	N/A	F10, F1, F14, F2, F16, F18, F17, F19, F20, F3, F15, F6, F5, F21, F4, and F7	14.44%
The proposed diagnostic algorithm	#65	F16, F21, F20, F6, F14, F8, F9, F11, F12, and F17	8.99%

5.6 Summary

This chapter presents an SVM-based diagnostic algorithm which contains SVM-based data cleaning algorithm and SVM-based feature selection algorithm that we propose. The former was inspired by observing that outliers close to the separating plane are misclassified by the SVM classifier. The latter was based on the weight vector of the SVM model which relates to the margin of SVM separation. Benchmark datasets have been used to validate these two proposed algorithms. Comparing both with reported methods demonstrates the effectiveness of the proposed algorithms. The proposed data cleaning algorithm is also used for the given slurry pump and the proposed feature selection algorithm for the given planetary gearbox. The results show the good potential of these two algorithms for engineering applications. Our results show also that employed together to classify impeller damage level in the slurry pump system, these algorithms achieved better diagnostic results than either could when used alone.

Chapter 6

Support-Vector-Machine-Based Prognostics

SVM-based prognostics employs SVM as a predictor to forecast condition indicator values based on which remaining useful life (RUL) can be estimated for a system. It is obvious that prediction results depend on the behavior of the SVM model, as determined by its parameters. One challenge is that random noise usually appears in the observations of condition indicators, and this can cause over-fitting or under-fitting when model parameters have not been appropriately selected. In the literature, two types of method are reported for selecting model parameters in order to relieve noise effects. One is the analytical method, and the other is the optimization-based method. The analytical method employs explicit expressions to compute SVM parameters directly, where statistical measures such as noise standard deviation are usually adopted [29–32]. This method can adjust the parameters as data are updated, but the obtained parameters are not optimal. The optimization-based method employs an optimization process to obtain parameters [24, 25], but do not contain relevant terms for dealing with noise effects and are unable to automatically adjust SVM parameters when data are updated.

This chapter presents an SVM-based online prognostic algorithm where SVM model parameters are determined by one proposed analytical method and one proposed optimization-based method. The proposed analytical method [110] focuses on selecting the regularization parameter, C, of the SVM model, which is modified from a reported method by introducing a measure of noise into the selection model. The proposed optimization-based method [111–113] adopts a new optimization model

and is incorporated with the cumulative sum (CUSUM) technique to adjust intelligently the SVM model parameters when current parameters are inappropriate. The performance of the two proposed methods for condition prognostics are examined using two simulation datasets, and their applications in condition prognostics are given for the two experimental systems of interest.

6.1 Preliminaries

6.1.1 Terminologies

Condition indicator (CI): a measure used to represent a system's health status. It can be either directly observed through monitoring instruments or extracted from raw condition monitoring (CM) data such as vibration signals, acoustic emission signals and particle counter data.

Observation: an observed value for CI. The observations are time series data, stored in sequence with respect to monitoring time. Observations are assumed to consist of the true value for CI and noise.

True value of CI (TVCI): the component of observations, reflecting real system condition without the involvement of noise.

Noise: noisy components appearing in the observations, due to various uncontrolled random factors, e.g. environmental conditions, instrument errors, human error, etc.

Noise level (NL): the standard deviation of noise.

Relative noise level (RNL): the ratio of NL to the standard deviation of observations. This measure is used to represent the noise effects. Compared to the well-known Signal-to-noise ratio (SNR), RNL has a very small definition region [0,1], because the variation of noise is involved in that of observations. Based on RNL, noise effects on observations can be readily evaluated following the rule, "the smaller the RNL value, the smaller the noise effects".

6.1.2 Datasets for Demonstrations

In practice, the TVCIs are unknown, so simulation datasets are very useful for demonstrating the performance of prognostic algorithms. This is because their TVCIs are available. We generate two simulation datasets labeled SD1 and SD2 to mimic the trend of realistic deterioration for a system. These two datasets have univariate input and are generated based on the same base model:

$$y_t = y_{t(\text{True})} + \delta_t \tag{6.1}$$

where $y_{t(\text{True})}$ represents the TVCI for time t, and δ represents the noise term.

SD1: a simulation of a monotonic trend of CI values the function of which is given as:

$$y_t = 10 + 10^{-3}e^t + c\delta_t \tag{6.2}$$

where δ_t represents additive noise following the Gaussian distribution with a mean of zero and a standard deviation of one, and c represents the expected NL. For SD1, we generated six sets of data based on RNL values, 0.2, 0.3, 0.4, 0.5, 0.6, and 0.7. Data length was 400 for each of the six datasets. Figure 6-1 shows a plot of SD1 with RNL=0.4.

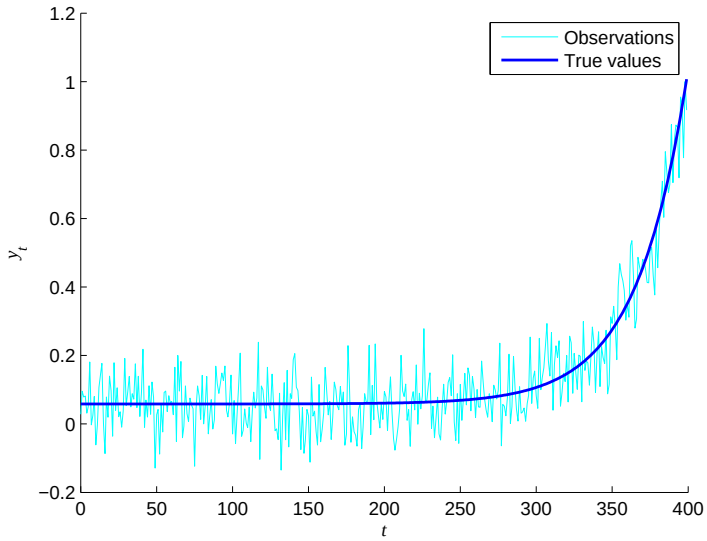

Figure 6-1: Plot of SD1 (RNL=0.4)

SD2: a simulation of a more complicated situation where a deterioration trend contains fluctuations due to seasonal components, the function of which is given as:

$$y_t = 10 + \sin(2\pi f t) + 10^{-3} e^t + c\delta_t \tag{6.3}$$

where f represents the sampling frequency (which is 500 for SD2). Other parameters have the same meanings as those of SD1. Six sets of data were generated as they were for SD1. Data length was 1000 for each set. Figure 6-2 shows a plot of SD2 with RNL=0.4.

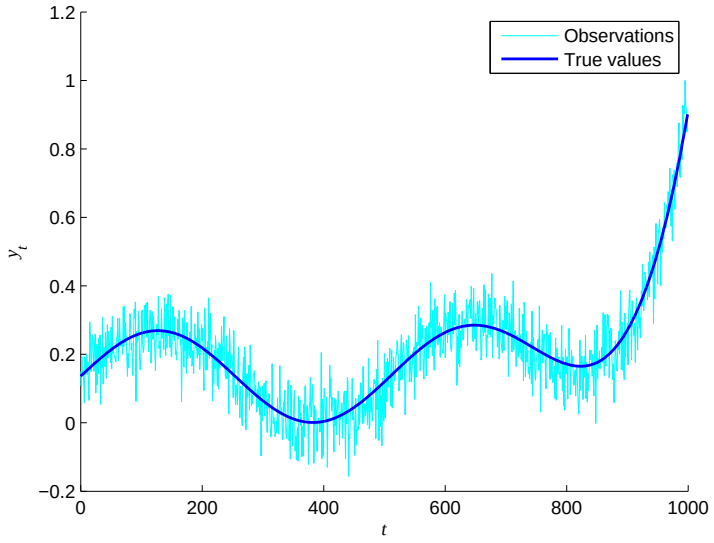

Figure 6-2: Plot of SD2 (RNL=0.4)

6.2 SVM-Based Prognostic Algorithm

The problems of system prognostics are diverse. In this research, we focus on addressing problems subject to the following assumptions.

1. One CI is always available for use. It can be either directly observed through instruments or extracted from CM data.

2. Observations of CI are not noise-free. The noise is due to various independent random factors. It is known from the central limit theorem [72] that irrespective of the distribution followed by the each random factor, the combination

of distributions can be reasonably approximated by a Gaussian distribution with a certain mean and a certain standard deviation.

3. Observations of CI are time series data which are subject to a deterioration trend, which is not necessarily strictly monotonic.

4. Monitored systems can be expected to work properly for a long enough time to ensure there will be a number of observations available for utilization.

5. There is a threshold for the boundary of system deterioration. Once a predicted CI exceeds this threshold, a fault with an unacceptable level or a failure has been detected and the system must be shutdown for maintenance immediately.

Multiple CIs could be used for prognostics. Monitoring and predicting multiple CIs at the same time may capture different characteristics of machine damage mechanism. To make a maintenance decision needs jointly considering every single CI. However, this thesis focuses only on the single CI case. The multiple CIs cases may be addressed in future. Consider that a time series of observations, $\boldsymbol{Y}_t = [y_1, y_2, \ldots, y_t]$, is known, where y_t is an observation at time t. Based on assumption 2, these observations can be modeled in the same way as shown in Eq. (6.1) where δ_t represents the noise following a Gaussian distribution with mean zero and a certain NL, $\sigma_{t(\text{Noise})}$. We further hypothesize that the $\sigma_{t(\text{Noise})}$ value may change over time, but a particular $\sigma_{t(\text{Noise})}$ will remain stable for a certain time span.

Therefore, we formulate the SVM-based prediction for the time point at a-step-ahead t as:

$$\hat{y}_{t+a} = \Theta(\boldsymbol{Y}_t^b, p_{\text{SVM}}) \tag{6.4}$$

where $\hat{y}_{t+a}$ represents the prediction for the time point a-step-ahead t, p_{SVM} represents the SVM model parameters, and $\boldsymbol{Y}_t^b$ represents a time series $[y_{t-b}, \ldots, y_t]$ where y_{t-b} represents the observation at b-step behind t. If $a = 1$, the prediction is a one-step forecast, whereas when $a > 1$ the prediction is a multi-step forecast. Although multi-step forecasting may capture some system dynamics, the performance is poor because there is an accumulation of errors [24]. For this reason, we use

$a = 1$ in this research. Eq. (6.4) can be interpreted as predicting $\hat{y}_{t+a}$ using the SVM predictor, $\Theta(\cdot)$, based on observations between time points $t - b$ and t.

To evaluate the predictions, the measure of normalized root mean square error (NRMSE) is used; this is given by:

$$\text{NRMSE} = \sqrt{\frac{\Sigma_{i=1}^{l}(y_{i(\text{True})} - \hat{y}_i)^2}{\Sigma_{i=1}^{l} y_{i(\text{True})}^2}} \tag{6.5}$$

where $y_{i(\text{True})}$ and $\hat{y}_i$ represent TVCI (known from the simulation datasets) and corresponding prediction, respectively. l represents the number of predictions available. It is believed that the smaller the NRMSE value, the better the method performs.

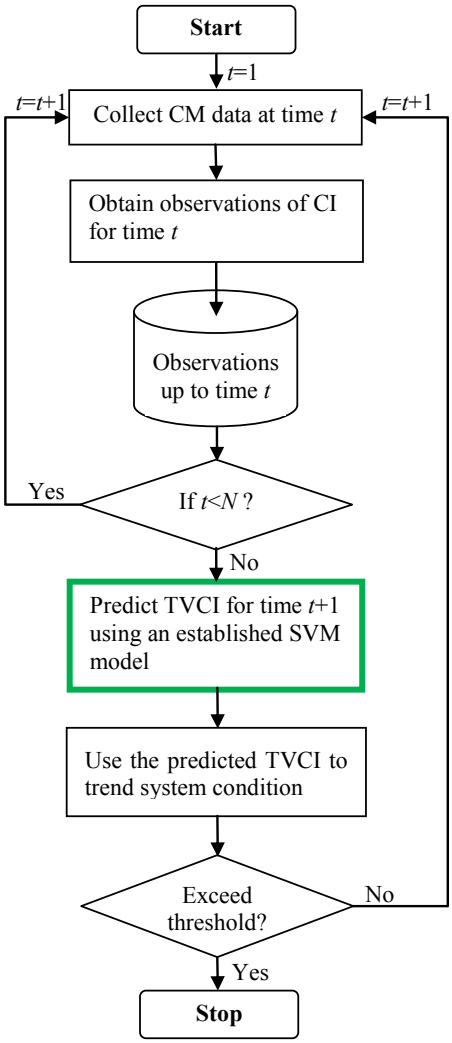

Figure 6-3: Procedure for SVM-based online prognostics

Figure 6-3 shows the procedure for an SVM-based online prognostic algorithm. Each step is explained below. For brevity, some steps (blocks) may be presented together.

Step 1: Collect CM data, obtain observations of CI, and store the observations. Figure 6-3 considers the situation that CI is not available from a direct measurement, so one has to extract/compute CI values from collected CM data. Observations of CI are continuously collected and stored in a dataset until a specified number of observations (N) are available. This is subject to assumption 4 which enables sufficient data to be used in the steps that follow.

Step 2: Check data sufficiency. The relationship between t and N is checked (t is a time label beginning with 1). If $t < N$, observations will be continuously gathered and stored. If $t = N$, SVM parameters will be determined for the first time in the next step. If $t > N$, observations will be directly used to train the SVM model for prediction. The observations in the dataset can be transferred through this step without any loss.

Step 3: Determine SVM parameters, build the SVM model using these parameters, and predict TVCI using the built the SVM model. The two proposed methods for determining SVM parameters are used in this step. The rationale for the proposed methods, and the procedure of implementation are presented in detail in the sections below.

Step 4: Check to see if the predicted TVCI exceeds a specified threshold. If yes, the whole procedure is stopped; otherwise, repeat Steps 1 to 3.

6.2.1 SVM Prediction Using Analytical Method

This section presents an analytical method of selecting the regularization parameter of the SVM model, parameter C. This method is a modification of an existing method [30]. Figure 6-4 shows the procedure for obtaining predictions using the proposed analytical method which can serve in the online prognostic algorithm, specifically in the bold block of Figure 6-3.

Figure 6-4 illustrates that one determines SVM parameters first, then uses those parameters to build an SVM model, and ultimately uses the built model to predict the TVCI value.

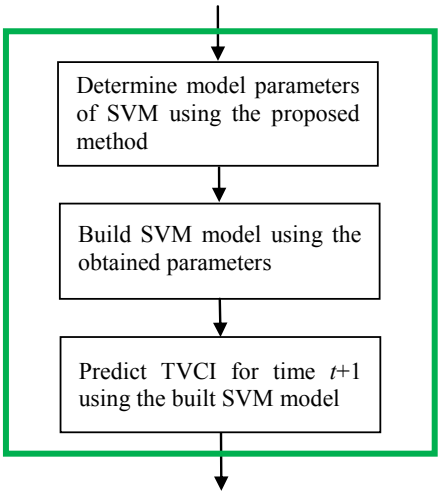

Figure 6-4: Procedure of obtaining TVCI using the proposed analytical method

It is reported in [30] that the optimal choice of parameter C can be derived from standard parameterization of the SVM solution according to Eq. (3.47); this is given by:

$$|f(\boldsymbol{x})| \leq |\Sigma_{i=1}^{p}(\alpha_i - \alpha_i^*)K(\boldsymbol{x}_i, \boldsymbol{x})|\,, \tag{6.6}$$

where p represents the number of support vectors. Incorporating the constraints of dual variables as given in Eq. (3.38), one has:

$$|f(\boldsymbol{x})| \leq pC\,|K(\boldsymbol{x}_i, \boldsymbol{x})|\,. \tag{6.7}$$

Eq. (6.7) can be transformed to:

$$\frac{|f(\boldsymbol{x})|}{M\,|K(\boldsymbol{x}_i, \boldsymbol{x})|} \leq \frac{p}{M}C. \tag{6.8}$$

where M represents the number of training data, so that $\frac{p}{M}$ represents the proportion of support vectors in the training data. Because p is always no larger than M, this proportion ranges between [0,1]; hence, it yields:

$$\frac{|f(\boldsymbol{x})|}{M\,|K(\boldsymbol{x}_i, \boldsymbol{x})|} \leq C. \tag{6.9}$$

In Eq. (6.9), $K(\boldsymbol{x}_i, \boldsymbol{x})$ is determined by a support vector $\boldsymbol{x}_i$, and a new input data point $\boldsymbol{x}$. The input of the new data point is easy to obtain, but the support

vector $\boldsymbol{x}_i$ is unavailable before the training process. Hence, the product of M and $K(\boldsymbol{x}_i, \boldsymbol{x})$ is not easy to determine. Because the product involves the size of the training data, M, it could be very large, making the LHS of the inequality very small. As reported in [84], when parameter C was very small compared to $|f(\boldsymbol{x})|$, it would be impossible to obtain a good prediction, therefore Eq. (6.9) can not be used. This is in accordance with the results of our independent experiments. A large C value is more favorable, and the following inequality can be used:

$$|f(\boldsymbol{x})| \leq C. \tag{6.10}$$

The parameter C can be selected using the lower bound of Eq. (6.9); this is given by:

$$C = \max |y| \tag{6.11}$$

where y instead of $f(\boldsymbol{x})$ represents the output of the training data for the sake of simplicity. The authors of [30] claimed that the range in the training outputs is very sensitive to outliers, so they proposed to computing parameter C using the following expression:

$$C = \max(|\overline{y} + 3\sigma_y|, |\overline{y} - 3\sigma_y|), \tag{6.12}$$

where $\overline{y}$ and σ_y represent the mean and the standard deviation of the training outputs, respectively.

It is reported in [32] that parameter C is associated with the size of the ϵ-insensitive zone, which in [82] was discovered to be proportional to NL. These two observations together suggest that the factor of NL should be considered in the selection model of parameter C. For these reasons, we propose the following model for parameter C:

$$C = \max(|\overline{y} + 3\sigma_y|, |\overline{y} - 3\sigma_y|)g(\sigma), \tag{6.13}$$

where $g(\sigma)$ is a function of σ, namely NL. It is believed that when NL is small the optimal parameter C tends to be large making the product of C and the empirical

risk comparable to the structural risk; it is thus reasonable to define σ inversely proportional to parameter C.

The following shows how we come up with the proposed analytical model for selecting parameter C. The procedure is introduced in detail but the numerical tests are not given due to the thesis length. First, a constant coefficient is applied to the function, and because parameter C should be inversely proportional to σ, the underlying function of $g(\sigma)$ is expressed as,

$$g(\sigma) = \frac{a}{\sigma} \tag{6.14}$$

Based on our experiences, the prediction results of SVM are sensitive to the C values in ten orders of magnitude. Thus, we use a series of values, $a = 10^{i-1}, i = 1, 2, \ldots, 8$, to conduct "trials and errors" tests. SVM is used with these eight values. Normalized root mean square error (NRMSE) is employed to assess the prediction results. The datasets used for testing are SD1 and SD2. The results are compared with the one reported in [30]. Unfortunately, we found that none of the used coefficients are able to provide consistent good prediction results.

Constant coefficients seem not working. As we know, NL is popularly used to measure the magnitude of noise; however, the noise effects on training outputs should not rely only on the NL but also on the magnitude of the training outputs. We can not say that noise with a small NL value produces little noise effect, because this is not true when the standard deviation of the training outputs is also very small. This concern inspires us to define the RNL measure as given in Section 6.1.1. The RNL is defined as the ratio of noise level to standard deviation of observations and it ranges from 0 to 1. A small RNL value represents small noise effects and vice versa. Because RNL is varied according to the data, it is more sensitive compared to the constant coefficient. Now, $g(\cdot)$ becomes a function of RNL. Following the same rationale of the constant coefficient case, the "trials and errors" tests are conducted for three forms of $g(\text{RNL})$ tested,

$$g(\text{RNL}) = \frac{1}{\text{RNL}^{0.5}} \text{ or } \frac{1}{\text{RNL}} \text{ or } \frac{1}{\text{RNL}^2}, \tag{6.15}$$

where RNL is placed in the denominator to make σ inversely proportional to the C. The numerical test results show that the first expression gives consistently good results, so the proposed analytical model for selecting parameter C is expressed as:

$$C = \max(|\overline{y} + 3\sigma_y|, |\overline{y} - 3\sigma_y|)\mathrm{RNL}^{-0.5} \tag{6.16}$$

$$= \max(|\overline{y} + 3\sigma_y|, |\overline{y} - 3\sigma_y|)\sqrt{\sigma_y}/\sqrt{\sigma}, \tag{6.17}$$

where it is seen that the max term is scaled by a factor relevant to the root of NL.

Over-fitting caused by noise is a usual problem that one has to face when doing prediction. SVM handles this problem by selecting appropriate user-defined model parameters. This section proposed an analytical method to calculate the appropriate values for parameter C. Methods reported in literature mainly relate parameter C to the magnitude of the observation of condition indicator. Our studies reveal that parameter C is also dependent on the magnitude of noise existing in the observations. RNL is thus defined and incorporated into the model for selecting parameter C. The proposed analytical method takes into account the noise effects in prediction and enables the parameter C value to be adjusted along with the variations of noise magnitude. It is a valuable contribution to analytically selecting SVM model parameters for prediction.

6.2.2 SVM Prediction Using Optimization-Based Method

This section presents an intelligent method of selecting SVM model parameters that is based on a developed optimization model which takes the noise effects into account in its objective function.

The least square SVM (LSSVM) for prediction is adopted because it has performed well for data with Gaussian noise. In order to accommodate variations in observations and noise, the cumulative sum (CUSUM) technique is employed. By this means, the re-determination of LSSVM parameters can be triggered when unallowable cumulated variations are detected. Figure 6-5 shows the procedure for obtaining a prediction which corresponds to the bold block of Figure 6-3. Figure 6-5 is explained step by step below. The label for each step begins with a 3, indicating a sub-step of Step 3 which is described in Section 6.2.

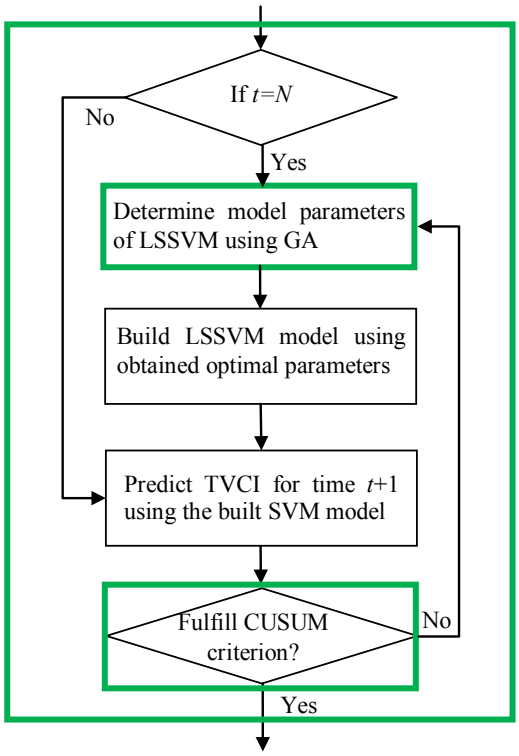

Figure 6-5: Procedure for obtaining TVCI using the proposed optimization-based method

Step 3-1: An optimization process is triggered when the condition $t = N$ is satisfied or the CUSUM criterion (Step 3-4) is not satisfied. This optimization problem is solved using genetic algorithms (GA). The MATLAB's GA toolbox is used to achieve the GA optimization. Figure 6-6 shows the procedure which is described step by step below. One can refer to MATLAB's Help document for the parameter settings of each step.

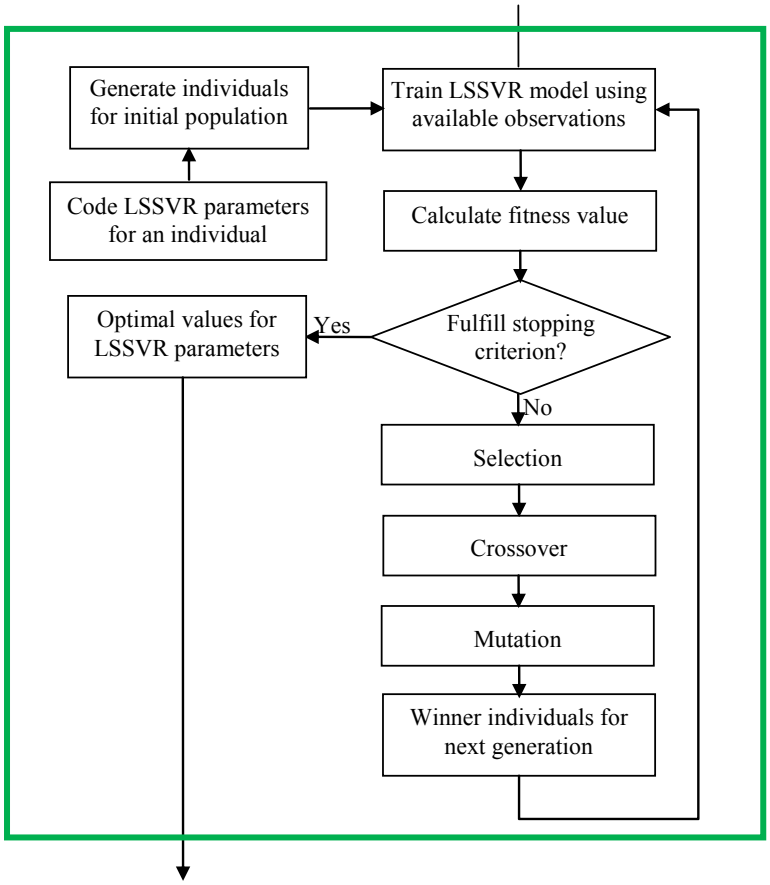

Figure 6-6: Procedure of GA optimization

Step 3-1-1: Code. LSSVM parameters are decision variables for the optimization problem. A real-value coding strategy is adopted; this means that parameter values are real numbers during the optimization.

Step 3-1-2: Generate initial population. An individual (chromosome) is composed of LSSVM parameters in the form of $[C, k_p]$ where k_p represents the parameter of the kernel function. A number of individuals (100) are generated at random for the initial population.

Step 3-1-3: Train the LSSVM model. The LSSVM model with the parameters carried by each individual is trained based on available observations.

Step 3-1-4: Calculating fitness value. Fitness function is the NRMSE for regular optimization-based methods. Such fitness values are easy to compute; however, the predictions obtained tend to over-fitting due to noise. We propose a new fitness function that addresses this problem.

The fitness function is designed to compel the difference between observations and corresponding predictions as close to noise as possible. The mathematical expressions of the fitness function and the constraints are given as:

$$\text{Minimize} \quad \left| \sqrt{\frac{1}{l}\Sigma_{i=t-l+1}^{t}[\Delta y_i - \frac{1}{l}\Sigma_{j=t-l+1}^{t}(\Delta y_j)]^2} - \hat{\sigma} \right| + \left| \frac{1}{l}\Sigma_{j=t-l-1}^{t}(\Delta y_j) \right| \tag{6.18}$$

$$\text{Subject to} \tag{6.19}$$

$$0 < C < C_{\text{Lim}}, \tag{6.20}$$

$$0 < k_p < k_{p(\text{Lim})}, \tag{6.21}$$

$$\Delta y_i = y_i - \hat{y}_i, \tag{6.22}$$

$$\hat{y}_{i+1} = \Theta(\boldsymbol{Y}_i^b; C, k_p), \tag{6.23}$$

where l represents the number of predictions needed for computation, the subscript of "Lim" represents the upper bound of parameters, $\hat{\sigma}$ represents the estimate of NL, and Δy_i represents the difference between observation and corresponding prediction for time i.

Based on Eq.(6.1), if Δy_i is equal to the noise term, δ_i, the TVCI, $y_{i(\text{True})}$, can be obtained given the observations. Because Gaussian noise can be represented by its mean and standard deviation, we build the fitness function in such a way as to enable Δy_i to have values identical to the mean and the standard deviation of δ_i. This fitness function thus allows SVM predictions to reach or approach the TVCIs. In Eq.(6.18), the term in the first absolute sign is the difference between the standard deviation of Δy_i and the estimated NL. The NL is obtained using an estimation method that is described below. The term in the second absolute sign is the difference between the mean of Δy_i and the ideal mean of noise, zero. We expect a scenario where the fitness function returns a zero value which yields $\hat{y}_i = y_{i(\text{True})}$, but realistically, this is unlikely to happen due to the randomness of noise. A more realistic expectation is for $\hat{y}_i$ to be as close to $y_{i(\text{True})}$ as possible. Since the fitness function aims to narrow the

deviations between the predictions and the TVCIs rather than between the predictions and the observations, it reduces the risk of over-fitting.

The estimated NL is obtained using k-nearest-neighbors regression [30]:

$$\hat{\sigma} = \sqrt{\frac{M^{1/5}k}{M^{1/5}k - 1}\frac{1}{M}\Sigma_{i=1}^{M}(z_i - \hat{z}_i)^2} \tag{6.24}$$

where $\hat{z}_i$ represents the estimate of the data point z_i obtained by k-nearest neighbor regression, k is the number of nearest data used to estimate z_i, and M is the sample size of the whole dataset.

Step 3-1-5: Selection. A stochastic uniform method is employed to select individuals. This is a default option of GA toolbox.

Step 3-1-6: Crossover. An arithmetic crossover is employed to create children (new individuals) that are the weighted arithmetic mean of two parents (old individuals). The weight is 0.8 for the individual with a greater fitness value and is 0.2 for the other individual.

Step 3-1-7: Mutation. A uniform mutation is employed to provide the necessary diversity of population. The mutation fraction is set at 0.1.

Step 3-1-8: Stopping criterion. The optimization process stops if there is no improvement in the objective function for 10 generations.

The reasons of using GA to solve the optimization problem is as follows. For the given optimization problem, the decision variables are the model parameters of SVM and the objective function is not differentiable with respect to the decision variables. For this reason, the optimization algorithms requiring function differentiability are not applicable. The algorithms that do not require strict mathematical properties of the objective function and the decision variables are needed. These algorithms may include genetic algorithm (GA), particle swarm optimization, ant colony algorithm, simulated annealing, etc. GA is finally chosen because it is widely acknowledged as being able to ensure general global optimality, without limitation for use. In addition, many GA toolboxes are available for MATLAB programming.

For the sub-steps of Step 3, several GA parameters are specified. These parameters control the GA optimization process. Different selections of them

may generate distinct results. For example, a certain group of parameters may get rid of a local optimality and reach a global optimality. However, this usually needs additional algorithms or multiple implementations of GA. Basically, we do not expect the proposed method much dependent on the parameters of GA optimization, because it will reduce the robustness of the proposed method. Hence, the parameters for GA are all default settings defined in GA toolbox. These settings are not optimal but generally applicable to most of optimization problems.

Step 3-2: Establish an LSSVM model and make a prediction. The LSSVM model is established using the optimal parameters returned by Step 3-1. This model is then used to predict as shown in Eq. (6.4).

Step 3-3: Check CUSUM criterion. CUSUM is used to determine the adequacy of the current LSSVM model and to trigger, if needed, the optimization process (Step 3-1) in order to re-determine the optimal parameters for the LSSVM model based on updated observations.

Suppose that a random variable, Z, is drawn from a normal distribution with a mean of μ and a standard deviation of σ. CUSUM detects the deviation of a certain observation from the mean by:

$$d_i = \frac{z_i - \mu}{\sigma}, \tag{6.25}$$

where d_i represents the multiple of σ that a certain observation, z_i, deviates from μ. In terms of a large sample size, the two statistical parameters, μ and σ, can be replaced with the sample mean and sample standard deviation.

CUSUM uses two sums to detect unallowable deviations. These are given by [114]:

$$UB_i = \max[0, (d_i - m) + UB_{i-1}], \tag{6.26}$$

$$LB_i = \max[0, (-d_i - m) + LB_{i-1}], \tag{6.27}$$

where UB_i is for detecting positive deviation and LB_i is for detecting negative deviation. The values for UB_0 and LB_0 are zero. The value for m is usually selected to be 0.5 which is appropriate for detecting a one-sigma deviation. There is a threshold, h, such that when it is exceeded by either sum, an unallowable deviation is detected. The h is usually selected to be 3 as suggested in [114].

We expect the predictions of LSSVM to be as close to the TVCIs as possible. As a result, the differences between observations, y_i, and corresponding predictions, $\hat{y}_i$, are normally distributed with a mean of zero and a certain standard deviation. CUSUM can be used to control the differences, $y_i - \hat{y}_i$. Figure 6-7 shows the CUSUM criterion in which the μ and σ of the differences up to time t are estimated by sample mean and sample standard deviation:

$$\mu = \frac{1}{t - t_0 + 1} \Sigma_{i=t_0}^{t} (y_i - \hat{y}_i), \tag{6.28}$$

$$\sigma = \sqrt{\frac{1}{t - t_0} \Sigma_{i=t_0}^{t} (y_i - \hat{y}_i - \mu)^2}, \tag{6.29}$$

where t_0 represents the time point that CUSUM begins to operate. Once either of the two sums exceeds the threshold, the sum values are reset to zero.

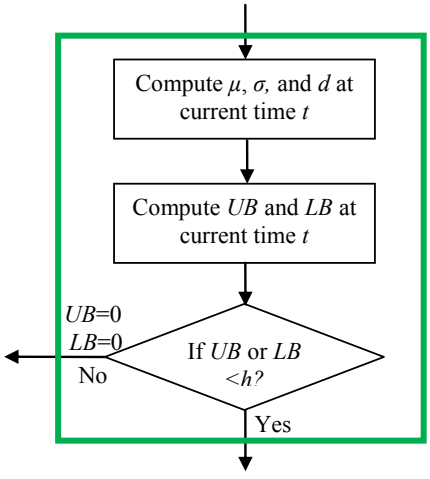

Figure 6-7: CUSUM criterion

This section proposed an optimization-based method to mitigate the noise effects

in prediction. Unlike reported optimization-based methods which determine SVM model parameters merely based on the observations collected in the initial stage, the proposed method employs the CUSUM to update the model parameters. This allows for using the up-to-date information for parameter selection and enables SVM to capture the change caused by noise during the whole process of prediction. More importantly, whenever the predicted errors are out of the specified tolerance, the SVM model is re-established to accommodate the present situation. For this reason, the proposed method has advantage over reported optimization-based methods and is an important contribution to SVM parameter selection for prediction.

6.3 Evaluation of the Proposed Methods

This section examines the two proposed methods using the two simulation datasets, SD1 and SD2. Since our focal point is how close the predicted TVCIs are to the real ones rather than obtaining a RUL for decision making, checking whether a predicted TVCI exceeds the threshold at Step 4 is not included in the examination.

6.3.1 Methods for Comparisons

Some reported analytical methods and optimization-based methods are compared with the proposed two methods. As the proposed analytical method is for selecting parameter C only, other SVM model parameters are selected exactly the same way for all analytical methods in order to ensure a fair comparison, i.e. $\epsilon = \frac{3\sigma\sqrt{\ln(M)}}{\sqrt{M}}$ [30], k_f is the Gaussian kernel, and $k_p = 1$. These selected k_f and k_p were determined by "trials and errors" tests.

Method 1: Analytical method, $C = \max(|\overline{y} + 3\sigma_y|, |\overline{y} - 3\sigma_y|)$ [30].

Method 2: Analytical method, $C = My_{\max}$ [83].

Method 3: Analytical method, $C = y_{\max}$ [84].

Method 4: The proposed analytical method, $\max(|\overline{y} + 3\sigma_y|, |\overline{y} - 3\sigma_y|)\sqrt{\sigma_y}/\sqrt{\sigma}$ [110].

Method 5: GA-based optimization method [24] where the SVM parameters, C, ϵ and k_p are decision variables. The procedure for implementation is similar to

that in Figure 6-6 except that the fitness function in Step 3-1-4 minimizes the NRMSE obtained from a 5-fold cross-validation.

Method 6: This is similar to Method 5 but SVM is replaced with LSSVM because LSSVM is optimal for data with Gaussian noise [115]. The LSSVM parameters, C and k_p, are decision variables.

Method 7: The proposed optimization-based method [113].

The data studied in this chapter are non-stationary. There are other reported methods that can be used for non-stationary time series data prediction such as autoregressive moving average (ARMA) [65] and Kalman filter (KF) [74]. Because the proposed methods aim to improve SVM performance in prediction, it is more reasonable to compare with the methods of the same kind. For this reason, the comparisons with other non-stationary methods such as ARMA and KF are not conducted.

6.3.2 Training Strategy

Training strategy is used only to determine SVM/LSSVM parameters for optimization-based methods, because analytical methods have mathematical expressions for directly computing the parameters. For Methods 5 and 6, cross-validation is used as reported in [24]. One-step ahead prediction is conducted according to Eq. (6.4). Table 6-1 shows the arrangements of training and test data where b is given in Eq. (6.4). The predictions obtained and the test data listed in the last column are input to Eq. (6.5) in order to compute the NRMSE value which is the fitness value for GA-based optimization. The parameters offering the minimum NRMSE value are selected as the optimal parameters. For Method 7, the training and test data are arranged in the same way as for Methods 5 and 6. The procedure for implementing Method 7 can be found in Section 6.2.2.

Table 6-1: Data arrangement for training

Training data				Testing data
y_1	y_2	⋯	y_b	$y_{b+1(\text{True})}$
y_2	y_3	⋯	y_{b+1}	$y_{b+2(\text{True})}$
y_3	y_4	⋯	y_{b+2}	$y_{b+3(\text{True})}$
⋯	⋯	⋯	⋯	⋯
y_{t-b}	y_{t-b+1}	⋯	y_{t-1}	$y_{t(\text{True})}$

The one-step-ahead prediction is conducted in this chapter. However, n-step-ahead prediction would be more useful for prognostics since it predicts the system conditions much ahead of the current time. This allows more time for arranging maintenance actions. Nevertheless, as stated in [24], n-step-ahead prediction is more likely to be corrupted by accumulated errors, so it may need additional methods to work with in order to reach satisfied prediction results. On the other hand, one-step-ahead prediction must be well addressed as it is the foundation of multi-step methods. For this reason, this thesis focuses only on the one-step-ahead prediction problem based on which multi-step problems could be possibly tackled in future.

According to Figure 2-4, one can use one-step-ahead prediction to estimate RUL. However, the condition is that the length of monitoring intervals should not be too small. If the interval is sufficiently large, for example weekly or biweekly, the RUL calculated may be adequate for scheduling maintenance activities

6.3.3 Criteria for Evaluation

The NRMSE is an important measure for evaluating the performance of prediction methods; however, it is not adequate, because a small NRMSE value may correspond to a trend with unacceptable fluctuations which give it a noisy pattern. For this reason, visual comparisons of a predicted trend and its corresponding TVCI trend are given. The criteria are described as follows:

Criterion 1: Given that two predicted trends are similar, one is preferred if it has a smaller NRMSE value than its counterpart.

Criterion 2: Given that two predicted trends correspond to comparable NRMSE values, one is preferred if its trend is smoother than that of its counterpart.

Criterion 3: Trend pattern has a higher priority than does NRMSE value, e.g. a trend with a smaller NRMSE value but a noisier trend is not favored.

For the simulation datasets, the NRMSE values and the plots of trends are both given, but for experimental datasets, only the plot of trend is given since TVCIs are not available in this situation.

6.3.4 Evaluation Using Simulation Datasets

This section shows the results of the seven methods using simulation datasets. The CI observations to be plotted are normalized using $(y - \min(y))/(\max(y) - \min(y))$ where y represents the observation. The NRMSE value is calculated for every RNL case. We show the plots of three RNL cases in which RNL=0.2 corresponds to a case where the noise effect is small and may be ignored, RNL=0.5 corresponds to a case where the NL can not be ignored, and RNL=0.7 corresponds to a case where the noise effect is significant. The last case is somewhat practically inapplicable because noise becomes the dominant component in the signal. We consider this case just for the interest of assessment. In the following, the analytical methods and the optimization-based methods are first compared separately. Later, we summarize in terms of these two categories of method.

The parameters of Methods 5, 6, and 7 for simulation datasets are given in Table 6-2. Parameter N denotes the number of observations available (see Figure 6-3), l denotes the number of predictions used to determine the SVM/LSSVM parameters (see Eq. (6.18)), and b denotes the number of observations used for training (see Eq. (6.4)). Parameter N is selected as being 100 for SD1 and SD2. For Methods 5 and 6, l is set at 5 as suggested in [24]. Because Method 7 uses a different optimization model, l is set at 30 in order to provide sufficient predictions for computing the mean and standard deviation of the noise. The selection of b is subject to a constraint that $b + l$ must be no larger than N. Parameters m and h are for the CUSUM of Method 7 (see Step 3-3 in Section 6.2.2 for their selections).

Table 6-2: Parameter settings for simulation datasets

Dataset \ Method	Methods 5 and 6			Method 7				
	N	*l*	*b*	*N*	*l*	*b*	*m*	*h*
SD 1	100	5	50	100	30	50	0.5	3
SD 2	100	5	50	100	30	50	0.5	3

6.3.4.1 Evaluation Using Simulation Dataset 1

Tables 6-3 and 6-4 lists the NRMSE values and CPU times for all seven methods used with SD1. The column labeled "original" shows the values when the observation values are used for $\hat{y}_i$ in Eq. (6.5). For the analytical methods, it is seen that when RNL is relatively small, i.e. 0.2 – 0.4, all methods provide a larger NRMSE value than those in the original column. When RNL is greater than 0.4, the situation is reversed. We see that Method 4 (the proposed analytical method) provides consistently smaller NRMSE values for all RNL cases than does Method 1. As Method 4 is based on Method 1, the results verify the effectiveness of the modification. This observation is also true of the comparisons with Method 3.

In terms of NRMSE, it seems that Method 2 is the best of the analytical methods. Figures 6-8, 6-9, and 6-10 show the predicted trends with RNL values of 0.2, 0.5, and 0.7, respectively. The "Errors" curve in the legend (dash curve) denotes the difference between the TVCIs and the corresponding predictions. It is seen that Method 2 is quite noisy. According to evaluation criterion 3, this method is not preferred. It is also seen that for Methods 1, 3, and 4 the predicted trends are much smoother than the observations. This is desirable. Nevertheless, the large deviation appears at a time when the TVCI is increasing rapidly. This deviation is believed to be the major cause of NRMSE values greater than the original ones. If such deviations can be corrected, these methods will be preferred.

Table 6-3: Results of SD1 (Methods 1 - 4)

RNL	Original	Analytical methods							
		Method 1		Method 2		Method 3		Method 4	
		NRMSE	CPU(s)	NRMSE	CPU(s)	NRMSE	CPU(s)	NRMSE	CPU(s)
0.2	0.1309	0.2585	3.0108	0.2052	2.9484	0.2630	2.5740	0.2480	2.8860
0.3	0.1753	0.2664	2.8392	0.2166	2.9328	0.2702	2.6364	0.2524	2.8860
0.4	0.2079	0.2434	2.8392	0.2174	2.6520	0.2460	2.7612	0.2363	2.6364
0.5	0.2266	0.1970	2.7768	0.1816	2.6520	0.1989	2.7300	0.1915	2.6208
0.6	0.2438	0.1725	2.8704	0.1721	2.7768	0.1739	2.7768	0.1639	2.7144
0.7	0.2576	0.1395	2.6208	0.1531	2.7144	0.1399	2.5428	0.1382	2.6364

Table 6-4: Results of SD1 (Methods 5 - 7)

RNL	Original	Optimization-based methods					
		Method 5		Method 6		Method 7	
		NRMSE	CPU (s)	NRMSE	CPU (s)	NRMSE	CPU (s)
0.2	0.1309	0.1342	104.645	0.1077	115.9867	0.0881	108.8455
0.3	0.1753	0.1803	74.4281	0.2992	117.8432	0.1306	599.2542
0.4	0.2079	0.1634	105.7843	0.1709	115.3627	0.1548	145.7267
0.5	0.2266	0.1035	262.5092	0.1071	115.7839	0.0823	439.2544
0.6	0.2438	0.1847	121.2128	0.1111	107.7638	0.0814	105.8468
0.7	0.2576	0.1241	163.6294	0.1126	108.8419	0.0755	648.1842

In terms of CPU time, all four analytical methods perform equally. For each of the four methods, the CPU times are comparable for the different RNL values. This indicates that the CPU time is not sensitive to the NL.

For the optimization-based methods, let us consider first the special case of RNL= 0.2, because Method 7 (the proposed optimization-based method) determines the parameters only once; this means that no CUSUM is involved. This special situation allows us to assess the performance of the developed optimization model of Eqs. (6.18) and (6.23). In terms of NRMSE, based on Table 6-4, we see that Method 7 has a smaller NRMSE value than do Methods 5 and 6. In terms of trend pattern, Method 7 is comparable to Method 6 and much better than Method 5; therefore, it can be said that the developed optimization model is effective.

Considering all RNL cases, the three optimization-based methods basically provide NRMSE values smaller than the original ones. It is seen that Method 7 provides

the smallest NRMSE values for all RNL cases. Method 6 is basically better than Method 5. Figures 6-8, 6-9, and 6-10 show that the predicted trends of Method 5 fluctuate more evidently than do those of the other two methods, especially for large RNL cases. Method 5's trends fluctuate even more than do those of Methods 1, 3 and 4. It is seen that these optimization-based methods have a slighter deviation problem than do the analytical methods when the TVCIs are increasing rapidly.

In terms of CPU time, Methods 5 is larger than Method 6 for the large RNL values, 0.5, 0.6, and 0.7. For RNL=0.2 and RNL=0.4, Methods 5 and 6 are comparable. Basically, it is reasonable that Method 5 uses more CPU times than Method 6, since Method 5 has 3 decision variables over 2 of Method 6. However, Method 5 is observed much smaller than Method 6 for RNL=0.3. This may be caused by GA. One possible reason is that the initial population contains individuals that are close to the optimal solution, which make GA converge to the optimal solution quicker than other RNL cases. Method 7 uses larger CPU times than Methods 5 and 6, especially for RNL=0.3, 0.5 and 0.7. This is because it conducts GA optimization multiple times, e.g. 8 times for RNL=0.5 and 12 times for RNL=0.7. When using Method 7, one needs to consider if large CPU time is acceptable.

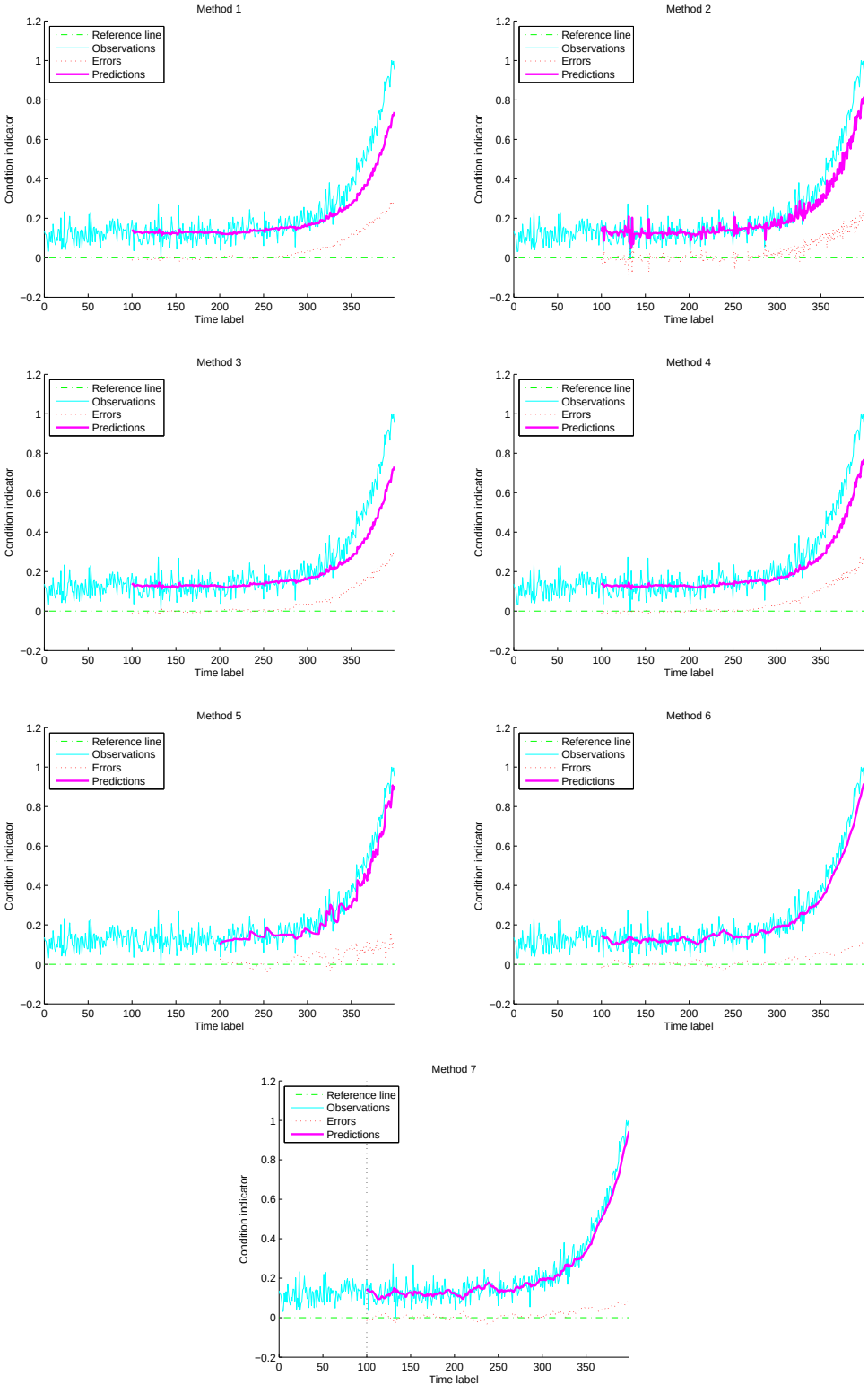

Figure 6-8: Trend prediction (RNL = 0.2)

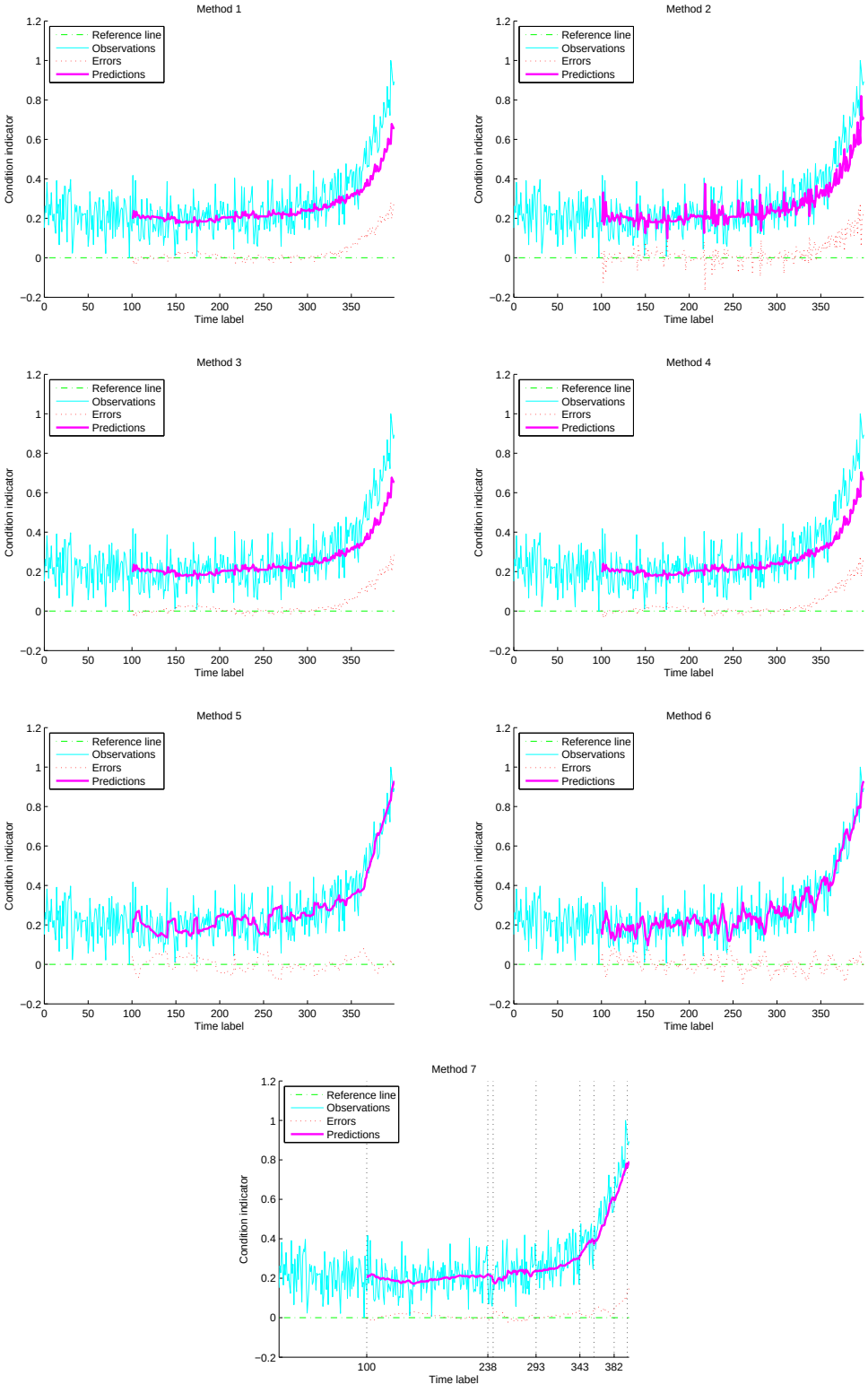

Figure 6-9: Trend prediction (RNL = 0.5)

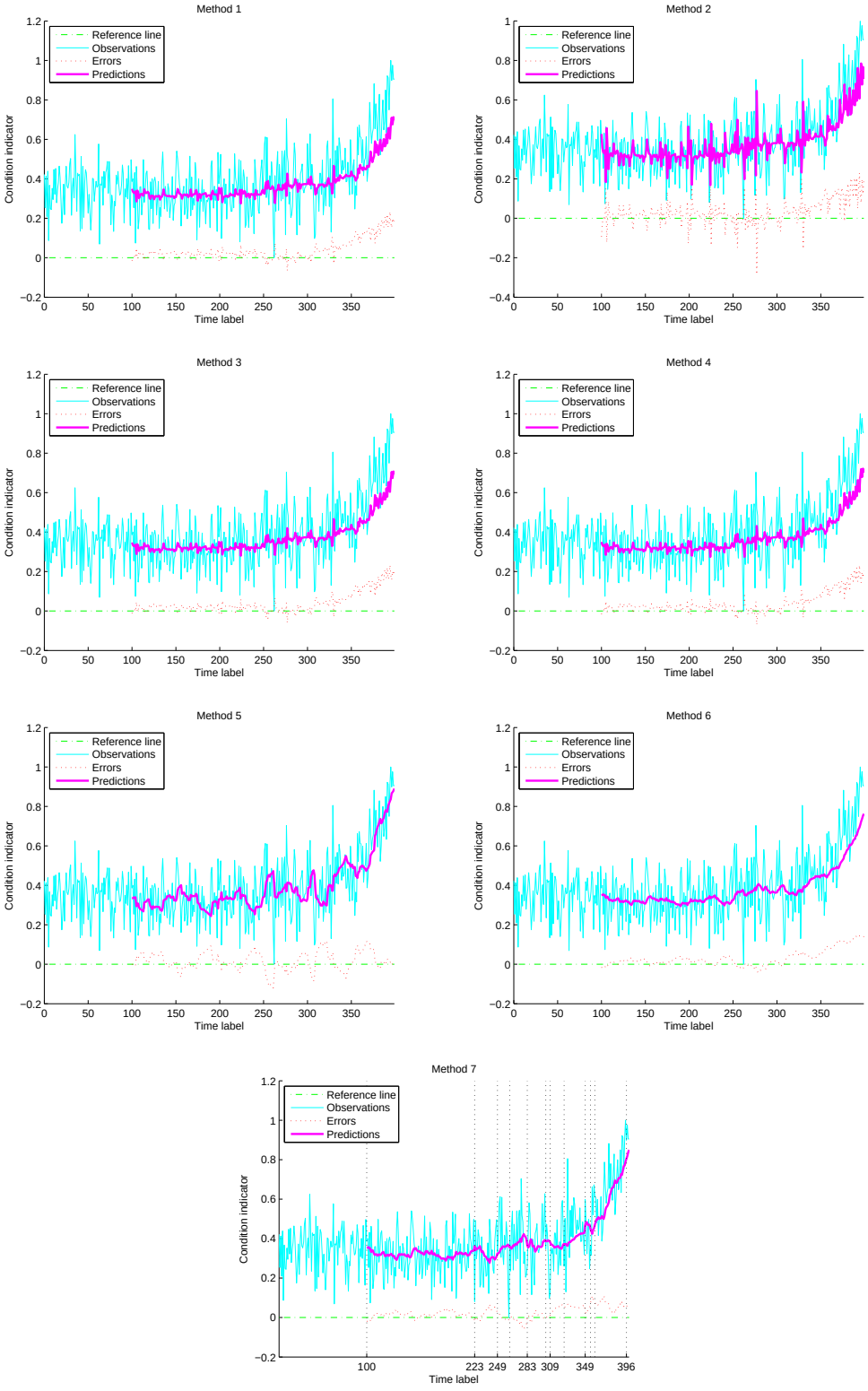

Figure 6-10: Trend prediction (RNL = 0.7)

For the two proposed methods, Methods 4 and 7, the former is much better than the latter in terms of NRMSE. Method 7 has a self-examining mechanism provided by CUSUM. We can see several vertical lines in Figures 6-9 and 6-10 for Method 7. These vertical lines correspond to the optimization process of re-determining LSSVM parameters triggered by CUSUM. Taking the case of RNL=0.7 as an example, Method 7 automatically triggers the optimization processes at times, 223, 249, 283, and so on. This allows the predicted trend to return to the right path without manual interruption when it deviates from the TVCIs. Conclusively, Method 7 has a significant advantage over Method 4 in terms of its NRMSE values and predicted trend patterns. In terms of CPU time, Method 7 uses much larger amount than Method 4 does. When selecting method, One needs to consider both prediction accuracy and CPU time.

6.3.4.2 Evaluation Using Simulation Dataset 2

Tables 6-5 and 6-6 lists the NRMSE values and CPU times for all seven methods for SD2. Basically, the observations are quite similar to those for SD1. For the analytical methods, Method 4 provides smaller NRMSE values than do Methods 1 and 3 for all RNL cases. Methods 1, 3 and 4 perform comparably to each other. Method 2 gives the smallest NRMSE values for RNLs of 0.2, 0.3 and 0.4, but the greatest NRMSE values for the rest of the RNL cases. Figures 6-11, 6-12, and 6-13 show the predicted trends when RNL is equal to 0.2, 0.5, and 0.7, respectively. Method 2 is always nosier than the others, which is not desired. Deviations happen earlier than for SD1 for all methods due to a large scale fluctuation generated by the sinusoidal term in Eq. (6.3). According to the evaluation criteria, Method 4 is the best, because it provides the best balance between NRMSE value and trend pattern.

In terms of CPU time, it is also similar to SD1. The four analytical methods basically use the same amount of CPU time, but the magnitude is larger than that of SD1. This is because SD2 has 1000 data points while SD1 has merely 400 data points.

For the optimization-based methods, Method 7 provides the smallest NRMSE value for all RNL cases. Method 5 seems promising as well; its NRMSE values are

comparable to those of Method 7 for half the RNL cases; however, when it comes to the patterns of predicted trends, Method 5's trends fluctuate more than those of Methods 6 and 7. Methods 6 and 7 offer comparable smooth trends most of the time. For Method 6, deviations become significant after time 800. Method 7 remedies this problem by triggering the optimization processes to re-determine the parameters. It is seen from Figure 6-13 that the Method 6's trend is subject to a small fluctuation throughout its entire time span. This is strangely distinct from its performance for other RNL cases where its trends are quite smooth. To eliminate the effect of singularity, we independently ran Method 6 ten times, but the trend patterns we obtained remained similar. Method 7 also provides a trend with some small-scale fluctuations in several discontinuous time intervals, but these fluctuations were quickly corrected.

In terms of CPU time, it is also similar to that of SD1. Method 5 basically uses the amount of CPU time comparable to and larger than Method 6. Method 7 uses even larger amount of CPU time due to the larger number of optimization runs. For SD2, the largest two CPU time appear at RNL=0.2 and RNL=0.3. This is different from SD1. Figure 6-11 reveals that most of CPU times are used around the end of time series when RNL=0.2.

The following summarizes the performance of the seven methods. In terms of the criteria mentioned in Section 6.3.3, Method 2 is the worst among all four analytical methods as its predicted trend is the noisiest. Compared to Methods 1 and 3, Method 4 provides comparable trends and smaller NRMSE values. Method 7 provides smaller NRMSE values than do Methods 5 and 6, and its trend is as smooth as that of Method 6. In conclusion, Method 4 is the best among the four analytical methods, and Method 7 is the best among the three optimization-based methods. Compared to Method 4, Method 7 is much better in terms of NRMSE and trend pattern. Method 4 is even worse than Method 6. In this sense, the optimization-based methods are superior to the analytical methods, and Method 7 is the best of the all seven methods.

Table 6-5: Results of SD2 (Methods 1 - 4)

RNL	Original	Analytical methods							
		Method 1		Method 2		Method 3		Method 4	
		NRMSE	CPU(s)	NRMSE	CPU(s)	NRMSE	CPU(s)	NRMSE	CPU(s)
0.2	0.0885	0.1295	9.2197	0.1071	8.5801	0.1308	8.8921	0.1239	8.8297
0.3	0.1268	0.1201	8.2837	0.1088	8.3773	0.1211	8.1433	0.1172	8.2681
0.4	0.1584	0.1195	8.1901	0.1151	8.8297	0.1204	8.2525	0.1174	8.3929
0.5	0.1753	0.1028	8.3305	0.1098	8.4397	0.1031	7.9093	0.1020	8.2057
0.6	0.2153	0.1087	8.4397	0.1228	8.6425	0.1089	8.1589	0.1081	8.1589
0.7	0.2305	0.0656	8.1277	0.0948	8.4709	0.0653	7.8781	0.0655	8.0809

Table 6-6: Results of SD2 (Methods 5 - 7)

RNL	Original	Optimization-based methods					
		Method 5		Method 6		Method 7	
		NRMSE	CPU (s)	NRMSE	CPU (s)	NRMSE	CPU (s)
0.2	0.0885	0.0747	285.4662	0.1530	165.6731	0.0751	801.1119
0.3	0.1268	0.0819	142.0701	0.1472	166.6751	0.0544	826.9010
0.4	0.1584	0.0632	133.6733	0.1366	164.1443	0.0619	304.9002
0.5	0.1753	0.0719	267.5573	0.1069	158.4658	0.0579	526.4098
0.6	0.2153	0.1484	166.4687	0.1227	162.1786	0.0635	664.4863
0.7	0.2305	0.0819	161.9758	0.0507	154.4254	0.0428	562.0104

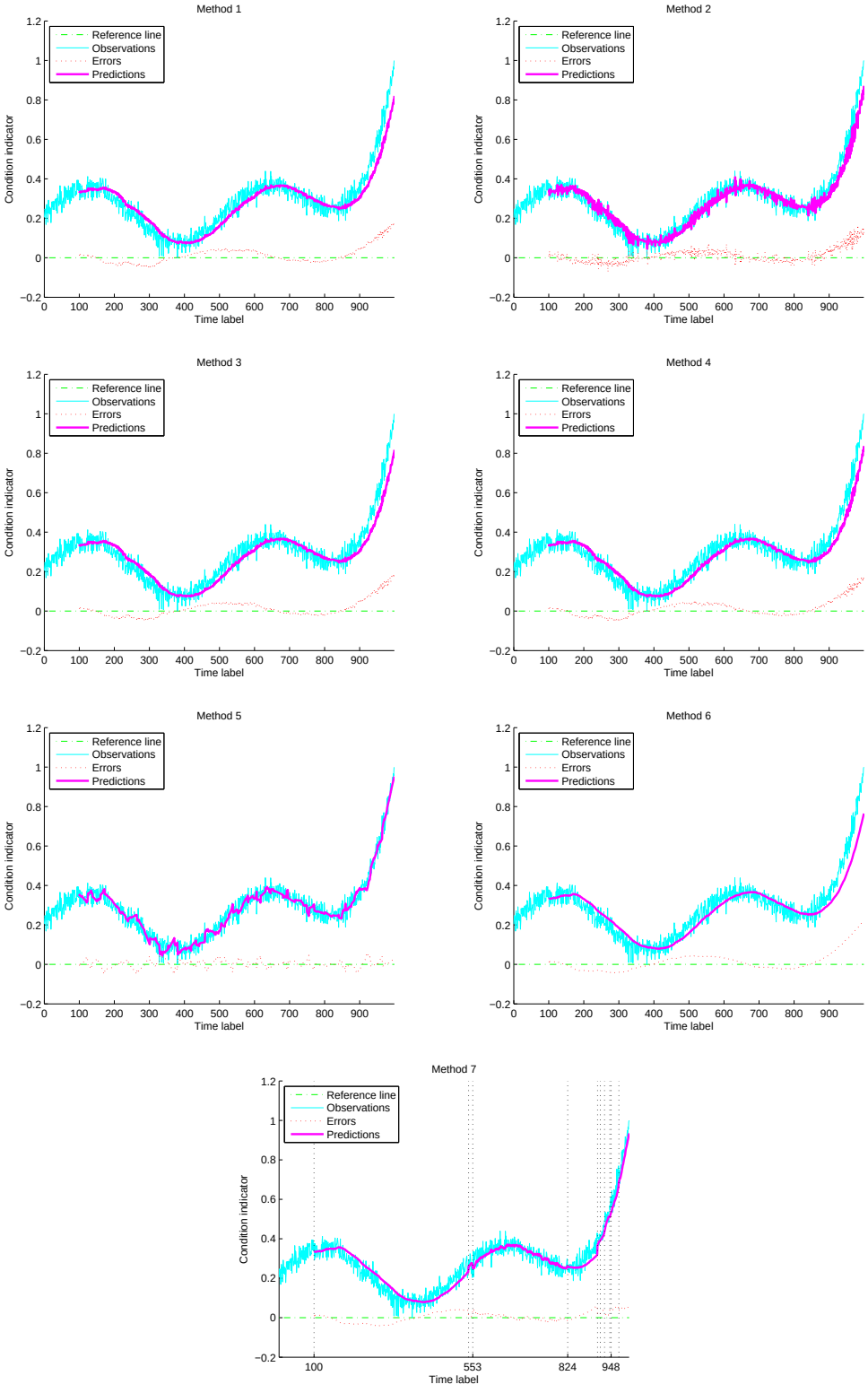

Figure 6-11: Trend prediction (RNL = 0.2)

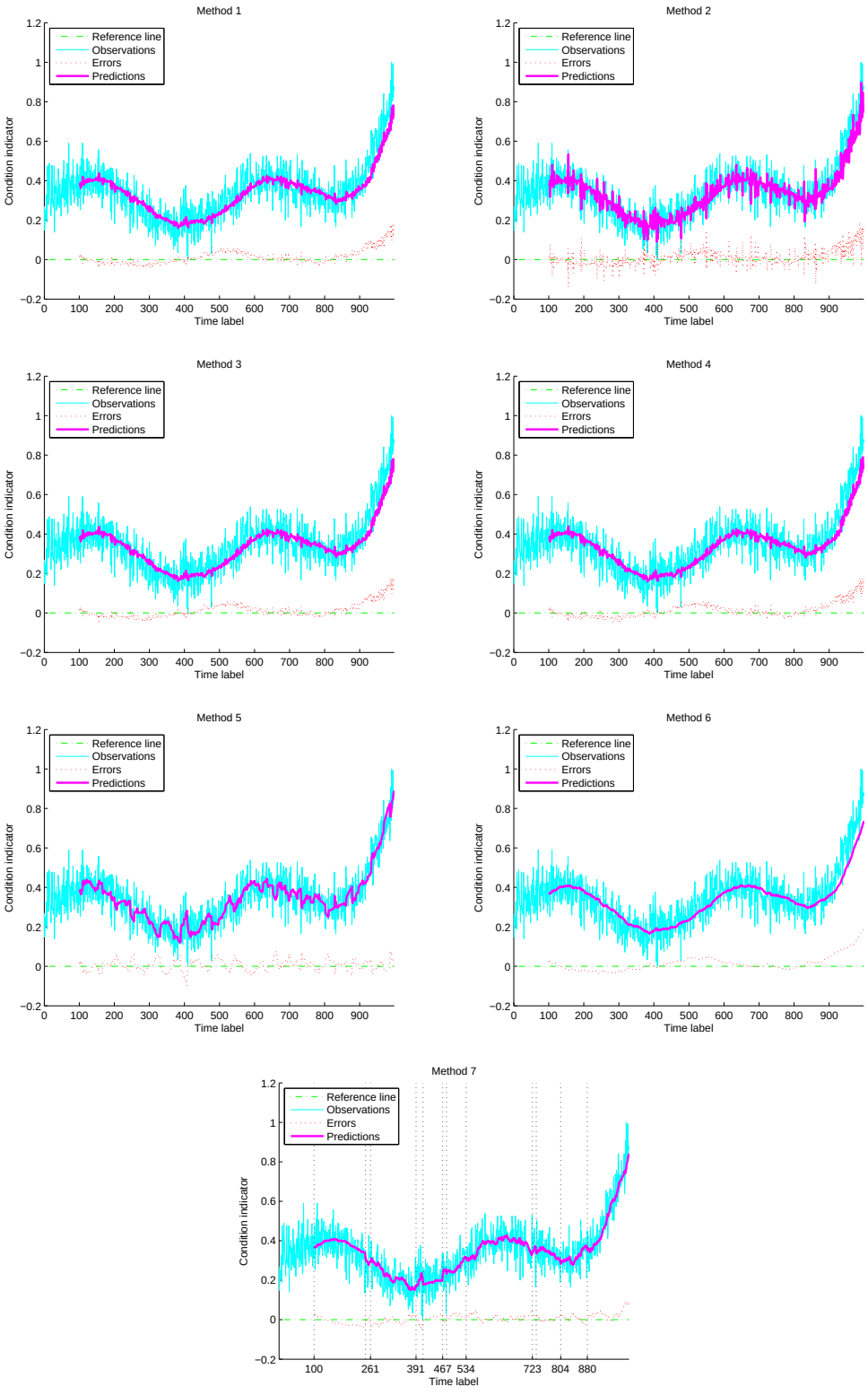

Figure 6-12: Trend prediction (RNL = 0.5)

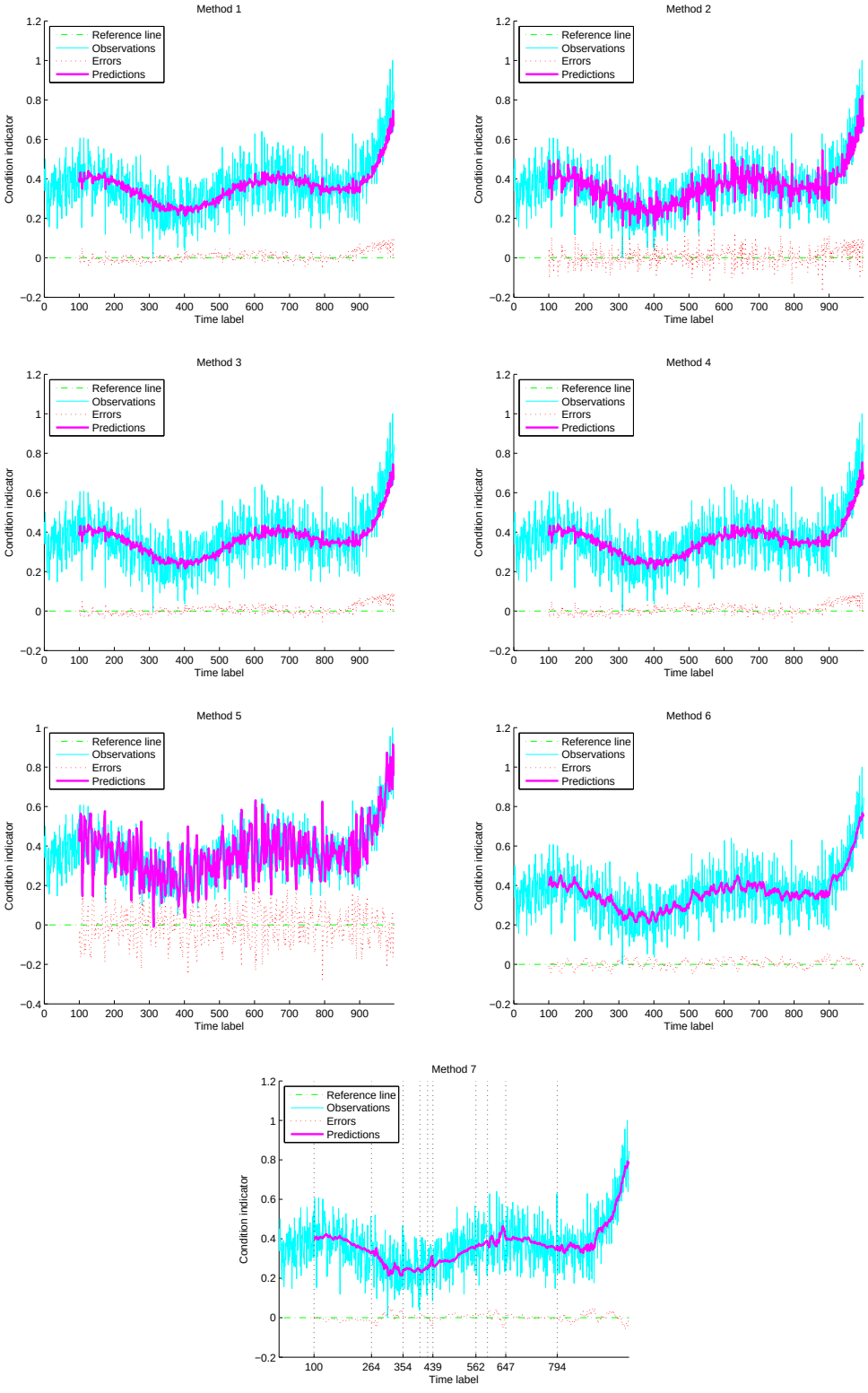

Figure 6-13: Trend prediction (RNL = 0.7)

In terms of the CPU time, the analytical methods, Methods 1, 2, 3 and 4, use comparable amount of CPU time. This amount is much lower than that of optimization-based methods, Methods 5, 6 and 7. Method 6 uses a bit larger amount of CPU time than Method 5 because it has 3 decision variables in optimization over 2 of Method 6. Method 7 may use very large amount of CPU time than Methods 5 and 6. This is dependent on the number of optimization runs it conducts. Generally, optimization-based methods use larger amount of CPU time than analytical methods. When selecting method for a certain problem, one needs to consider both prediction accuracy and CPU time to make an appropriate decision.

6.3.4.3 Discussions

It is seen that the error between TVCIs and predictions increases as time passes. This error becomes larger when the TVCI increases in a high rate. When the TVCI is large to a certain extent the noise level may be ignored. In this case, one needs a large parameter C value to enable SVM to capture the change of the TVCI. Unfortunately, none of the four used analytical methods realized this. For optimization methods, especially Methods 5 and 6, this error still grows larger when the TVCI increases, even though the error value is smaller than those of the analytical methods. This is due to that the SVM model parameters are determined based on the observations of the initial stage (times 1 to 100), and these parameters are not good enough to make SVM capture the change occurring in the end stage. In comparison, the proposed Method 7 performs well, because it responds to the changes by re-determining SVM model parameters.

Basically, the errors are the consequence of the fast increase of TVCI. They can not be eliminated, even if one-step-ahead prediction which is less likely to create large prediction errors is used. The way to mitigate the errors is to select the SVM model parameters according to the changes of TVCI values. This answers why the proposed Method 7 beats its counterparts by giving the smallest error values.

6.4 Applications

This section presents the applications of the proposed algorithm in condition prognostics for the two pertinent experimental systems. For the slurry pump system,

two conditions are to be studied, one where the pump impeller has vane trailing edge damage and vane leading edge damage, respectively. For the planetary gearbox, we deal with the condition of the gearbox in the course of the RTF test. The proposed online prognostic algorithm uses Methods 4 and 7 to determine the parameters of the SVM/LSSVM model for prediction. Method 6 is also employed for comparison purposes, because it is the second best among the seven methods according to the results from comparing the simulation datasets.

The parameters of Methods 6 and 7 for the two experimental datasets are given in Table 6-7. The definitions of the parameters are exactly the same as those in Table 6-2. The settings for the parameters were also the same except that parameter N was set at 200 for the planetary gearbox dataset.

Table 6-7: Parameter settings for experiment datasets

Dataset \ Method	Methods 6			Method 7				
	N	*l*	*b*	*N*	*l*	*b*	*m*	*h*
Slurry pump dataset	100	5	50	100	30	50	0.5	3
Planetary gearbox dataset	200	5	50	200	30	50	0.5	3

6.4.1 Condition Prognostics for Slurry Pump System

6.4.1.1 Database Establishment

As mentioned in Section 4.2.2.1, 5-minute time span vibration data were collected for each combination of damage modes, damage levels, and pump speeds. The software for data acquisition automatically split and saved the data in 40 separate data files each of which is an 8-second time record. As a result, each file should contain 72000 (9000×8) data points; however, a close look at all data files suggests that some had fewer data points than 72000. For this reason, we selected 3 consecutive data segments from each data file. Each segment contained 18000 data points. This data length is sufficient to capture the lowest frequency component of interest (30 Hz) which corresponds to a pump speed of 1800 rpm. Thus, for a combination of pump speed and damage mode, we had 120 (3×40) data segments for a single damage level and 480 (4×120) data segments for all four damage levels. We connected these data segments from baseline to severe in order to construct a series of degradation

data. We built slurry pump datasets using vibration signals from A1-z direction, collected at a pump speed of 2600 rpm. Kurtosis and the second harmonic of pump speed (2X) were computed and employed as CIs representing pump conditions. This section focuses only on the prediction performance. For the performance of CPU time, one can refer to the discussions of the two simulation datasets.

6.4.1.2 Results

(1) Trailing Edge Damage

Figure 6-14 shows the results of 2X. Basically, the observations show an increasing trend as time passes. Because the TVCIs of 2X are unknown, we cannot provide NRMSE values for quantitative comparison. This is true of all experimental datasets. We see that the trend of Method 4 is the noisiest of the three methods. Method 6 performs well, but it shows greater fluctuation than does Method 7.

Figure 6-15 shows the results of kurtosis. The observations are evidently subject to a decreasing trend as time passes. The trend of Method 4 is still slightly noisier than the trends of the others, especially at the beginning. Methods 6 and 7 perform comparably. It is seen that observations of kurtosis dropped suddenly between times 224 and 228. Method 7 spent 16 (228 – 244) time intervals capturing this change and triggered the re-determination of LSSVM parameters at time 244. As a result, the predicted kurtosis value becomes 0.1228 at time 244 which is much closer to the TVCI via visual estimation, compared to the 0.1783 of Method 6.

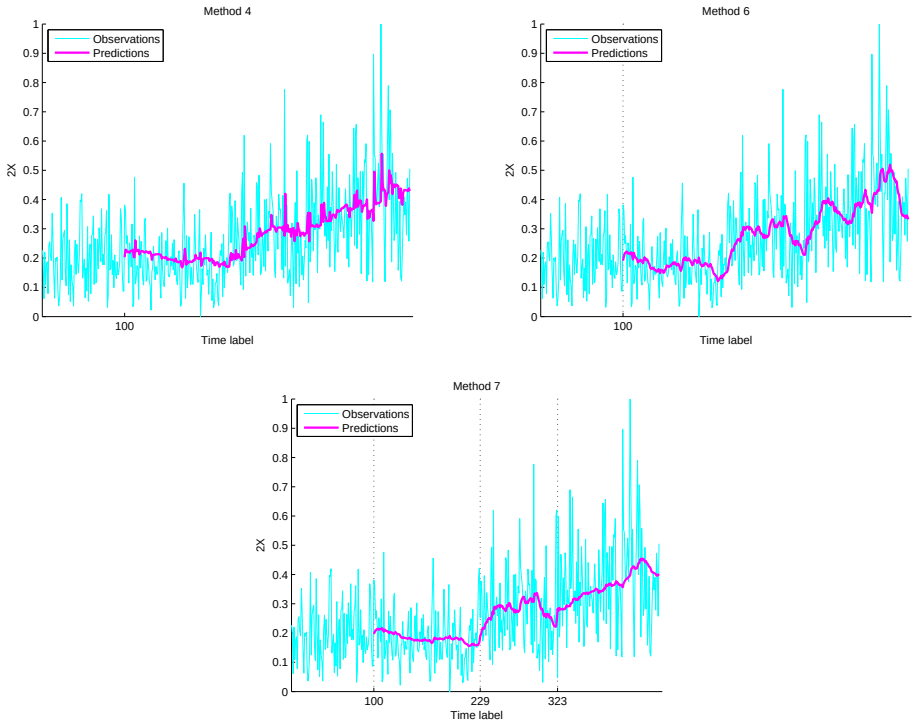

Figure 6-14: Trend prediction of 2X for trailing edge damage

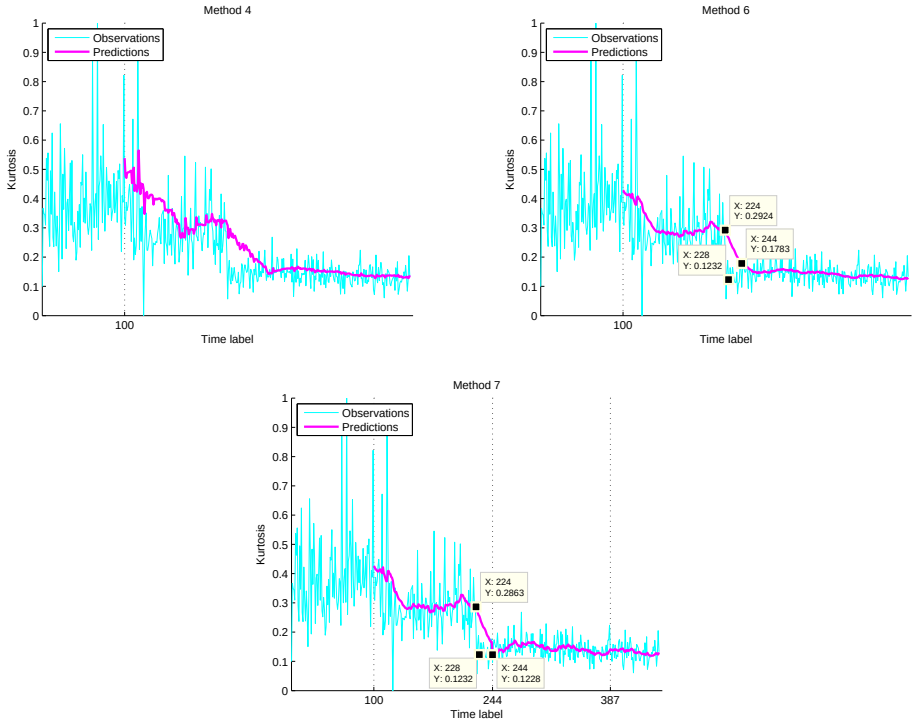

Figure 6-15: Trend prediction of kurtosis for trailing edge damage

For trailing edge damage, 2X shows an increasing trend and kurtosis shows a decreasing trend. Basically, they are good condition indicators as their values change monotonically with the passage of time. In terms of 2X, its values are in the same level for baseline and slight damages. When it comes to medium and severe damages, the increasing trend becomes obvious. In terms of kurtosis, its values for baseline damage are around three (The values shown in Figure 6-15 are normalized). For trailing edge damage, it is found that the large area of material loss on trailing edge causes the reduction of kurtosis values.

(2) Leading Edge Damage

Figure 6-16 shows the results of 2X. Basically, the observations of 2X do not show a degradation trend. The magnitude of the observations changes significantly and that of noise also varies noticeably. This is not a good trend for representing system condition; however, we found an interesting property of Method 7 through it. Method 7 predicted two sudden peaks appearing at times 359 and 439. This is a good property because peaks may carry useful information one does not want to miss regarding machine conditions. A possible explanation for the peaks is that a fresh set of LSSVM parameters is determined at time 353 which is coincidently in the middle of a process where observation values increase, so the parameters determined are able to capture the rapid increase in observation values. Nevertheless, Method 7 may not always successfully predict a peak, because CUSUM governs when to trigger the determination of LSSVM parameters. This leaves us with a problem for future work.

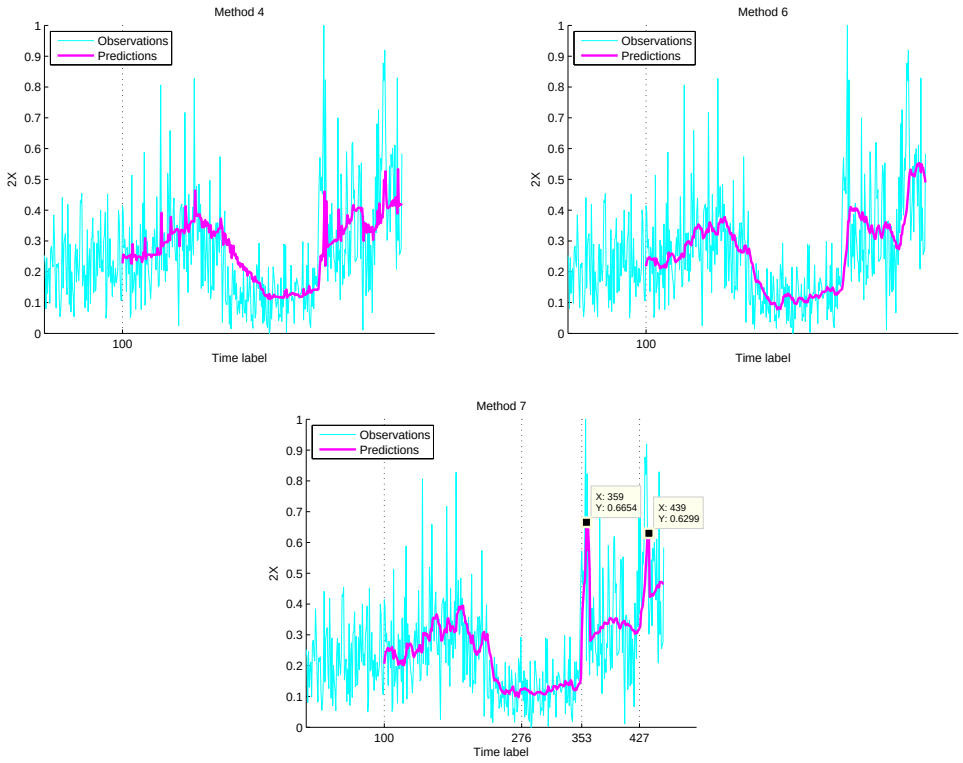

Figure 6-16: Trend prediction of 2X for leading edge damage

Figure 6-17 shows the results of kurtosis. The trend displays a sudden drop between times 229 and 232. The predicted trend of Method 4 is a little noisier than those of Methods 6 and 7, which are comparable. These observations are similar to those from the case of trailing edge damage; however, Method 7 performed somewhat interestingly for this case; it re-determined LSSVM parameters at time 224 which is actually in advance of the sudden drop. As a result, Method 7 basically captured this drop and provided a desirable trend for this time span. A possible explanation for this phenomenon is that the trend of observations declines generally from the starting at time point 100, but the LSSVM model is established upon the first 100 data points of a trend which is otherwise increasing, so deviations of predictions from unknown TVCIs accumulated to the point of exceeding the threshold. This caused the LSSVM parameters to be re-determined at time 224.

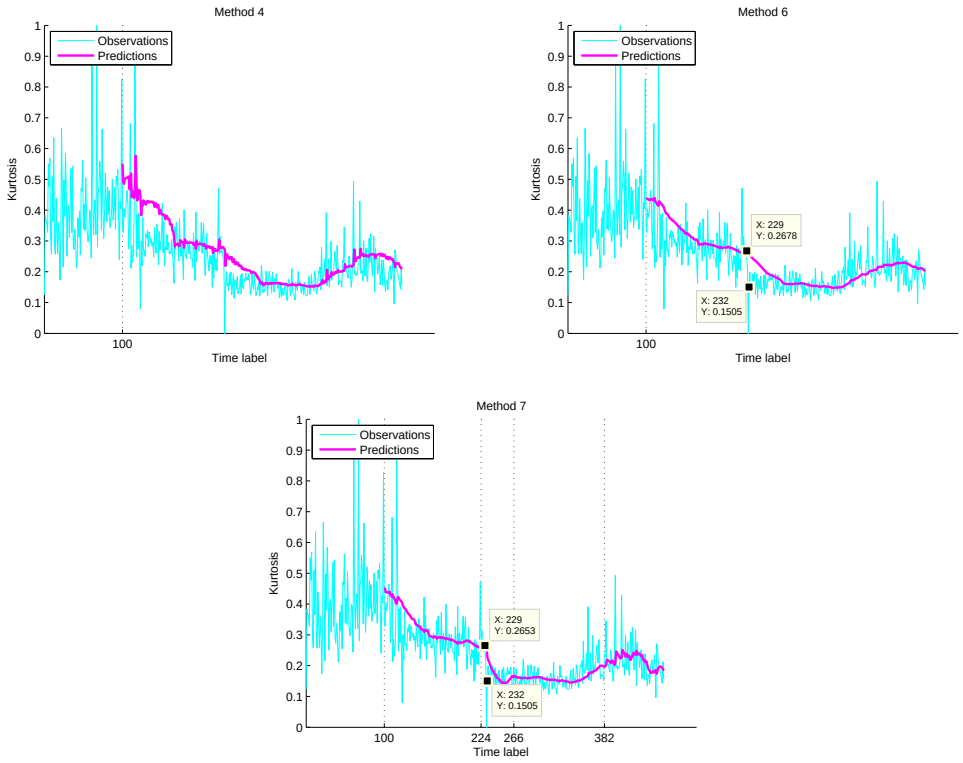

Figure 6-17: Trend prediction of kurtosis for leading edge damage

For leading edge damage, 2X is not as good as that for trailing edge damage. The 2X values of medium damage is smaller than those of baseline and slight damages. This makes 2X inappropriate to be the condition indicator for monitoring leading edge damage. The kurtosis basically performs consistently as does for the trailing edge damage. It shows a deceasing trend for baseline, slight and medium damage levels as expected, but a small fluctuation appears for the severe damage level around time 382. This is caused by some unknown factors that need further studies.

6.4.2 Condition Prognostics for Planetary Gearbox System

6.4.2.1 Database Establishment

Vibration signals from the run-to-failure (RTF) experiment are used. The 5-minute time span data were collected every two hours for each of 19 runs. The time span data were further split into 10 segments of equal length. All data segments were connected to construct a series of degradation data. We used the first order sideband

of the 2nd stage planetary gearbox to represent conditions at this stage.

6.4.2.2 Results

Figure 6-18 shows the results of the sideband CI. The trend of Method 6 seems to be noisier than that of Method 4 for this dataset. This is different from the slurry pump dataset. In contrast, Method 7 provides a much smoother trend and successfully captures the peaks. It is observed that Method 7 triggers 14 times re-determination of the LSSVM parameters; this reveals the complexity of this dataset.

As mentioned in Section 4, the RTF experimental data are continuously collected from natural progression of gear damage, so the data should be able to show a pattern of degradation. The CI used for tracking damage progression is the first order sideband whose magnitude increases as gear damage progresses. According to Figure 6-18, it is seen that the trend is flat from initial stage to time 1485 (run 12). This accords with our observation that obvious pitting damage does not appear on gear teeth until run 11. There are two unexpected big drops at times 701 and 830. They may be caused by inappropriate disassembly and assembly for gear damage inspection. After run 11, pitting damage grows faster and faster and the size of pits becomes larger, so we expect an increase on the trend values. The increase happens around time 1631, but a sudden drop is observed after that. This action forms a peak that is beyond our expectation. Again, it may be the consequence of disassembly for run 13. A consistent increase is finally observed after time 1995. This reflects the fast deterioration of gear until the experiment is terminated when more than 60% materials are worn off from gear teeth. Basically, the trend reflects the degradation of the gearbox, but it is not as good as our expectation. Further studies should be done for developing CIs that are more sensitive to the degradation.

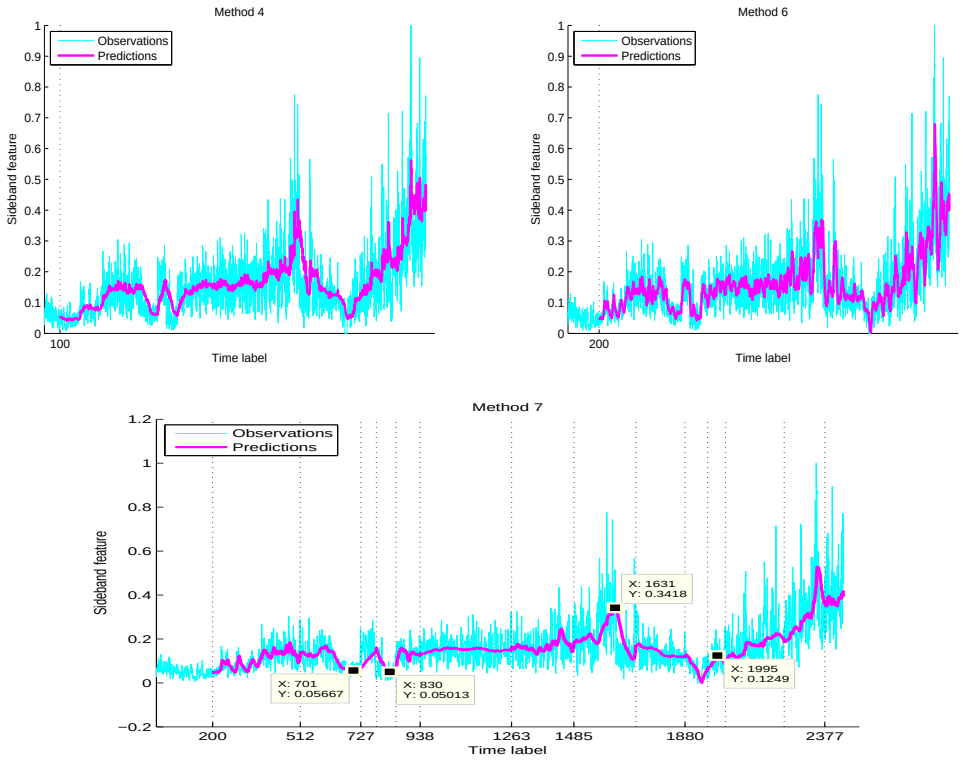

Figure 6-18: Trend prediction of sideband for planetary gearbox

6.5 Summary

This chapter presents an online prognostic algorithm where SVM and LSSVM are employed for the stage of prediction. One analytical method and one optimization-based method are proposed for selecting SVM/LSSVM model parameters based on which noise effects in observations can be mitigated from predictions. The proposed analytical method is for selecting the regularization parameter, C, of SVM. It incorporates a measure of relative noise level into a reported analytical method to address noise effects in TVCI predictions. The proposed optimization-based method determines the parameters using an optimization model developed to exclude noise-like components from its predictions. The cumulative sum technique is used to intelligently trigger the optimization process for re-determining the parameters.

The two proposed methods were incorporated into the online prognostic algorithm and their performance was demonstrated using simulation datasets. They were then compared with three reported analytical methods and two reported optimization-

based methods. Also the online prognostic algorithm was used with the two proposed methods in condition prognostics of the slurry pump and the planetary gearbox. The results show that in terms of prediction accuracy, the proposed two methods both perform better than their counterparts in their category. In terms of CPU time, the proposed analytical method performs comparable to its counterpart. The proposed optimization-based method usually needs more CPU time to reach the results. In general, the optimization-based methods performed better than the analytical methods with respect to both prediction errors and trend patterns, but consume more CPU time than analytical methods.

Chapter 7

Conclusions and Future Work

Diagnostics and prognostics are effective and efficient techniques for identifying system operating conditions and preventing unexpected system failures. They are an integrated process covering many aspects. Each aspect is uniquely essential and should be taken care of to ensure the successful application of diagnostics and prognostics. My research focuses on one particular aspect of diagnostics and prognostics, respectively. This chapter summarizes my contributions to these aspects, describes some problems that remain unaddressed, and suggests directions that hold potential for future work.

7.1 Conclusions

(1) SVM-Based Diagnostics

Diagnostics includes several constituent stages. Data cleaning and feature selection are two of them, responsible for providing reliable processed data. In practice, raw data collected from monitoring instruments may be corrupted by noise resulting from various random factors occurring in the course of data collection. Usually, the raw data are in a form that can not be used directly for diagnostics. Sometimes, the dimensions of data are too large to be efficiently used. All these concerns affect the results of diagnostics. In this thesis, we propose an SVM-based diagnostic algorithm that is able to deal with such concerns.

SVM as a basis is employed in the stages of data cleaning, feature selection, and classification. SVM has good generalization ability, as has been acknowledged

by many researchers. Its superior performance for small sample cases has also attracted a lot of attention. Another advantage of using SVM in multiple stages of diagnostics is reducing duplicated steps in determining classifier model parameters which requires much training and testing, resulting in significant computational expense.

The proposed SVM-based data cleaning algorithm was inspired by the observation that outliers close to the separating plane are often misclassified by SVM classifiers. Random sub-sampling validation is able to capture these misclassified data points by implementing a training process a specified number of times. A fraction value for each data point that has been misclassified is then calculated, and candidate outliers are identified and ranked according to their fraction values. Finally, the backward selecting scheme is used to determine which are the real outliers, following the rule that the removal of a real outlier improves classification performance. The proposed data cleaning algorithm is examined using three benchmark datasets. The results show its good effectiveness. We also used it for a slurry pump system to clean data for impeller damage level classification. Our results show that one outlier exists in the slurry pump dataset, and eliminating that outlier increases classification accuracy by 3.17%.

The proposed SVM-based feature selection algorithm aims at removing irrelevant and redundant features and reducing the dimensions of feature space; this is because irrelevant features tend to impair classification performance and high dimension of feature space consumes a huge amount of computational time. According to the SVM theorem, the margin of SVM classification is related to the separation between two classes of data points. This margin value can be directly represented by a weight vector. The proposed feature selection algorithm uses this margin value to rank features in terms of their importance to classification. A recursive backward selecting scheme was developed to select the most useful features. It outperforms the forward selecting scheme in terms of retaining useful features. Unlike regular backward selecting schemes, the recursive one eliminates useless features repeatedly until no more can be removed. This allows the maximum possible reduction of feature space dimension. The proposed feature selection algorithm was examined using three benchmark datasets, and the results are very promising. To study

its performance in practical applications, the proposed feature selection algorithm was used for classifying the level of pitting damage in a planetary gearbox. Various conditions of speed and load were studied. It is found that, on average, classification accuracy is increased by more than 30%, and the dimension of feature space is reduced from 136 to about 5. These results show the great potential of the proposed feature selection algorithm for engineering applications.

The proposed SVM-based diagnostic algorithm which employs both the proposed SVM-based data cleaning and feature selection algorithm was used for the slurry pump datasets. Our results show that the classification accuracy is increased by 3.11% when using data cleaning only. The classification accuracy remains unchanged but the dimension of feature space is reduced by 5 when using feature selection only. When using both, the classification accuracy is increased by 5.45%, and the dimension of feature space is reduced by 11. These results show the effectiveness of the proposed SVM-based diagnostic algorithm and indicate the importance of combining data cleaning and feature selection for diagnostics.

In a word, the contributions of this thesis to diagnostics can be summarized as:

1. Develops an SVM-based data cleaning algorithm to detect and remove outliers for effective diagnostics. This algorithm specially cleans the data for classification purpose that is required by diagnostics, but has not been studied in the literature.

2. Develops an SVM-based feature selection algorithm to remove redundant and irrelevant features for effective diagnostics. This algorithm uses a special measure of SVM to rank features and uses the recursive backward selecting scheme to remove as many as possible the useless features.

3. Develops an SVM-based algorithm that employs the SVM-based data cleaning, feature selection, and classification for effective diagnostics.

(2) SVM-Based Prognostics

Prognostics also contains several stages where prediction is essential to provide reliable predictions of true values for the condition indicator based on which decisions

regarding preventive activities can be made. SVM as a predictor is one of the good choices because of its good generalization ability. We propose an on-line prognostic algorithm using SVM for the prediction stage. It is realistic to expect that noise will affect the observed condition indicator values. Directly using such values for prediction could result in unreliable predictions because noise affects the predictions to some extent. One can relieve such noise effects by selecting appropriate SVM model parameters. In the literature, two methods are usually used, the analytical and optimization-based methods. We propose one method for each of the two categories. We have used the proposed methods to determine SVM model parameters for the on-line prognostic algorithm in order to predict a smooth trend of the true values for the condition indicator.

The proposed analytical method is for selecting a regularization parameter, C. It is inferred from reported studies that parameter C should be proportional to the noise level; however, this does not adequately to express the noise effects, since they should be evaluated in terms of the magnitude of the observations. For this reason, we propose a measure of relative noise level and incorporate it into a reported analytical model. The proposed method is validated using two simulation datasets. Three reported analytical methods are compared with the proposed one, and the results show that the proposed method is better than its counterparts in terms of not only prediction errors but also trend patterns. The proposed method is used in an on-line prognostic algorithm for a slurry pump and planetary gearbox.

The proposed optimization-based method aims at selecting a set of optimal parameters for the SVM model. We use LSSVM instead of regular SVM due to its superior prediction performance for data with Gaussian noise. The proposed method adopts a new optimization model developed for determining parameters. The genetic algorithm is employed to address the optimization problem. Due to possible variations in condition indicator values, a single set of optimal parameters may not work for the whole process of prediction. The cumulative sum technique is thus adopted to detect any error accumulated between observations and predictions. When accumulated error exceeds a specified threshold, the cumulative sum technique will trigger another optimization process to obtain a new set of parameters based on up-to-date observations. This proposed method is demonstrated using the

same simulation datasets as before. Two optimization-based methods are compared which determine model parameters for SVM and LSSVM based on cross-validation methods. The results show that the proposed method is superior to its counterparts in terms of prediction accuracy and trend patterns, but one drawback is that it needs longer time to reach the results. The proposed method is applied in the on-line prognostic algorithm for the slurry pump and planetary gearbox. The results show good potential of the proposed method in practical applications.

It is observed that optimization-based methods provide high prediction accuracy and good trend patterns than do analytical methods. This makes sense since optimization-based methods determine parameters based on the observation values for a given problem and is therefore case-specific; in contrast, analytical methods merely use statistical measures of observations which are not specific to any given problem. Nevertheless, the optimization-based methods usually need longer CPU time due to the optimization process embedded.

Briefly, the contributions of this thesis to prognostics can be summarized as:

1. Develops an analytical method to select SVM model parameters. This method modifies a reported method by considering noise effects in observations. Based on our tests, it provides better results for on-line system condition prognostics than does the reported one.

2. Develops an intelligent optimization-based method to select SVM model parameters. This method proposes an optimization model that considers noise effects in observations and uses the cumulative sum technique to control the process of determining SVM parameters. Based on our tests, it outperforms the traditional optimization-based and analytical methods and provides the best results for the tested cases of on-line system condition prognostics.

3. Develops an SVM-based algorithm that employs respectively the developed analytical and optimization-based methods for on-line system condition prognostics.

7.2 Future Work

(1) Classification of Fault Mode and Fault Level

One ultimate goal of this thesis has been to conduct a successful fault level classification, but fault mode classification, which is another important aspect of diagnostics, has not been studied. For desired diagnostic results, one may need to know not only what type of fault a system is currently experiencing, but also how severe the fault is. This requires doing both fault mode classification and fault level classification in a single diagnostic task. Unfortunately, we have not found reported studies that address this concern. For this reason, an SVM-based diagnostic algorithm that is able to classify both fault mode and fault level is developed. The algorithm will contain an off-line process and an on-line process, where the former achieves training SVM using available data, and the latter achieves identifying fault mode and fault level on the basis of new condition monitoring data. The difficulties of this work will be:

1. how to assess the overall fault level of a system if multiple fault modes exist;
2. how to identify or determine unknown fault modes, since it is impossible to have training data for all possible fault modes;
3. how to deal with the fault level of unknown fault modes.

(2) Parameter Selection for the Proposed Optimization-Based Prognostics

This thesis presents an optimization-based method of determining LSSVM model parameters. This method uses the cumulative sum technique to govern when to re-determine a model's parameters. There are two parameters, m and h, which specify deviations to be detected and accumulated errors not to be tolerated. In the tests given in Chapter 6, these two parameters are set constantly; however, they may not be the best choice. Based on our studies, where there are small RNL values, small m values tend to give better results, and vice versa. These observations are also true for parameter h. In order to obtain better predictions, we will develop criteria or algorithms for selecting optimal values for m and h.

(3) Threshold Determination for Prognostics

This thesis specifically addresses the noise effects on the prediction of condition indicators for prognostics; however, we do not present a case study that illustrates deciding when to stop an operating system for an overhaul. This is because we need to pre-specify a condition indicator threshold that represents a failure condition, and the value for this threshold usually relies on solid knowledge of and expertise with the system that is not easy to obtain. This topic will be worked on in future.

Bibliography

[1] http://www.syncrude.ca/users/folder.asp?Folder ID=5703, 2010.

[2] J. Chadwick. Oil sand in the sand pit. *International Mining*, pages 25–30, 2005.

[3] H.Z. Wang. A survey of maintenance policies of deteriorating systems. *European Journal of Operational Research*, 139:469–486, 2002.

[4] R.K. Mobley. *Maintenance Fundamentals*. Elsevier Butterworth-Heinemann, Burlington, MA01803, USA, 2004.

[5] A. Heng, S. Zhang, A.C.C. Tan, and J. Mathew. A rotating machinery prognostics: state of the art, challenges and opportunities. *Mechanical Systems and Signal Processing*, 23:724–739, 2009.

[6] L. Zhang, X. S. Li, and J. S. Yu. A review of fault prognostics in condition based maintenance and opportunities. *The 6th International Symposium on Instrumentation and Control Technology: Signal Analysis, Measurement Theory, Photo-Electronic Technology, and Artificial Intelligence*, 6357(52):1–6, 2006.

[7] A.K.S. Jardine, D. Lin, and D. Banjevic. A review on machinery diagnostics and prognostics implementing condition-based maintenance. *Mechanical Systems and Signal Processing*, 20:1483–1510, 2006.

[8] A.H. Christer, W. Wang, and J.M. Sharp. A state space condition monitoring model for furnace erosion prediction and replacement. *European Journal of Operational Research*, 101:1–14, 1997.

[9] S. Lu, Y.C. Tu, and H.T. Lu. Predictive condition-based maintenance for continuously deteriorating systems. *Quality and Reliability Engineering International*, 23:71–81, 2007.

[10] V. Makis and A.K.S. Jardine. Optimal replacement in the proportional hazards model. *Information Systems and Operational Research*, 30(1):172–183, 1992.

[11] A. Widodo and B.S. Yang. Support vector machine in machine condition monitoring and fault diagnosis. *Mechanical Systems and Signal Processing*, 21:2560–2574, 2007.

[12] R. Kothamasu, S.H. Huang, and W.H. Verduin. System health monitoring and prognostics - a review of current paradigms and practices. *International Journal of Advance Manufacturing Technology*, 28:1012–1024, 2006.

[13] P.D. Samuel, D.J. Pines, and D.G. Lewicki. Comparison of stationary and non-stationary metrics for detecting faults in helicopter gearboxes. *Journal of the American Helicopter Society*, 45:125–136, 2000.

[14] R.Y. Fujimoto S. Vicente and L.R. Padovese. Rolling bearing fault diagnostic system using fuzzy logic. *IEEE International Fuzzy System Conference*, 2001.

[15] C.K. Mechefske. Objective machinery fault diagnosis using fuzzy logic. *Mechanical Systems and Signal Processing*, 12(6):855–862, 1998.

[16] J.D. Wu and C.C. Hsu. Fault gear identification using vibration signal with discrete wavelet transform technique and fuzzy-logic inference. *Expert systems with applications*, 36:3785–3794, 2009.

[17] T. Brotherton, G. Jahns, J. Jacobs, and D. Woblewski. Prognosis of faults in gas turbine engines. *IEEE Aerospace Conference*, 6:163–171, Big Sky, MT, March 2000.

[18] K.L. Butler. An expert system based framework for an incipient failure detection and predictive maintenance system, intelligent systems applications to power systems. *International Conference of ISAP' 96*, pages 321–326, 1996.

[19] B. Samanta and C. Nataraj. Prognostics of machine condition using soft computing. *Robotics and Computer-Integrated Manufacturing*, 24:816–823, 2008.

[20] B. Samanta. Gear fault detection using artificial neural networks and support vector machines with genetic algorithms. *Mechanical Systems and Signal Processing*, 18:625–644, 2004.

[21] F.J. He and W.G. Shi. WPT-SVMs based approach fault detection of valves in reciprocating pumps. *Proceedings of American control Conference*, Anchorage, AK, May 2002.

[22] S.F. Yuan and F.L. Chu. Support vector machines-based fault diagnosis for turbo-pump rotor. *Mechanical Systems and Signal Processing*, 20:939–952, 2006.

[23] S.F. Yuan and F.L. Chu. Fault diagnosis based on support vector machines with parameter optimization by artificial immunization algorithm. *Mechanical Systems and Signal Processing*, 21:1318–1330, 2007.

[24] K.Y. Chen. Forecasting systems reliability based on support vector regression with genetic algorithms. *Reliability Engineering and System Safety*, 92:423–432, 2007.

[25] P.F. Pai. System reliability forecasting by support vector machines with genetic algorithms. *Mathematical and Computer Modeling*, 43:262–274, 2006.

[26] H.E. Kim, A.C.C. Tan, J. Mathew, E.Y.H. Kim, and B.K. Choi. Machine prognostics based on health state estimation using SVM. *Proceedings Third World Congress on Engineering Asset Management and Intelligent Maintenance Systems Conference*, 199:834–845, Beijing, China, 2008.

[27] W.C. Hong and P.F. Pai. Predicting engine reliability by support vector machines. *International Journal of Advance Manufacturing Technology*, 28: 154–161, 2006.

[28] O. Gualdrón, J. Brezmes, E. Llobet, A. Amari, X. Vilanova, B. Bouchikhi, and X. Correig. Variable selection for support vector machine based multi-sensor system. *Sensors and Actuators*, B 122:259–268, 2007.

[29] S. Balakrishnan, R. Narayanaswamy, N. Savarimuthu, and R. Samikannu. SVM ranking with backward search for feature selection in type II diabetes databases. *IEEE International Conference on Systems, Man and Cybernetics*, 1-6:2632–2637, 2008.

[30] V. Cherkassky and Y. Ma. Practical selection of SVM parameters and noise estimation for SVM regression. *Neural Computation*, 17:113–126, 2004.

[31] J.T. Kowk and I.W. Tsang. Linear dependency between p and the input noise in ϵ - support vector regression. *IEEE transactions on neural networks*, 14(3): 544–553, 2003.

[32] E.M. Jordaan and G.F. Smits. Estimation of the regularization parameter for support vector regression. *Proceedings of the International Joint Conference on Neural Networks*, pages 2192–2197, 2002.

[33] P. Večeř, M. Kreidl, and R. Smíd. Condition indicators for gearbox condition monitoring systems. *Acta Polytechnica*, 45(6):35–43, 2005.

[34] M.J. Zuo, W. Li, and X.F. Fan. Statistical methods for low speed planetary gearbox monitoring. *Technical Report*, Department of Mechanical Engineering, University of Alberta, Edmonton, 2005.

[35] P.D. Samuel, D.J. Pines, and D.G. Lewicki. A comparison of stationary and non-stationary metrics for detecting faults in helicopter gearboxes. *Journal of the American Helicopter Society*, pages 125–136, 2000.

[36] S. Pöyhöne, P. Jover, and H. Hyotyniemi. Signal processing of vibration for condition monitoring of an induction motor. *ISCCSP: 2004 First International Symposium on Control, Communication and Signal Processing*, pages 499–502, New York, 2004.

[37] P.W. Tse and D.P. Atherton. Prediction of machine deterioration using vibration based fault trends and recurrent neural networks. *Journal of Vibration and Acoustics*, 121:355–362, 1999.

[38] P.K. Young. *Wavelet theory and its applications*. Kluwer academic publishers, Boston, 1993.

[39] N. Saravanan and K.I. Ramachandran. Fault diagnosis of spur bevel gear box using discrete wavelet features and decision tree classification. *Expert Systems with Applications*, 36:9564–9573, 2009.

[40] N. Saravanan and K.I. Ramachandran. Incipient gear box fault diagnosis using discrete wavelet transform (DWT) for feature extraction and classification using artificial neural network (ANN). *Expert Systems with Applications*, 37: 4168–4181, 2010.

[41] X.Y. Wang, V. Makis, and M. Yang. A wavelet approach to fault diagnosis of a gearbox under varying load conditions. *Journal of Sound and Vibration*, 329:1570–1585, 2010.

[42] Q.W. Wang, M.F. Golnaraghi, and F. Ismail. Prognosis of machine health condition using neuro-fuzzy systems. *Mechanical Systems and Signal Processing*, 18:813–831, 2004.

[43] F.E. Grubbs. Procedures for detecting outlying observations in samples. *Technometrics*, 11:1–21, 1969.

[44] P.J. Rousseeuw and B.C.V. Zomeren. Unmasking multivariate outliers and leverage points. *Journal of the American Statistical Association*, 85(411):633–639, 1990.

[45] W.K. Fung. Outlier diagnostics in several multivariate samples. the statistician. *Technometrics*, 48(1):73–84, 1999.

[46] A.D. Shieh and Y.S. Hung. Detecting outlier samples in microarray data. *Statistical Applications in Genetics and Molecular Biology*, 8(1):Article 13, 2009.

[47] N.S. Chaudari, A. Tiwari, and J. Thomas. A novel SVM based approach for noisy data elimination. *The 11th International Conference on Control Automation Robotics & Vision (ICARCV)*, pages 7–10, Singapore, Dec. 2010.

[48] K.L. Li and G.F. Teng. Unsupervised SVM based on p-kernels for anomaly detection. *ICICIC 2006: First International Conference on Innovative Computing, Information and Control*, 2:59–62, 2006.

[49] I. Guyon, J. Weston, and S. Barnhill. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46:389–422, 2002.

[50] B. Li, P.L. Zhang, S.B. Liang, and G.Q. Ren. Feature extraction and selection for fault diagnosis of gear using wavelet entropy and mutual information. *The 9th International Conference on Signal Processing*, 1-5:2843–2847, Singapore, October 26-29,2008.

[51] N. Baydar, Q. Chen, A. Ball, and U. Kruger. Detection of incipient tooth defect in helical gears using multivariate statistics. *Mechanical systems and signal processing*, 15:303–321, 2001.

[52] G.H. John, R. Kohavi, and K. Pfleger. Irrelevant features and the subset selection problem. *Proceedings of the Eleven International Conference of Machine Learning*, pages 121–129, San Francisco, CA,1994.

[53] J. Qu and M.J. Zuo. Support vector machine based data processing algorithm for wear degree classification of slurry pump systems. *Measurement*, 43:781–791, 2010.

[54] J. Qu, C.X. Miao, M. Hoseini, and Ming J. Zuo. Wear degree prognostics for slurry pumps using support vector machines. *The Proceedings of 8th International Conference on Reliability,Maintainability, and Safety*, pages 940–943, Chendu, China, July 20-24, 2009.

[55] V.A. Skormin, L.J. Popyack, V.I. Gorodetski, M.L. Araiza, and J.D. Michel. Approaches for reliability modeling of continuous-state devices. *IEEE Transactions on Reliability*, 48(1):9–18, 1999.

[56] P.K. Dash and S.R. Samantaray. An accurate fault classification algorithm using a minimal radial basis function neural network. *Engineering Intelligent Systems for Electrical Engineering and Communications*, 12(4):205–210, 2004.

[57] R.K.Y. Chang, C.K. Loo, and M.V.C. Rao. Enhanced probabilistic neural network with data imputation capabilities for machine-fault classification. *Neural Computing & Applications*, 18(7):791–800, 2009.

[58] J.D. Wu and C.H. Liu. Investigation of engine fault diagnosis using discrete wavelet transform and neural network. *Expert Systems with Applications*, 35: 1200–1213, 2008.

[59] J. Rafiee, P.W. Tse, A. Harifi, and M.H. Sadeghi. A novel technique for selecting mother wavelet function using an intelligent fault diagnosis system. *Expert Systems with Applications*, 36(3):4862–4875, 2009.

[60] J.D. Wu and J.M Kuo. An automotive generator fault diagnosis system using discrete wavelet transform and artificial neural network. *Expert Systems with Applications*, 36(6):9776–9783, 2009.

[61] J.S.R. Jang. ANFIS: Adaptive-network-based fuzzy inference system. *IEEE Transactions on Systems, Man, and Cybernetics*, 23(3):665–685, 1993.

[62] V.T. Tran and B.S. Yang. Machine fault diagnosis and condition prognosis using classification and regression trees and neuro-fuzzy inference systems. *Control and Cybernetics*, 39(1):25–54, 2010.

[63] F.V. Nelwamondo and T. Marwala. Early classification of bearing faults using hidden markov models, gaussian mixture models, mel-frequency cepstral coefficients and fractals. *International Journal of Innovative computing, Information and Control*, 2(6):1281–1299, 2006.

[64] Z.N. Li, Z.T. Wu, Y.Y. He, and C.F. Lei. Hidden markov model-based fault diagnostics method in speed-up and speed-down process for rotating machinery. *Mechanical Systems and Signal Processing*, 19(2):329–339, 2005.

[65] J.H. Yan, M. Koc, and J. Lee. A prognostic algorithm for machine performance assessment and its application. *Production Planning & Control*, 15(8):796–801, 2004.

[66] E. Sutanto, K. Warwic, and M. Griffin. Applications of logistic regression for fault analysis in an industrial printing process. *Instrumentation and Measurement Technology Conference*, pages 675–680, Metropolitan, NY, USA, May 12, 1992.

[67] T.M. Mitchell. *Machine Learning*. The McGraw-Hill Companies, Inc., 1997.

[68] M. Maragoudakis, E. Loukis, and P.P. Pantelides. Gas turbine fault diagnosis using random forests. *18th European Conference on Artificial Intelligence*, 178:769–770, Patras, Greece, July 21-25, 2008.

[69] D. Michie, D.J. Spiegelhalter, and C.C. Taylor. *Machine Learning, Neural and Statistical Classification*. 1998.

[70] J.D. Wu, C.C. Hsu, and G.Z. Wu. Fault gear identification and classification using discrete wavelet transform and adaptive neuro-fuzzy inference. *Expert Systems with Applications*, 36(3):6244–6255, Part 2, 2009.

[71] T. Nguyen and Y. Liao. Transmission line fault type classification based on novel features and neuro-fuzzy system. *Electric Power Components and Systems*, 38(6):695–709, 2010.

[72] W. Kuo and M.J. Zuo. *Optimal Reliability Modeling*. John Wiley & Sons, Inc., New Jersey, 2003.

[73] M.J. Zuo, R.Y. Jiang, and R.C.M. Yam. Applications of cluster analysis in diagnostics-related problems. *Proceedings of 1999 IEEE Aerospace Conference*, 3:161–168, Snowmass at Aspen, CO, USA, 1999.

[74] H. Tanizaki. *State-space model in linear case, Chapter 1 of Nonlinear Filters: Estimation and Application*. Springer-Verlag, 1996.

[75] Y.F. Zhou, L. Ma, R.C. Wolff, and H.E. Kim. Asset life prediction using multiple degradation indicators and lifetime data: a gamma based state space

model approach. *The 8th International Conference on Reliability, Maintainability and Safety*, Chengdu, China, July 20-24, 2009.

[76] D. Banjevic, A.K.S. Jardine, V. Makis, and M. Ennis. A control-limit policy and software for condition-based maintenance optimization. *Information Systems and Operational Research*, 39:32–50, 2001.

[77] X.D. Zhang, R. Xu, C. Kwan, S.Y. Liang, Q.L. Xie, and L. Haynes. An integrated approach to bearing fault diagnostics and prognostics. *American Control Conference*, Portland, OR, USA, 2005.

[78] F.L. Greitzer, L.J. Knagas, and K.M. Terrones. Gas turbine engine health monitoring and prognostics. *International Society of Logistics Symposium*, Las Vegas, Nevada, 1999.

[79] C.J.C. Burges. *A Tutorial on Support Vector Machines for Pattern Recognition*. Kluwer Academic Publishers, Boston, 1994.

[80] N. Cristianini and J.S. Taylor. *An introduction to support vector machines and other kernel-based learning methods*. Cambridge, 2000.

[81] C.G. Fei, Z.Z. Han, and Q.K Liu. Ultrasonic flaw classification of seafloor petroleum transporting pipeline based on chaotic genetic algorithm and SVM. *Journal of X-Ray Science and Technology*, 14:1–9, 2006.

[82] A. Smola B. Sch´okopf. *Learning with kernels: support vector machines, regularization, and beyond*. Cambridge, MA: MIT Press, 2002.

[83] A. Chalimourda, B. Schölkopf, and A.J. Smola. Experimentally optimal υ in support vector regression for different noise models and parameter settings. *Neural Networks*, 17:127–141, 2004.

[84] D. Mattera and S. Haykin. *Support vector machines for dynamic reconstruction of a chaotic system, in B. Schölkopf and A. Smola (Eds.), Advances in Kernel Methods: Support Vector Machine*. Cambridge, 1999.

[85] X.J. Zhou, Y.G. Lei, D. Wolfe, and M.J. Zuo. Gear stress calculation of the planetary gearbox system (chinese standard). Technical report, Department of Mechanical Engineering, University of Alberta, Edmonton, Alberta, 2008.

[86] M. Hoseini and M.J. Zuo. A literature survey on the creating and quantifying faults on planetary gearbox. Technical report, Department of Mechanical Engineering, University of Alberta, Edmonton, Alberta, 2009.

[87] M.J. Zuo, S.Y. Wu, Z.P. Feng, A. Aulakh, and C.X. Miao. Condition monitoring of slurry pumps for wear assessment. Technical report, Department of Mechanical Engineering, University of Alberta, March 30, 2007.

[88] A.S. Aulakh and S.Y. Wu. Slurry pump CBM project. Technical report, Syncrude Canada Ltd, August 21, 2006.

[89] J. Qu, Z.L. Liu, M.J. Zuo, and H.Z. Huang. Feature selection for damage degree classification of planetary gearboxes using support vector machine. *Proceedings of the Institution of Mechanical Engineers, Part C, Journal of Mechanical Engineering Science*, 225(C9):2250–2264, 2011.

[90] J. Qu, Z.L. Liu, and M. J. Zuo. Damage degree classifications for planetary gearboxes using information fusion and support vector machine. *2010 MITACS and CORS Annual Conference*, 2010.

[91] http://en.wikipedia.org/wiki/Confusion-matrix.

[92] A. Asuncion and D.J. Newman. *UCI Machine Learning Repository [http://www.ics.uci. edu/ mlearn/MLRepository.html].* Irvine, CA: University of California, School of Information and Computer Science.

[93] U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine. Broad patterns of gene expression revealed by clustering of tumor and normal colon tissues probed by oligonucleotide arrays. *Proceedings of the National Academy of Science USA*, 96:6745–6750, 1999.

[94] S.R. Gunn. Support vector machines for classification and regression. Technical report, Department of electrical and computer science, University of Southampton, 1998.

[95] L. Li, T.A. Darden, C.R. Weinberg, A.J. Levine, and L.G. Pedersen. Gene assessment and sample classification for gene expression data using a genetic

algorithm/k-nearest neighbor method. *Chemistry & High Throughput Screening*, 4:727–739, 2001.

[96] H.Y. Li and M. Niranjan. Outlier detection in benchmark classification tasks. *The 2006 IEEE International Conference on Acoustics Speech and Signal Processing*, 5:14–19, 2006.

[97] M. Lebold, K. McClintic, R. Campbell, C. Byington, and K. Maynard. Review of vibration analysis methods for gearbox diagnostics and prognostics. *Proceedings of the 54th Meeting of the Society for Machinery Failure Prevention Technology*, pages 623–634, Virginia Beach, VA, May 1-4, 2000.

[98] G. White. *Introduction to machine vibration*. Bainbridge Island, DLI Engr. Corp., 1995.

[99] J. K. Sinha and A. R. Rao. Vibration based diagnosis of a centrifugal pump. *Structural Health Monitoring*, 5(4):325–332, 2006.

[100] M. A. Langthjem and N. Olhoff. A numerical study of flow-induced noise in a two-dimensional centrifugal pump. *Journal of Fluids and Structures*, 19(3): 369–386, 2004.

[101] Y.G. Lei and M.J. Zuo. Gear crack level identification based on weighted k nearest neighbor classification algorithm. *Mechanical Systems and Signal Processing*, 23(5):1535–1547, 2009.

[102] M. Inalpolat and A. Kahraman. A theoretical and experimental investigation of modulation sidebands of planetary gear sets. *Journal of Sound and Vibration*, 323:667–696, 2009.

[103] H.J. Decker. Crack detection for aerospace quality spur gears. Technical report, NASA/TM-2002-211492, ARL-TR-2682.

[104] F.K. Choy, S. Huang, J.J. Zakrajsek, R.F. Handschuh, and D.P. Townsend. Vibration signature analysis of a faulted gear transmission system. *AD-A290195; ARL-TR-475; E-8914; NAS 1.15:106623; NASA-TM-106623; Joint Propulsion Conference, Indianapolis, IN, United States*, pages 27–29, 1994.

[105] J.J. Zakrajsek, D.P. Townsend, and H.J. Decker. An analysis of gear fault detection methods as applied to pitting fatigue failure data. *Technical Report*, NASA TM-105950, AVSCOM TR-92-C-035, NASA and the US Army Aviation Systems Command,January 1993.

[106] H.J. Decker and D.G. Lewicki. Spiral bevel pinion crack detection in a helicopter gearbox. *Proceedings of the American Helicopter Society 59th Annual Forum*, pages 1222–1232, Phoenix, AZ, 2003.

[107] H.R. Martin. Statistical moment analysis as a means of surface damage detection. *Proceedings of the Seventh International Modal Analysis Conference, Society for Experimental Mechanics*, pages 1016–1021, Schenectady, NY, 1989.

[108] P.D. Samuel and D.J. Pines. A review of vibration-based techniques for helicopter transmission diagnostics. *Journal of Sound and Vibration*, pages 475–508, 2005.

[109] H.J. Decker, R.F. Handschuh, and J.J. Zakrajsek. An enhancement to the na4 gear vibration diagnostic parameter. *Technical Report*, NASA TM-106553, ARL-TR-389, NASA and the US Army Research Laboratory, July 1994.

[110] J. Qu, M.J. Zuo, and H.Z. Huang. Selection of regularization parameter for support vector regression. *Neural processing letters*, submitted, 2010.

[111] J. Qu and M.J. Zuo. SVM-based prognosis of machine health condition. *Proceedings of the 2010 International Conference on Mechanical, Industrial, and Manufacturing Technologies*, pages 337–341, Sanya, China, January 22-24, 2010.

[112] J. Qu and M.J. Zuo. An LSSVR-based machine condition prognostics algorithm for slurry pump systems. *Proceedings of the Canadian Society for Mechanical Engineering Forum 2010, CSME FORUM 2010*, Victoria, British Columbia, Canada, June 7-9, 2010.

[113] J. Qu and M.J. Zuo. An LSSVR-based algorithm for online system condition prognostics. *Expert Systems with Applications*, Submitted, 2011.

[114] T.P. Ryan. *Statistical Methods for Quality Improvement.* John Wiley & Sons, 1989.

[115] J.A.K. Suykens, J.D. Brabanter, L. Lukas, and J. Vandewalle. Weighted least squares support vector machines: robustness and sparse approximation. *Neurocomputing*, 48:85–105, 2002.