An Intelligent System for Induction Motor Health Condition Monitoring

by

Dezhi Li

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Mechanical Engineering

Waterloo, Ontario, Canada, 2015

AUTHOR'S DECLARATION

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Induction motors (IMs) are commonly used in both industrial applications and household appliances. An IM online condition monitoring system is very useful to identify the IM fault at its initial stage, in order to prevent machinery malfunction, decreased productivity and even catastrophic failures. Although a series of research efforts have been conducted over decades for IM fault diagnosis using various approaches, it still remains a challenging task to accurately diagnose the IM fault due to the complex signal transmission path and environmental noise. The objective of this thesis is to develop a novel intelligent system for more reliable IM health condition monitoring. The developed intelligent monitor consists of two stages: feature extraction and decision-making. In feature extraction, a spectrum synch technique is proposed to extract representative features from collected stator current signals for fault detection in IM systems. The local bands related to IM health conditions are synchronized to enhance fault characteristic features; a central kurtosis method is suggested to extract representative information from the resulting spectrum and to formulate an index for fault diagnosis. In diagnostic pattern classification, an innovative selective boosting technique is proposed to effectively classify representative features into different IM health condition categories. On the other hand, IM health conditions can also be predicted by applying appropriate prognostic schemes. In system state forecasting, two forecasting techniques, a model-based pBoost predictor and a data-driven evolving fuzzy neural predictor, are proposed to forecast future states of the fault indices, which can be employed to further improve the accuracy of IM health condition monitoring. A novel fuzzy inference system is developed to integrate information from both the classifier and the predictor for IM health condition monitoring. The effectiveness of the proposed techniques and integrated monitor is verified through simulations and experimental tests corresponding to different IM states such as IMs with broken rotor bars and with the bearing outer race defect.

The developed techniques, the selective boosting classifier, pBoost predictor and evolving fuzzy neural predictor, are effective tools that can be employed in a much wider range of applications. In order to select the most reliable technique in each processing module so as to provide a more positive assessment of IM health conditions, some more techniques are also proposed for each processing purpose. A conjugate Levebnerg-Marquardt method and a Laplace particle swarm technique are proposed for model parameter training, whereas a mutated particle filter technique is developed for

system state prediction. These strong tools developed in this work could also be applied to fault diagnosis and other applications.

Acknowledgements

I would like to express my greatest gratitude to my supervisors, Professor Wilson Wang and Professor Fathy Ismail, for their enthusiastic supervision and patient guidance. I cannot imagine having any better advisors for my doctoral study regarding their knowledge, perceptiveness, and encouragement.

I am grateful to my PhD committee members, external examiner Professor Chris Mechefske, and Professors Fakhri Karray, Jan P. Huissoon, and William W. Melek, for their instructive guidance and help.

Financial support from the Ontario Graduate Scholarship (OGS), University of Waterloo, and my supervisors, is greatly appreciated.

Many thanks to my friends: Ran Peng, Ning Qin and many others for their friendship and help.

I am indebted, forever, to my beloved wife, Ling Zhang and my parents, for their endless support and encouragement.

Dedication

To my family

Table of Contents

List of Figures

List of Tables

List of Abbreviations

AB	AdaBoost
AC	Alternating Current
AR	Autoregressive
ARMA	Autoregressive Moving Average
CEC	Cumulative Evolving Clustering
CG	Conjugate Gradient
DC	Direct Current
DENFIS	Dynamic Evolving Neuro-Fuzzy Inference System
eFNN	Evolving Fuzzy Neural Network
eNF	Evolving Neuro-Fuzzy
ECM	Evolving Clustering Method
EF	Evolving Fuzzy
EFFNN	Enhanced Fuzzy Filtered Neural Network
EFuNN	Evolving Fuzzy Neural Network
FFT	Fast Fourier Transform
FIS	Fault Information Spectrum
FT	Fourier Transform
GD	Gradient Descent
IFD	Integrated Fuzzy Diagnostic
IM	Induction Motors
LM	Levenberg-Marquardt
MAE	Mean Absolute Error
MF	Membership Function
ND	Noise Detection Based AdaBoost Algorithm
NF	Neuro-Fuzzy
NN	Neural Network
PSD	Power Spectral Density
RMS	Root Mean Square
RMSE	Root Mean Square Error

RLSE	Recursive Least Square Estimate
SNR	Signal-To-Noise Ratio
SR	Successful Classification Rate
SS	Spectrum Synch
STFT	Short Time Fourier Transform
VARMA	Vector Autoregressive Moving Average
WAB	Weight Decay AdaBoost
WLSE	Weighted Least Square Estimate
WPD	Wavelet Packet Decomposition
WT	Wavelet Transform

Chapter 1
Introduction

1.1 Overview

Electric motors are machines that convert electrical energy into mechanical energy. They can be classified into three categories: direct current (DC) motors, alternating current (AC) motors, and special motors. There are two types of AC motors: induction motors (IMs) and synchronous motors. IMs employ electromagnetic induction to provide torque for driven machines.

IMs are the workhorse of many industries such as manufacturing, and mining; and more importantly, they consume up to 50% of the generated electrical energy in the world [1]. Due to these facts, a series of R&D activities have been directed, for decades, to improve the performance and efficiency of IMs. For example, in industrial applications, an effective and reliable condition monitoring system is valuable in the detection of an IM fault at its earliest stage in order to prevent performance reduction and malfunction of the driven machinery. It could also be utilized to schedule predictive maintenance operations. Maintenance costs can be further reduced (especially for large expensive motors) by quickly identifying the faulty component(s) without inspecting all components in the IM.

Figure 1.1 shows an example of IM structure, consisting of a stator, a rotor, a shaft, rolling element bearings, and the related supplementary components. IM components could be damaged during operations due to impacts, fatigue, insufficient lubrication, aging, and so on. The IM faults can be classified as mechanical defects and electrical faults. Mechanical faults consist of bearing faults, shaft unbalance and so on. Electrical faults contain broken rotor bars, stator winding defects, phase imperfection and so on. Investigations have revealed that bearing faults are the major IM defects, which account for approximately 75% of small-sized and medium-sized motor defects and 41% of large motor imperfections in domestic and industrial applications [2]. The broken rotor bars account for 10% of IM faults.

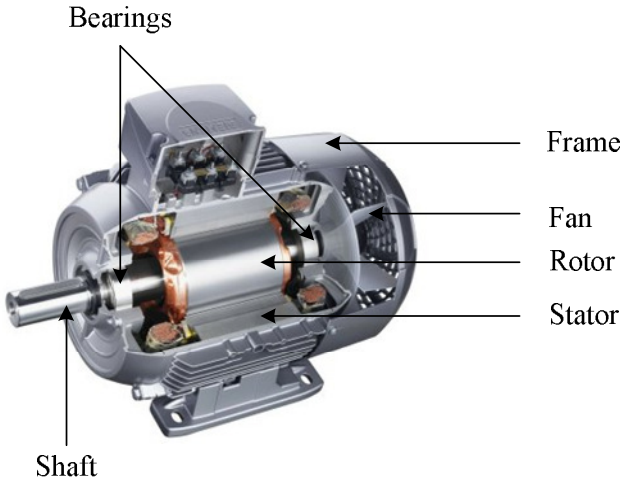

Figure 1.1. View of an induction motor, reproduced from [3].

The traditional IM fault diagnostic method, which is still widely practiced by maintenance crews in industry, relies on human diagnosticians for periodic inspections based on warning signs such as smell, temperature increase, excessive vibration, and increased acoustic noise level. It is difficult, however, for a diagnostician to handle multiple features from different signal processing techniques and explore their inner relationship especially when the signals are contaminated with noise. The alternative is for automated (online) IM condition monitoring with the use of computered-aided diagnosis expert systems, which could quickly predict IM health conditions from various features. IM online condition monitoring consists of several modules, as illustrated in Figure 1.2. The function of each module is described as follows:

a) Data acquisition: is to measure IM signals using appropriate sensors, pre-process the data and send them to a computer for further processing.

b) Feature extraction: is to extract representative features from collected signals by the use of appropriate fault detection techniques.

c) Classification: is to categorize the extracted representative features from several fault detection techniques for automatic fault diagnosis of IMs.

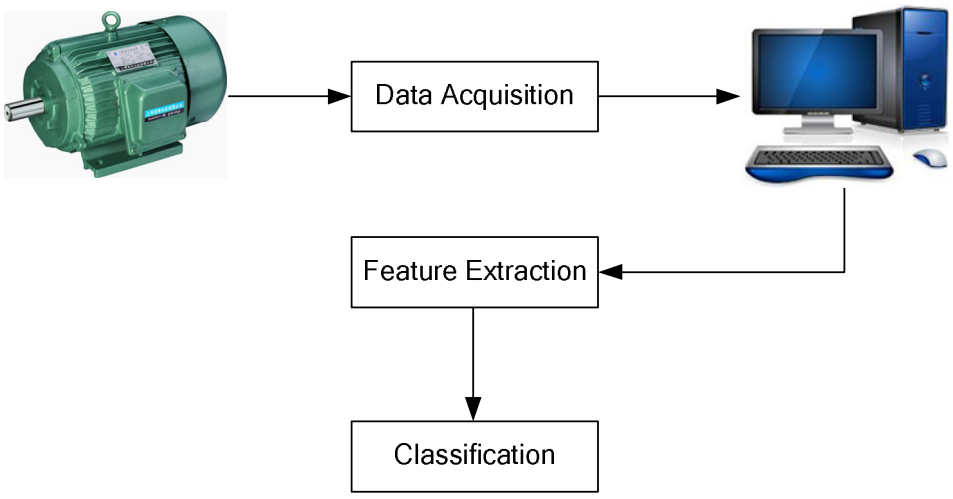

Figure 1.2. A flow chart of IM online condition monitoring system.

1.2 Literature Review

The proposed IM intelligent monitoring system involves processes (modules) of fault detection, pattern classification and system state prediction, which will be elaborated on in Subsection 1.3. Thus technologies in each of these three modules are reviewed in this subsection. Through this literature review, the related work on each module is summarized and discussed. Furthermore, the motivation of the proposed research in this thesis will be explicitly explained.

1.2.1 IM Fault Detection

Fault detection is a process to extract representative features from the collected signals. Traditional machinery fault detection is based on thermal signals [4], acoustic signals [5], and vibration signals [6-9]. The local or bulk temperature can be used to diagnose IM defects, however the heat accumulation and progression are slow, which may not be suitable for incipient fault detection. The acoustic noise can indicate IM faults, especially for severe and distributed defects; however the acoustic signal is prone to contamination by background noise such as noise from other machines in the vicinity. The acquisition of vibration signals requires the related vibration sensors mounted in the vicinity of the IM components to be monitored. Although vibration signals have relatively high signal-to-noise ratio, the pieasoelectric accerometers are expensive and their installation requirements are high; in addition, it is difficult to use vibration signals to detect IM electrical defects such as

broken rotor bars. The alternative is to use the stator current signal for analysis, which is non-invasive to the IM structure [10], and could also be used to detect IM electrical faults. Thus the proposed research in this work will focus on IM fault diagnosis using stator current signals.

Several motor current signature analysis techniques have been proposed in the literature for fault detection in IMs, mainly for rotors and bearings, which are briefly summarized next:

1) Fault detection of IM rotors

Broken rotor bars are common rotor defects that will render asymmetries of an IM rotor. The rotor bar failures can be caused by several factors [11-12], such as:

(a) overheating due to frequent starts under loading,

(b) unbalanced thermal load due to air gap eccentricity,

(c) manufacturing defects, and

(d) corrosion of rotor material caused by chemicals or moisture.

Because of the aforementioned reasons, the rotor bar(s) may be fully or partially damaged, which will cause the rotor cage asymmetry and result in asymmetrical distribution of the rotor currents. When a crack forms in a rotor bar, the cracked bar will overheat and tend to break. Then the adjacent bars have to carry higher currents and consequently they become prone to damage, leading to multiple bar fractures [12]. Moreover, the broken parts from the faulty bars may hit the end winding of the motor and cause serious mechanical damage to the IM [13].

The Fast Fourier Transform (FFT) spectral analysis is a commonly used method for rotor bar breakage detection, by examining the characteristic frequency components in the spectrum. For example, Elkasabgy *et al.* [14] used spectral analysis of IM current signals to detect rotor bar breakage. It has been reported that the IM current signal becomes non-stationary if rotor bars are damaged. However, the FFT is useful for stationary signal analysis only, which lacks the capability of capturing the transitory characteristics such as drifts, abrupt changes and frequency trends in non-stationary signals. To solve the problem, time-frequency methods, such as short time Fourier transform (STFT), can be used to process small segments of non-stationary signals for broken rotor bar defect detection. For example, Arabaci *et al.* [15] applied the STFT to detect IM rotor bar faults. However, the STFT cannot provide the information corresponding to different time resolutions and frequency resolutions due to its fixed length windowing functions [16]. To solve this problem, the wavelet transform (WT) can be employed to explore the information associated with different

frequency resolutions. For example, Daviu *et al.* [17] and Mehala *et al.* [21] used discrete WT to detect IM broken rotor bars fault. The wavelet packet decomposition (WPD) was used to explore the whole frequency range with high resolution. For example, Sadeghian *et al.* [18] used WPD to extract features and applied neural networks to diagnose IM rotor bar breakage. Pineda-Sanchez *et al.* [19] employed polynomial-phase transform to diagnose broken rotor bar fault in time-varying condition. Riera-Guasp *et al.* [20] extracted broken rotor bar fault features from transient state of IM using Gabor analysis. Although the WPD can explore details of the signal for some advanced signal analysis, it is usually difficult to recognize the real representative features from the map with redundant and misleading information. Akin *et al.* [22] performed real-time fault detection using the reference frame theory. Soualhi *et al.* [23] diagnosed broken rotor bar fault through the classification of selected fault features using the improved artificial ant clustering method. Günal *et al.* [123] conducted IM broken rotor bar fault diagnosis by using fault indices in the time domain. Nevertheless, the aforementioned techniques only focus on limited fault information, thus their performance may be degraded.

2) Fault detection of IM bearings

Rolling element bearings are commonly used in rotating mechanical/electromechanical equipment. As illustrated in Figure 1.3, a rolling element bearing is a system consisting of an outer ring (usually the fixed ring), an inner ring (usually the rotating ring), a number of rolling elements, and a cage.

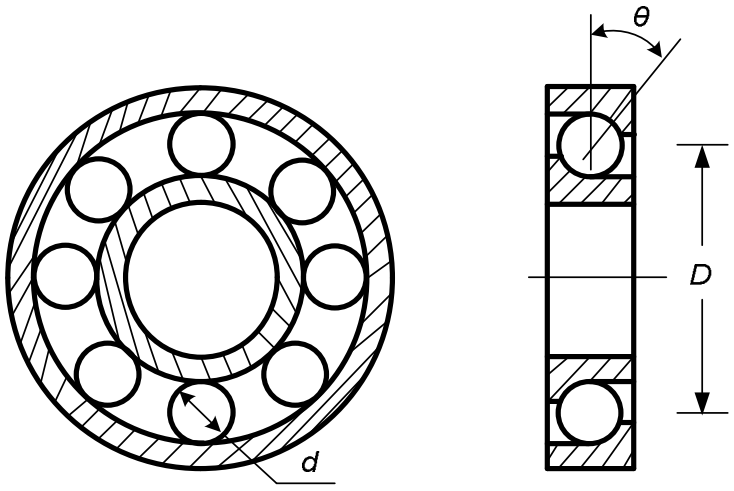

Figure 1.3. The geometry of a rolling element bearing. d = diameter of the rolling element; θ = angle of contact; D = pitch diameter.

Since bearing materials are subjected to dynamic loading, fatigue pitting is the most common defect in bearing components. The bearing defects can occur on the outer race, inner race, and rolling elements. Under normal operating conditions, after the load cycles exceed some threshold, fatigue (usually pitting) may occur on the fixed ring race first, and then on the rotating race and rolling elements. Pitting defects do not only deteriorate transmission accuracy, but also generate excessive vibration and noise. Other bearing defects, such as scoring and severe wear, can be generated by several external causes such as [11]:

(a) Impacts, overloading and overheating

(b) Inadequate lubrication;

(c) Contamination and corrosion from abrasive particles or acid;

(d) Improper installation of a bearing, which will introduce excessive misalignment errors.

When a bearing component is damaged, the corresponding characteristic vibration frequencies depend on the bearing geometry, rotation speed and defect location, as illustrated in Figure 1.3. Suppose the outer race of a bearing is fixed and the inner race rotates with the shaft, which is common in most applications. The outer race defect characteristic frequency f_{obr}, inner race defect characteristic frequency f_{ibr}, cage defect characteristic frequency f_{cbr}, and rolling element defect characteristic frequency f_{rbr} are given by

$$f_{obr} = \frac{N}{2}\left[f_i \left(1 - \frac{d\cos(\theta)}{D} \right) \right] \tag{1.1}$$

$$f_{ibr} = \frac{N}{2}\left[f_i \left(1 + \frac{d\cos(\theta)}{D} \right) \right] \tag{1.2}$$

$$f_{cbr} = \frac{1}{2}\left[f_i \left(1 - \frac{d\cos(\theta)}{D} \right) \right] \tag{1.3}$$

$$f_{rbr} = \frac{D}{2d} f_i \left[1 - \left(\frac{d\cos\theta}{D} \right)^2 \right] \tag{1.4}$$

where f_i is the inner race rotating speed or shaft speed; d is the ball diameter; D is the pitch diameter;

θ is the contact angle. Derivation of these rolling element bearing defect characteristic frequencies is provided in Appendix A.

When bearing defects occur, these bearing characteristic vibration frequencies f_v (i.e., f_{obr}, f_{ibr}, f_{cbr} and f_{rbr}) will be modulated with the power supply frequency f_p in the spectrum of stator current signals, because of the air gap eccentricity and load torque variations. Thus the characteristic stator current frequencies f_c in terms of characteristic vibration frequencies f_v will be calculated by

$$f_c = \left| f_p \pm m f_v \right|, \ m = 1, 2, 3, \ldots \tag{1.5}$$

The details of derivation of Equation (1.5) can found in [25].

For IM bearing fault detection, the characteristic stator current frequency components can be used as frequency domain indicators in spectrum analysis [26]. Several techniques have been proposed in the literature for IM bearing fault detection using stator current signals. For example, Benbouzid *et al.* [27] and Devaney *et al.* [28] applied IM stator current spectrum analysis for bearing defect detection. FFT can be used to conduct spectrum analysis, so as to detect IM bearing faults under deterministic motor conditions. Similar to the previous discussion regarding broken rotor bar analysis, the WT can be used to catch the transitory characteristics of the signal for IM bearing fault detection. For example, Konar *et al.* [29] employed discrete WT to detect IM bearing faults. The WPD can also be employed to explore transient fault information for IM bearing fault detection [30-32]. Nevertheless, the WPD generates massive non-fault-related information that may mask the fault features in the map, and increase the difficulties in fault detection. Frosini and Bassi [33] used features from stator current signals and IM efficiency for bearing fault detection. Zhou *et al.* [34] utilized the Wiener filter for noise reduction, so as to detect IM bearing defect. Romero-Troncoso *et al.* [35] conducted online IM fault detection using information entropy and fuzzy inference. Pineda-Sanchez *et al.* [36] employed Teager-Kaiser energy operator to enhance fault features to detect IM bearing defect. Nevertheless, these available techniques conduct IM bearing fault detection based on limited fault information rather than comprehensively explore fault features from the time domain, the frequency domain and the time-frequency domain simutaneouly. Therefore their performance may be degraded.

1.2.2 Automatic IM Fault Diagnostics

IM health condition diagnosis is a process to classify the representative features obtained by using appropriate signal processing techniques into different IM health condition categories. The classical automatic diagnostics is mainly performed by model-based approaches such as Bayes decision rule [37, 38], which minimizes the average possibility of errors by examining the posterior probabilities and the penalties associated with each decision. Nevertheless, the prior density functions of different classes are not known in most practical situations, therefore posterior probabilities cannot be properly determined [39]. Another well-accepted model-based classifier is the linear discriminant analysis [40, 41], which aims to minimize the variance in each group and maximize the variance between groups. But its application is limited to the measurements in which each group is normally distributed and all the groups have the same co-variance structure [42].

The alternative is the use of knowledge-based soft-computing tools such as neural networks (NNs) [43-45] and fuzzy systems for diagnostic pattern classification. The fuzzy systems conduct inference to mimic human IF-THEN reasoning rules, however the fuzzy model parameters are usually difficult to optimize. NNs can update the parameters using training algorithms, but their reasoning structures are opaque to users and the reasoning mechanism is difficult to interpret. The neuro-fuzzy (NF) systems employ the fuzzy inference and the NN-based training to improve the pattern classification performance. These knowledge-based intelligent tools are universal approximators [46] and, in theory, they can closely approximate any arbitrary function. Furthermore, they make no assumptions of prior density functions or distributions of measurements in each group. To optimize the parameters of the NNs and NF systems, several training algorithms have been proposed in the literature. For example, the classical gradient descent (GD) method searches optimal solutions in the steepest descent direction [47]. The weighted least square estimate (WLSE) minimizes the least square error with penalties associated with each input pattern [48, 49]. The conjugate gradient (CG) method employs a search direction at each iterative step that conjugates to all previous descent directions to improve training convergence [50, 51]. The Levenberg-Marquardt (LM) method adaptively optimizes the search direction between Newton's direction and steepest descent direction to enhance convergence [52, 53].

An evolving system is a data-driven scheme that can adaptively update its inference model structure and parameters based on the input data. Some evolving systems have been presented

recently to dynamically establish modeling structure based on characteristics of the input data [54, 55]. The model structure formation of evolving systems is dependent on the sequence in which the data are fed to the system [56].

AdaBoost, short for Adaptive Boosting, is a machine learning algorithm, which aims to combine the strength of many weak learners to form a strong learner. It updates the model structure by incorporating a new base learner into the ensemble at each step, which is insensitive to the sequence that data are input to the system. The classical AdaBoost has merits such as solid theoretical basis and ease of implementation [57]. Although the classical AdaBoost techniques have been used in many applications such as medical diagnoses [58, 59], object detection [60, 61] and network intrusion detection [62], they have some limitations such as being sensitive to noisy samples and the overfitting problem [63, 64].

Many research efforts have been made in literature to improve the AdaBoost [65-67]. For example, a Real AdaBoost algorithm was proposed in [57] to calculate the probability and confidence limits to estimate the degree to which a sample belongs to a specific class so as to speed up convergence. A LogitBoost method [68] employed an additive logistic regression model as a cost function to improve the convergence. These methods aim to correctly classify hard-to-learn samples at each step, which may mitigate the classification accuracy of easy-to-learn samples, and result in overfitting [63]. To reduce the overfitting problem, Ratsch *et al.* [69] proposed a boosting technique with soft margins to deal with noisy samples by computing the aggregated weights of each sample and reducing the penalties for hard-to-learn samples; however some hard-to-learn samples may be clean, but not be noise-affected, and therefore their penalty should not be reduced. Cao *et al.* [70] applied a noise detection method to identify noisy samples in the training data, and then assigned a different class label to each noise sample in the binary classification problem. Nevertheless, noise samples may not belong to a different class, and this approach may reduce classification accuracy. Gao *et al.* [71] applied a weighted *K*-nearest-neighbor algorithm in a transformed space to identify and discard suspicious noisy samples, but some of these discarded samples may be clean and useful for classification operations.

1.2.3 IM System State Prediction

IM health condition monitoring can also be performed from a different perspective; via system state prediction. System state prediction is a process to forecast future states of a system based on the

previous and current system states. Several prediction techniques have been applied in the literature for IM health condition monitoring. The classical system state prediction approaches are based mainly on stochastic models such as autoregressive (AR) model, autoregressive moving average (ARMA) model [72-74] and grey models [75,76]. These techniques have been employed in some applications for system state prediction. Nevertheless, the fixed structure of these techniques may limit the prediction accuracy for complex dynamic systems, like IMs, especially under variable machinery operating conditions.

An alternative is the use of knowledge-based soft-computing tools such as NNs [77-80], fuzzy systems [81-84], NF systems [85-91] and evolutionary computation [92-95]. However, as discussed in the previous subsection, a NN predictor has an opaque reasoning structure, which makes it difficult to explain the resulting prediction reasoning (i.e., knowledge extraction). NF schemes are superior to NNs in mimicking human reasoning processes and extracting knowledge as interpretable IF-THEN rules. Both NF systems and NNs, however, suffer from an inefficient training process and suboptimal network structures [96]. Evolutionary computation methods, such as genetic algorithm, particle swarm optimization, and simulated annealing, are usually time-consuming in reasoning and cannot be used for real time IM system state prognostic applications.

In system state prognosis, the prediction accuracy can be improved by using hybrid modeling: linear models (e.g., AR and ARMA) can be used to mainly characterize linear properties of data sets and nonlinear models (e.g., NNs and NF schemes) can be utilized to characterize nonlinear properties of data sets. For example, Medeiros *et al.* [97] proposed a neural coefficient smooth transition autoregressive model for system state prediction. Khashei *et al.* [98] integrated the NNs and ARMA models to conduct time series prediction. One remarkable merit of using the hybrid modeling strategy lies in its capacity for dealing with non-stationary data sets. The linear non-stationary components could be modeled by using a linear modeling method, and nonlinear components can be characterized by some nonlinear modeling method [99-102]. However, these available approaches cannot make their reasoning mechanism adapt to new system conditions in real time (online), and consequently their model structures may still be suboptimal.

In recent years, more work has focused on the use of evolving NF paradigms that can adaptively adjust their network structures in response to new system conditions (i.e., data sets). For example, Kasabov *et al.* proposed evolving fuzzy NN models (EFuNN) [103-105] and dynamic evolving NF

inference system (DENFIS) techniques [106] for different applications such as learning, knowledge acquisition, and system state prediction. These evolving paradigms employ clustering algorithms to adaptively tune model structure and parameters as new data become available. However, the established clusters (or rules) in these evolving systems are usually massive in order to capture all details of the data (i.e., both linear properties and nonlinear properties of the data), which increases the computational burden and reduces training efficiency, especially for real-time applications.

1.3 Objective and Strategy

To tackle the limitations of reviewed techniques in each aforementioned aspect (e.g., fault detection, pattern classification, and system state prediction), the main goal of this work is to develop new techniques and an intelligent system for IM health condition monitoring. The detailed objectives include:

a) Propose a new signal processing technique to extract representative features from the stator current signals for fault detection in IMs. This work will focus on broken rotor bar fault and bearing fault detection. The proposed technique should be robust; be sensitive to IM faults, while at the same time insensitive to environmental noise.

b) Propose a pattern classification technique to effectively categorize the representative features from the selected technique(s) to provide a more accurate assessment of IM health condition.

c) Develop a model-based predictor and a knowledge-based data-driven predictor to forecast future states of the IM conditions. The forecast information from one of them will be properly fed into fault diagnosis to further improve the reliability of IM condition monitoring.

d) Propose a new diagnostic scheme to effectively integrate the diagnostic and prognostic information for IM health condition monitoring, and to conduct automatic diagnostic decision-making.

e) A side objective is to develop other model parameter training techniques and system state estimation tools that could be employed for fault diagnosis. The purpose is to provide more options in each module so as to choose the best technique for each process in order to provide a more accurate assessment of IM health conditions.

The strategy of this research is to use the most robust and efficient techniques in each processing module in order to provide a more reliable IM condition monitoring system. With the assistance of

the condition monitoring information, preventative maintenance can be effectively scheduled to avoid unexpected equipment downtime, and to enhance productivity levels of the related machines. The proposed methods for IM fault detection, pattern classification and system state prediction, can also be applied to other related signal processing, pattern classification, and modeling applications.

1.4 Thesis Outline

In Chapter 2, a novel signal processing technique, the spectrum synch technique, is proposed to extract representative features from the stator current signals for IM fault detection. The proposed technique can synchronize fault-related local bands in the IM stator current spectrum, to generate a more reliable fault indicator. The effectiveness will be verified through experimental testing.

In Chapter 3, a new selective boosting classifier is proposed to categorize the IM health conditions. The proposed classifier can adaptively mitigate the penalties of hard-to-learn samples based on their level of noise, in order to reduce overfitting and improve the classification accuracy.

In Chapter 4, a model-based pBoost predictor is developed to conduct dynamic system state forecasting. It combines the strength of many AR predictors using boosting methods in order to improve prediction accuracy. Each AR predictor is employed as a base learner. A novel sample weight regulation method is proposed to reduce the overfitting problem. The effectiveness of the proposed pBoost predictor is examined by using simulations and real applications.

In Chapter 5, a data-driven fuzzy NN predictor is developed to capture both linear and nonlinear characteristics of data sets using different modeling strategies. An innovative cumulative clustering algorithm is proposed to establish fuzzy reasoning rules for nonlinear modeling.

The classification and prediction techniques proposed in Chapters 3-5 were evaluated using benchmark data from the literature and subjected to rigorous testing methods that are common in the fields of data classification and forecasting. These proposed techniques can also be used for general classification and prediction applications.

In Chapter 6, a new fuzzy inference system is developed to integrate information from both the selective boosting classifier and the pBoost predictor, in order to conduct a more accurate fault diagnosis in IMs.

Concluding remarks and ideas for future research are given in Chapter 7.

Other related techniques for model parameter training and system state estimation, which are developed to achieve the objective of (e) in Subsection 1.3, are listed in Appendices B-D.

Chapter 2
A Spectrum Synch Technique for IM Fault Detection

2.1 Problem Definition

Typically, the onset of IM faults begins with small imperfections, and advances to a severe stage as the operation progresses. The severe IM faults will cause machinery malfunction, and even catastrophic failures. Therefore, the detection of IM faults at their earliest stage is of great importance in IM condition monitoring. The IM fault features from stator current signals would be associated with fault size, motor type, supply frequency, load condition and so on. To date, fault feature extraction from IM current signals, especially associated with bearing defects, still remains a challenging task due to the complex transmission path from the damaged bearing vibration symptom to the stator current signal. In this chapter, a spectrum synch technique is proposed to gather fault-related information and generate representative features of IM faults, such as broken rotor bar fault and outer race defect in a bearing [107].

2.2 Theory of Spectrum Synch Analysis

The proposed spectrum synch (SS) technique is composed of two procedures: local band synch and central kurtosis analysis. The local band synch is used to form the fault information spectrum (FIS) and accentuate fault features. The central kurtosis is suggested to generate fault indices for IM health condition monitoring.

2.2.1 Local Band Synch

The IM fault characteristic frequency components are distributed over the spectrum, which, however, are usually difficult to recognize due to their low amplitude. To highlight fault features in the spectrum, the FIS is used to enhance the local peakedness of the fault frequency components. Firstly, to lessen the noise effect in the IM current signal, the spectrum averaging of J data sets ϕ_j, $j = 1, 2, \ldots, J$, is applied to improve the signal-to-noise ratio (SNR), computed by

$$\Phi = \frac{1}{J}\sum_{j=1}^{J} \log\left(P(\phi_j)\right) \tag{2.1}$$

where Φ is the averaged spectrum over J spectra; $P(\phi_j)$ represents the nonparametric power spectral density (PSD) estimate of the data set [108], given by

$$P(\phi_j) = \frac{2}{f_s N} \sum_{i=1}^{N/2+1} \left| F_j(i) \right|^2 \tag{2.2}$$

where F_j is the spectrum of ϕ_j using the Fourier transform (FT); N is the length of ϕ_j; and f_s is the sampling frequency.

The fault features are related to fault characteristic frequencies, most of which are masked over the local bands by some other higher level frequency components considered as noise. To tackle this problem, some representative local bands containing the fault characteristic frequencies are selected and synchronized to reduce the noise effect and protrude the fault frequency components. In each selected local band, the fault frequency component f_c is located in the center of the window, and the width of the local band is selected to properly reveal the peakedness of f_c.

To synchronize the corresponding bands at different locations (frequencies) of the spectrum, the spectrum is transformed from frequency domain $\Phi(f)$ to discrete point representation $\Phi(d)$. Each frequency f can be represented by its nearest discrete point d. Then, fault characteristic frequency $f_c(k)$ is transformed into a discrete point, $d_c(k)$, whose corresponding frequency is the one closest to f_c, where k = 1, 2, …, K, and K is the total number of fault characteristic frequencies considered. Thus K local bands will be used for this synch operation. The widths of local bands are identical in this work to facilitate the synch operation. Given the bandwidth in frequency f_w, the length of the local band in discrete point representation, d_w, will be

$$d_w = 2R\left\langle \frac{1}{2} f_w \frac{D_s}{f_s} \right\rangle \tag{2.3}$$

where f_s is the sampling frequency in Hz, D_s is the discrete point representing f_s, and $R\langle \bullet \rangle$ represents round-off operation. The k^{th} local band ψ_k in the discrete point representation can be determined by

$$\psi_k = \{\Phi(i)\}_{i=d_c(k)-\frac{1}{2}d_w,\ldots,d_c(k)+\frac{1}{2}d_w} - \frac{1}{d_w+1}\sum_{i=d_c(k)-\frac{1}{2}d_w}^{d_c(k)+\frac{1}{2}d_w}\Phi(i) \tag{2.4}$$

The i^{th} discrete point in the k^{th} local band ψ_k is denoted as $\psi_{i,k}$, i = 1, 2, …, d_w+1; k = 1, 2, …, K. The i^{th} discrete points over K local bands $\{\psi_{i,k}\}$ are sorted in a descending order in terms of their values to generate $\pi_{i,k}$, k = 1, 2, …, K; the synchronized band FIS will be

$$g_i = \begin{cases} \dfrac{2}{(K-1)}\displaystyle\sum_{j=1}^{(K-1)/2}\pi_{i,j} & K \text{ is odd} \\ \dfrac{2}{K}\displaystyle\sum_{j=1}^{K/2}\pi_{i,j} & K \text{ is even} \end{cases}, \quad i = \frac{1}{2}d_w + 1 \tag{2.5}$$

$$g_i = \xi\{\pi_{i,j}\}_{j=1,2,\ldots,K}\ ,\ i = 1, 2, \ldots, \frac{1}{2}d_w,\ \frac{1}{2}d_w + 2, \ldots, d_w + 1 \tag{2.6}$$

where $\xi\{\bullet\}$ represents the computation of median value. The top 50% high amplitude center frequency components in local bands are averaged in Equation (2.5) to enhance the fault feature. The median value calculation in Equation (2.6) will suppress other frequency components in local bands and reduce the amplitude of outliers. The processing procedures of the proposed FIS formation are illustrated in Figure 2.1, where the frequency resolution Δf = 0.5Hz.

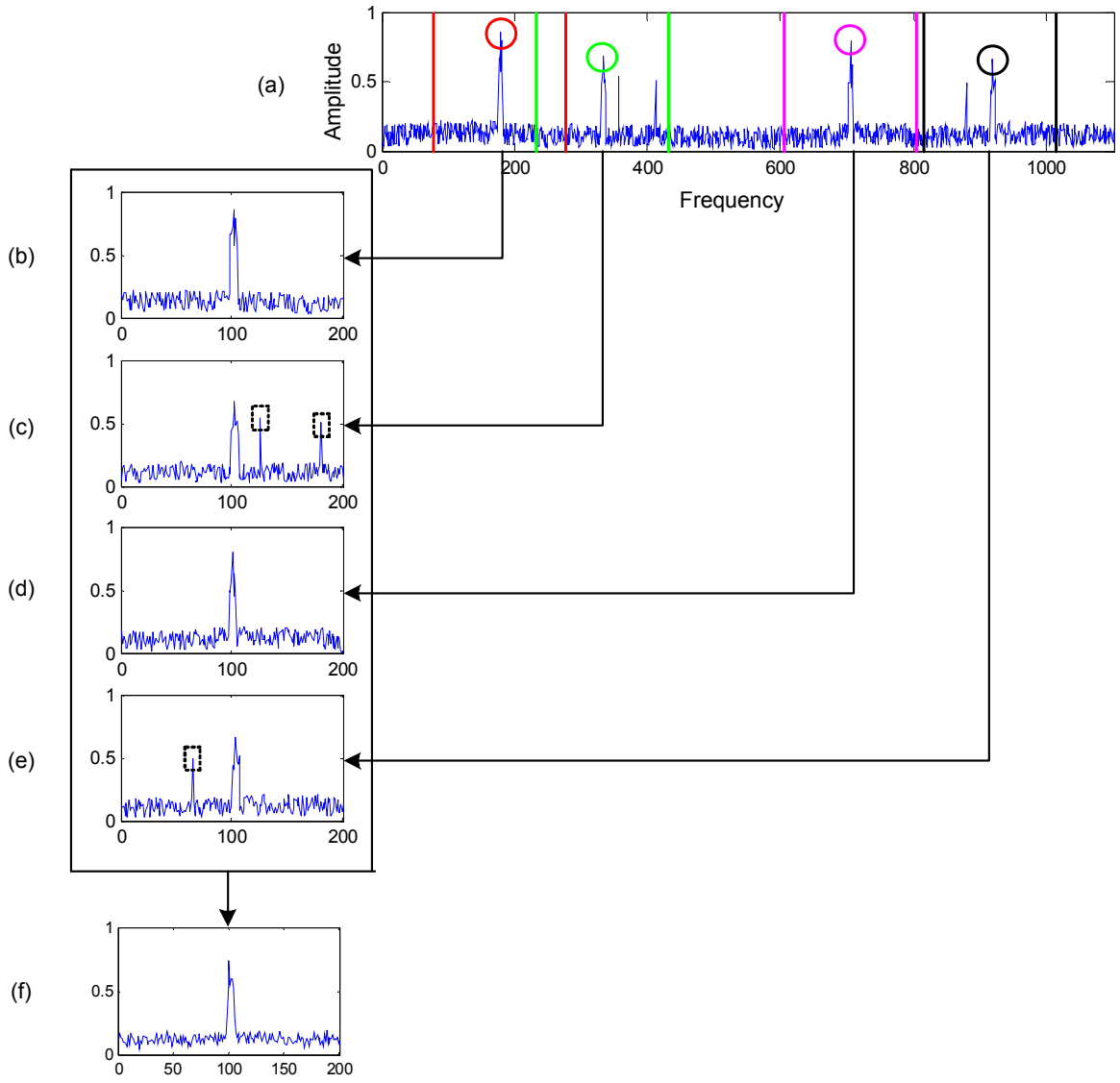

Figure 2.1. The formulation of FIS. (a) is the original spectrum; (b) – (e) are respective extracted local bands corresponding to the four circled fault frequency components (red, green, pink and black); (f) is the formulated FIS. The dotted lines in graph (a) represent the boundaries of the local bands; the dashed rectangular boxes represent outliers in the local bands.

2.2.2 Central Kurtosis Analysis

The classic kurtosis is a measure of the peakedness of a signal, computed as $\chi = \frac{\mu_4}{\sigma^4}$, where σ and μ_4 are the standard deviation and the fourth moment of the signal distribution, respectively. The

classic kurtosis measures all peaked frequency components of the FIS, which may not properly reveal the fault information. In this work, the fault detection aims to evaluate the peakedness of the center frequency component in the FIS. Therefore, a central kurtosis indicator is proposed to facilitate fault detection. Given the FIS $g(i)$; i = 1, 2, …, d_w+1, the relative amplitude of the center frequency components can be determined by

$$\nu_s = g_s - \xi\{g\} \tag{2.7}$$

where $g_s = \{g_i\}_{i=d_w/2+1}$ is the center discrete point in the FIS. The amplitude of fault frequency component over synchronized local bands (i.e., FIS), rather than the entire spectrum as in the classical methods, is used to examine fault information.

The variation of the FIS excluding center frequency component g_s can be evaluated by

$$\sigma_s = E\left\{\left(\tilde{g} - \xi\{\tilde{g}\}\right)^2\right\}^{\frac{1}{2}} \tag{2.8}$$

where $E\{\bullet\}$ represents the expectation function, and $\tilde{g} = \{g_i, i = 1, 2, ..., d_w/2, d_w/2+2, ..., d_w+1\}$.

Then the peakedness of the fault frequency component in the FIS can be measured by the central kurtosis, determined by

$$\chi_s = \begin{cases} \nu_s^{\,4} / \sigma_s^{\,4} & if\ \nu_s > 0 \\ 0 & if\ \nu_s \le 0 \end{cases} \tag{2.9}$$

2.2.3 Implementation of Spectrum Synch Technique

To recapitulate, the proposed SS technique is implemented for IM defect detection in the following steps:

1) Collect J stator current data sets ϕ_j, j = 1, 2, …, J, with the same time delay.

2) Determine the spectrum average Φ over J spectra. Then extract characteristic local bands using Equation (2.4). Synchronize the local bands to form the FIS using Equations (2.5) and (2.6), in order to reduce the noise effect and highlight fault features.

3) Compute the center frequency representative feature using Equation (2.7), and the variation of the FIS using Equation (2.8). The fault diagnosis can be performed by analyzing the central kurtosis computed from Equation (2.9).

2.3 Performance Evaluation

To evaluate the effectiveness of the proposed SS technique for IM fault detection, a series of tests have been conducted for the two common types of IM defects, IM broken rotor bar fault and IM bearing defect, using stator current signals. In rolling element bearings, defect occurs on the race of the fixed ring first since fixed ring material over the load zone experiences more cycles of fatigue loading than other bearing components (i.e., the rotating ring and rolling elements). Correspondingly, this test focuses on incipient bearing defect, or fault on the outer race (fixed ring in this case). The tests are conducted for two power supply frequencies f_p: f_p = 35Hz and 50Hz.

2.3.1 Experimental Setup

Figure 2.2 shows the experimental setup employed in the current work. The speed of the tested IM is controlled by a VFD-B AC speed controller (from Delta Electronics) with output frequency 0.1~400Hz. A magnetic particle clutch (PHC-50 from Placid Industries) is used as a dynamometer for external loading. Its torque range is from 1 to 30 lb·ft (1.356-40.675 N·m). The motor used for this research is made by Marathon Electric, and its specifications are summarized in Table 2.1. The gearbox (Boston Gear 800) is used to adjust the speed ratio of the dynamometer. The current sensors (102-1052-ND) are used to measure different phase currents. A rotary encoder (NSN-1024) is used to measure the shaft speed with the resolution of 1024 pulses per revolution. Stator current signals are collected using a Quanser Q4 data acquisition board, which are then fed to a computer for further processing.

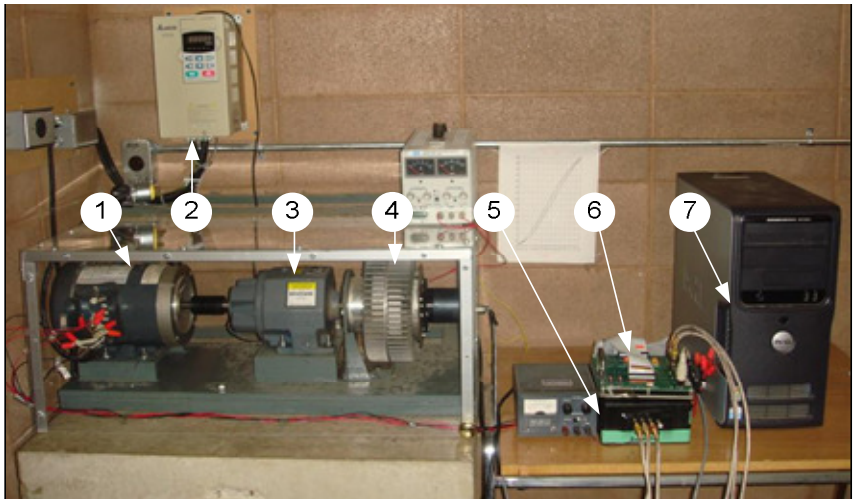

Figure 2.2. The motor experiment setup: (1) the tested IM, (2) the speed controller, (3) the gearbox, (4) the load system, (5) current sensors, (6) the data acquisition system, (7) the computer.

Table 2.1. Motor specifications.

Phase	3	Connection	Y
Poles	2	RPM	2850
HP	1/3	Rotor Bars	34
HZ	50	Stator Slots	24

2.3.2 Broken Rotor Bar Fault Detection

The fault detection of IM broken rotor bar defect is generally based on spectral analysis by inspecting fault-related sideband components in the spectrum:

$$f_{bl} = (1 - 2ks) f_p \tag{2.10}$$

$$f_{br} = (1 + 2ks) f_p \tag{2.11}$$

where f_{bl} and f_{br} are the respective left sideband and right sideband of the IM broken rotor bar fault, k = 1, 2, … ; f_p is the power supply frequency of the IM; $s = \frac{n_s - n_a}{n_s} \times 100\%$ is the slip of the IM. n_s (rpm) is the speed of rotating magnetic field, and n_a (rpm) is the shaft rotating speed. In the following tests, an IM containing three broken rotor bars is used to evaluate the proposed SS technique.

To examine the effectiveness of the proposed SS technique, the PSD based fault detection and the envelope analysis based fault detection are used for comparison. The PSD explores the energy distribution of the data over the spectrum; the envelope analysis performs amplitude demodulation to reveal fault features. In the PSD-based fault detection, the fault index can be represented as,

$$\chi_p = \begin{cases} \nu_p^{\,4} / \sigma_p^{\,4} & if \ \nu_p > 0 \\ 0 & if \ \nu_p \le 0 \end{cases} \tag{2.12}$$

The fault index of envelope analysis is given as

$$\chi_s = \begin{cases} \nu_e^{\,4} / \sigma_e^{\,4} & if \ \nu_e > 0 \\ 0 & if \ \nu_e \le 0 \end{cases} \tag{2.13}$$

where ν_p and ν_e represent the averages of top 50% high amplitude fault frequency components from PSD and envelope analysis respectively; σ_p and σ_e represent the standard deviations of the entire spectrum band of interests from PSD and envelope analysis respectively; χ_p and χ_e are the respective fault indices from PSD and envelope analysis. Therefore, these two techniques can be used to compare the local band synch method in the proposed SS technique, and the corresponding central kurtosis index.

1) 35Hz Supply Frequency: The first test aims to detect the IM with three adjacent broken rotor bars, 35Hz power supply frequency and medium-load condition (50% of rated power). To reduce the noise effect in the spectrum, twenty data sets are collected for spectrum averaging (i.e., J = 20). Other

settings are f_s = 65,536 Hz and f_w = 10Hz. Since 1,024 low-to-high voltage transitions represent one shaft revolution in the encoder signal, the high sampling frequency f_s is chosen to properly capture the encoder signal, so as to accurately estimate shaft speed (i.e., rotor speed). The frequency band [25Hz 45Hz] is used to detect broken rotor fault, because the amplitudes of high order (i.e., k in Equations (2.10) and (2.11)) characteristic frequencies are not prominent in the spectrum. Figure 2.3 illustrates the PSD of a healthy IM (Figure 2.3a), the PSD of an IM with broken rotor bars (Figure 2.3b), the envelope analysis of a healthy IM (Figure 2.3c), and the envelope analysis of an IM with broken rotor bars (Figure 2.3d), respectively. From Figures 2.3(b) and 2.3(d), it is seen that the broken rotor bar fault frequency components, although visible, do not prominently protrude in the spectrum. Therefore, a better fault detection technique is needed to extract useful information from multiple characteristic frequency components in the spectrum to generate a more reliable fault index.

The FIS, corresponding to a healthy IM (Figure 2.3a) and a broken rotor bar faulted IM (Figure 2.3b), are given in Figures 2.4(a) and 2.4(b), respectively. The unit of amplitude of the FIS is dB because the local bands are extracted from the PSD logarithmic spectrum. It is seen from Figure 2.4 that the center frequency component marked by the red dashed line (i.e., synchronized broken rotor bar fault frequency components) in Figure 2.4(b) has higher relative amplitude than that in Figure 2.4(a), which is evaluated by the index ν_s in the proposed SS technique. Figure 2.4(b) has similar spectrum variation as in Figure 2.4(a), which is examined by the value σ_s in the SS technique. Therefore the fault information in the FIS can be characterized by the index χ_s using the proposed SS technique.

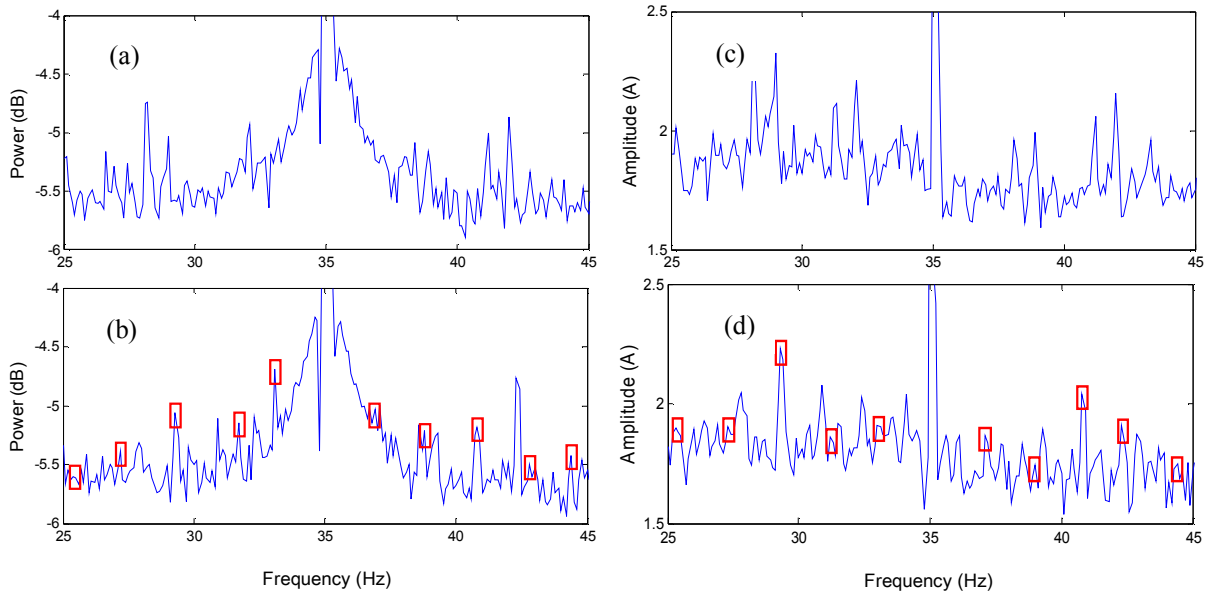

Figure 2.3. The spectrum average Φ corresponding to (a) a healthy IM using PSD; (b) an IM with broken rotor bar fault using PSD; (c) a healthy IM using envelope analysis; and (d) an IM with broken rotor bar fault using envelope analysis, at 35Hz supply frequency and medium-load condition. The red solid rectangular boxes in (b) and (d) highlight fault frequency components.

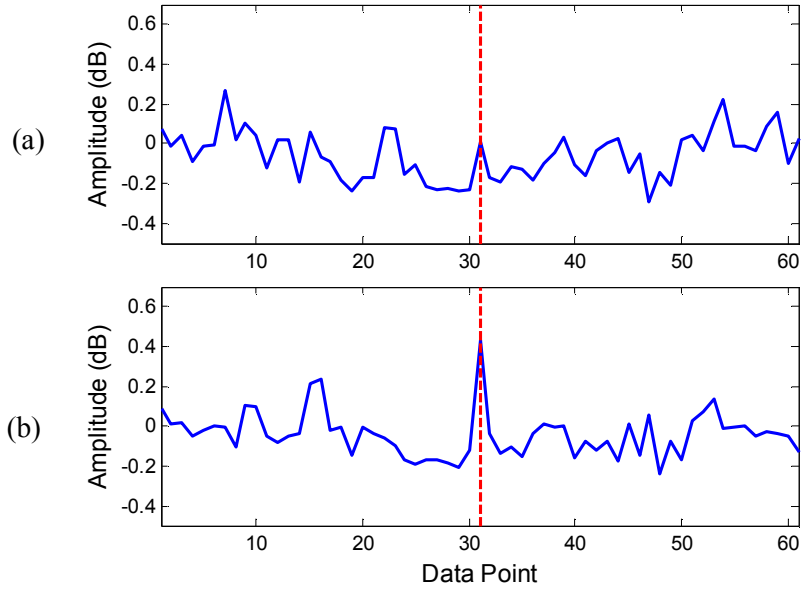

Figure 2.4. The FIS generated by the SS technique at 35Hz and medium-load condition (a) from a healthy IM, (b) from an IM with broken rotor bar fault.

The values of IM speed f_r (Hz) and indices corresponding to PSD, envelope analysis and the proposed SS are summarized in Table 2.2. From Table 2.2, it is seen that it is difficult to differentiate the IM broken rotor bar faulted condition from the IM healthy condition using envelope analysis, because the values of χ_e corresponding to these two IM conditions are similar. The PSD has a relatively large difference of χ_p of different IM conditions in this case; however the PSD suffers from interference of non-fault-related high amplitude frequency components and its χ_p values are too small to be relied on. The IM with broken rotor bar defect has considerably larger value of χ_s than that of the healthy IM using the proposed SS technique. Consequently, the proposed SS technique associated with its index χ_s can be used as a fault index for IM broken rotor bar fault detection in the stator current spectrum.

Table 2.2. Comparisons of central kurtosis indices for IM broken rotor bar fault detection.

Methods	PSD		Envelope analysis		SS	
Conditions	Healthy	Faulty	Healthy	Faulty	Healthy	Faulty
f_r	34.333	34.343	34.333	34.343	34.333	34.343
$\nu_p;\nu_e;\nu_s$	0.012	0.289	0.132	0.159	0.048	0.472
$\sigma_p;\sigma_e;\sigma_s$	0.751	0.764	0.176	0.161	0.122	0.098
$\chi_p;\chi_e;\chi_s$	6.519e-8	0.021	0.316	0.951	0.024	538

2) 50Hz Supply Frequency: The proposed SS technique is then used for IM broken rotor bar fault detection with 50Hz supply frequency and medium-load condition (50% of rated power). Other settings remain the same as in the previous tests. The spectrum of frequency band [35Hz 65Hz] is used for fault diagnosis. The selected band is shown in Figure 2.5 using PSD in Figure 2.5(a) and the envelope analysis in Figure 2.5(b), respectively. It is seen from Figure 2.5 that most of the fault frequency components are masked by noise, which cannot be used effectively for reliable fault diagnosis.

The FIS, corresponding to a healthy IM and an IM with broken rotor bar fault are given in Figures 2.6(a) and 2.6(b), respectively. The related IM condition indices are summarized in Table 2.3. It is seen that the relative amplitude of fault frequency component in Figure 2.6(b) is greater than that in Figure 2.6(a). From Table 2.3, the IM with broken rotor bar fault has a larger value of χ_s than that of a healthy IM, which indicates a broken rotor bar fault. The IM health condition with broken rotor bars cannot be differentiated from healthy condition using envelope analysis, associated with its fault index χ_e. Although the PSD index χ_p has a relatively large difference corresponding to different IM conditions, the performance of PSD may be degraded by the interference of non-fault-related high amplitude frequency components and its χ_p values are too small to be relied on. Hence, the proposed SS technique associated with its fault index χ_s can accurately discern the health condition of IMs with broken rotor bar fault under different supply frequencies, when compared to the related two classical methods.

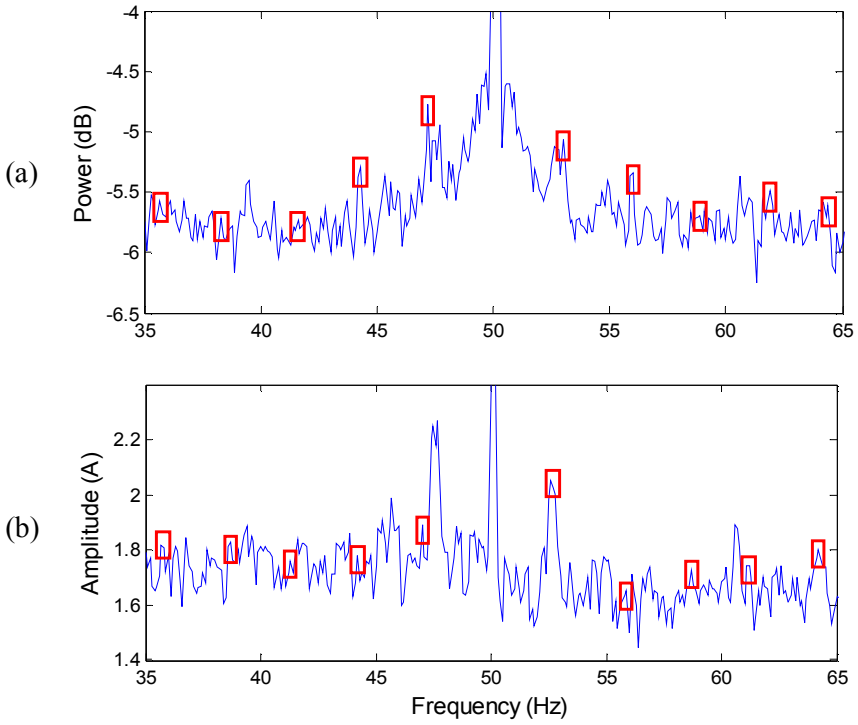

Figure 2.5. The spectrum Φ for an IM with broken rotor bar fault, 50Hz supply frequency and medium-load condition, using (a) PSD; and (b) envelope analysis. The red solid rectangular boxes highlight fault frequency components.

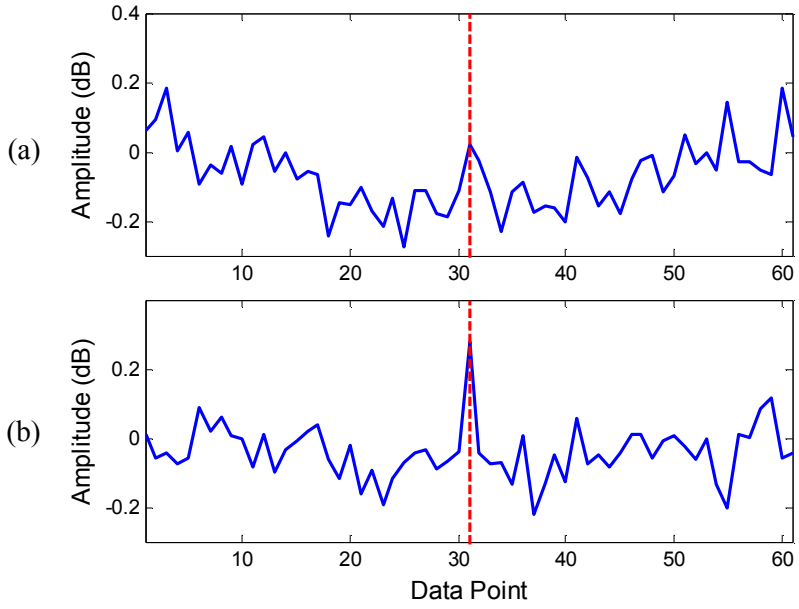

Figure 2.6. The FIS generated by the SS technique at 50Hz supply frequency and medium-load level (a) from a healthy IM; (b) from an IM with broken rotor bars.

Table 2.3. Comparisons of central kurtosis indices for IM broken rotor bar fault detection.

Methods	PSD		Envelope analysis		SS	
Conditions	Healthy	Faulty	Healthy	Faulty	Healthy	Faulty
f_r	49.180	49.089	49.180	49.089	49.180	49.089
$\nu_p;\nu_e;\nu_s$	0.103	0.195	0.083	0.119	0.092	0.325
$\sigma_p;\sigma_e;\sigma_s$	0.634	0.653	0.149	0.146	0.098	0.068
$\chi_p;\chi_e;\chi_s$	6.966e-4	0.008	0.096	0.441	0.777	522

2.3.3 Incipient Bearing Defect Detection

As mentioned earlier, bearing defects are the most common faults in IMs [2], which also represent the most challenging task in IM health condition monitoring, especially when using stator current signals [109]. A small dent of diameter approximately 1/16-inch was introduced on the outer race of

the bearing to simulate fatigue pitting defect. Whenever a rolling element rolls over the damaged region, impulses are generated, which then excite the resonance frequencies of the IM structures. The vibration-related outer race bearing defect characteristic frequency f_v is given in Equation (1.7). The corresponding characteristic current frequency f_c can be calculated using Equation (1.13).

1) 35Hz Supply Frequency: The proposed SS technique is first tested with stator current signals collected from an IM with the outer race defect, 35Hz power supply frequency, and light-load condition (20% of rated power). The settings for the proposed SS technique are selected as $J = 20$, $f_s =$ 65,536Hz, $f_w =$ 10Hz, and $f_v = 3.066\, f_r$. The high sampling frequency is used to accurately estimate the IM shaft speed. To obtain representative fault features, the frequency band [1000Hz, 2000Hz] is selected for bearing fault detection.

To have a clear view of fault frequency components, the frequency band [1090Hz 1360Hz] from an IM with outer race bearing defect is shown in Figure 2.7 using PSD (Figure 2.7a) and envelope analysis (Figure 2.7b). It is seen that the bearing fault frequency components are difficult to recognize due to the modulation of the signals with other IM frequency components.

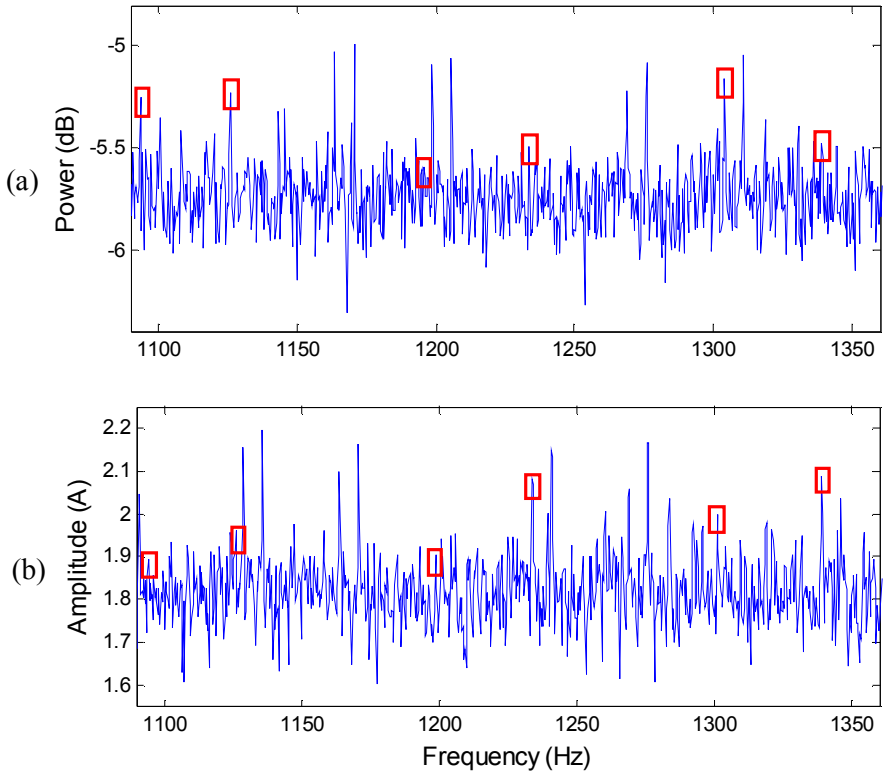

Figure 2.7. The spectrum average Φ for an IM with outer race bearing defects, 35Hz supply frequency, and light-load condition, using (a) PSD; and (b) envelope analysis. The rectangular boxes indicate bearing fault frequency components.

The FIS, corresponding to a healthy IM and an IM with the outer race defect, are given in Figures 2.8(a) and 2.8(b), respectively. The values of indices corresponding to these three fault detection techniques are summarized in Table 2.4. It is seen from Figure 2.8 that the fault frequency component in Figure 2.8(b) protrudes more significantly than that in Figure 2.8(a). In Table 2.4, the fault index χ_s of the IM with faulty bearing is greater than that of a healthy IM using the proposed SS technique, whereas the envelope analysis, associated with its index χ_e, cannot recognize different IM health conditions. The PSD index χ_p generates small values that cannot be relied on. Therefore, the SS technique can be used effectively for IM outer race bearing fault detection in this case, when compared to PSD and envelope analysis.

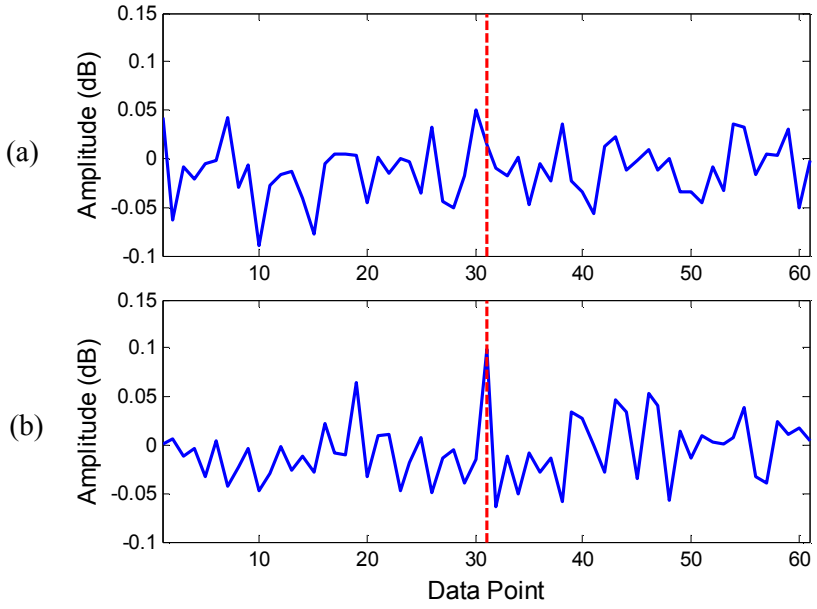

Figure 2.8. The FIS generated by the SS technique at 35Hz supply frequency and light-load condition (a) from a healthy IM; (b) from an IM with outer race bearing defect.

Table 2.4. Comparisons of central kurtosis indices for IM outer race bearing fault detection.

Methods	PSD		Envelope analysis		SS	
Conditions	Healthy	Faulty	Healthy	Faulty	Healthy	Faulty
f_r	34.746	34.716	34.746	34.716	34.746	34.716
$\nu_p;\nu_e;\nu_s$	0	0.006	0	0	0.023	0.107
$\sigma_p;\sigma_e;\sigma_s$	0.203	0.161	0.102	0.081	0.029	0.029
$\chi_p;\chi_e;\chi_s$	0	1.929e-6	0	0	0.396	185

2) 50Hz Supply Frequency: In this test, the IM supply frequency is set as 50Hz. The other settings remain the same as in previous test. The frequency band [1000Hz 2000Hz] of an IM with an outer race bearing defect is used for testing. The band [1400Hz 1750Hz] is shown in Figure 2.9, using PSD (Figure 2.9a) and envelope analysis (Figure 2.9b), respectively. From Figure 2.9, the bearing fault frequency components in the spectrum are masked by higher amplitude frequency components unrelated to the bearings, which will degrade the fault detection reliability.

The FIS of a healthy IM and an IM with outer race bearing defect are shown in Figure 2.10. The values of the shaft speed f_r and indices of the PSD, the envelope analysis and the proposed SS technique are listed in Table 2.5. It is seen that peaked center frequency component can be highlighted in Figure 2.10(b) than in Figure 2.10(a). From Table 2.5, it is seen that the values of χ_s in the proposed SS technique are much greater than that from a healthy IM. Thus the SS technique and its index χ_s can be used for IM outer race bearing defect detection at different supply frequencies.

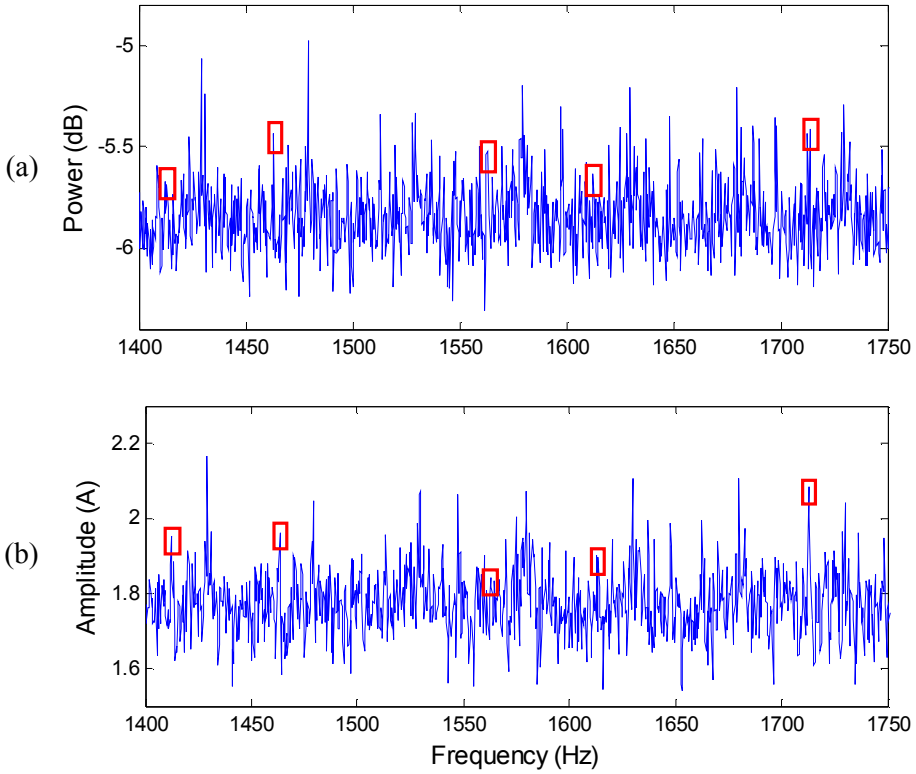

Figure 2.9. The spectrum average Φ for an IM with outer race bearing defects, 50Hz supply frequency, and light-load condition, using (a) PSD; and (b) envelope analysis.

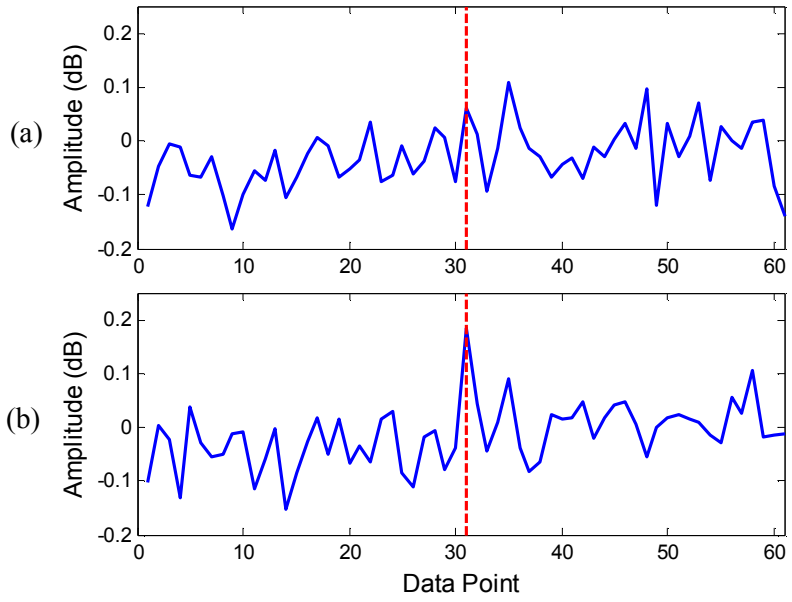

Figure 2.10. The FIS generated by the SS technique at 50Hz supply frequency and light-load condition (a) from a healthy IM; (b) from an IM with an outer race bearing defect.

Table 2.5. Comparisons of central kurtosis indices for IM outer race bearing fault detection.

Methods	PSD		Envelope analysis		SS	
Conditions	Healthy	Faulty	Healthy	Faulty	Healthy	Faulty
f_r	49.757	49.705	49.757	49.705	49.757	49.705
$\nu_p;\nu_e;\nu_s$	0	0.041	0.005	0.007	0.090	0.196
$\sigma_p;\sigma_e;\sigma_s$	0.176	0.173	0.087	0.086	0.055	0.052
$\chi_p;\chi_e;\chi_s$	0	0.003	1.091e-5	4.389e-5	7.170	202

2.4 Summary

A spectrum synch (SS) technique has been proposed in this work for IM fault detection using stator current signals. It could handle IM fault detection when the change in speed and/or load is not dramatic. This research focuses on broken rotor bar fault and incipient bearing fault detection. The local band synch technique is employed to synthesize bearing fault related features, so as to enhance

IM defect-related features. A central kurtosis analysis method is proposed to extract representative features from the FIS, which are then used to formulate fault indices. The effectiveness of the proposed IM fault detection technique is verified experimentally using IMs with the bearing defect and IM with the broken rotor bars, under different operating conditions. Test results have showed that the proposed SS technique and the related central kurtosis indices can recognize IM defect features effectively and reliably and can thus provide more relaible IM health condition monitoring information. For each IM fault, the proposed SS technique can be used for fault detection over certain load range. The fault detection in a broader load range will be discussed in Chapter 6.

The presented SS technique in this chapter is to achieve the first research objective in Section 1.3: 1) *Propose a new signal processing technique to extract representative features from the stator current signals for fault detection in IMs.*

Chapter 3
Selective Boosting Classifier

In this chapter, a novel selective boosting (sBoost) classifier is developed to effectively categorize the representative features extracted from selected signal processing techniques, in order to achieve a more accurate assessment of the IM health condition [110]. The proposed sBoost classifier will be presented, and subsequently verified by simulation tests. Its implementation for IM fault diagnosis will be presented in Chapter 6.

3.1 Classical AdaBoost Algorithm

Consider a binary classification problem with training data samples (x_1, y_1), (x_2, y_2), … , (x_N, y_N), where the class label is $y_i \in \{-1, +1\}$ and N is the number of samples in the training data set. If T is the maximum number of weak learners, the AdaBoost algorithm takes the following processes:

(a) Initialize the distribution of training data set $L_1(i) = 1/N$, i = 1, 2, …, N.

FOR t = 1 : T

(b) Train the weak learner $h_t \in \{-1, +1\}$ using training data set with distribution L_t.

(c) Compute the sum of weights of misclassified samples $\phi_t = \sum_{i=1}^{N} L_t(i)[\![h_t(x_i) \neq y_i]\!]$. The logic operator is defined as $[\![\Gamma]\!] = \begin{cases} 1 & \Gamma \neq 0 \\ 0 & \Gamma = 0 \end{cases}$.

(d) Calculate the weight of the weak learner h_t, $\alpha_t = \frac{1}{2}\ln\left(\frac{1-\phi_t}{\phi_t}\right)$.

(e) Update the weights of the training samples, $L_{t+1}(i) = \frac{L_t(i)\exp(-\alpha_t h_t(x_i) y_i)}{J_t}$, where $J_t = \sum_{i=1}^{N} L_t(i)\exp(-y_i \alpha_t h_t(x_i))$.

END

(f) Normalize the weights of weak learner h_t, $\alpha_t' = \frac{\alpha_t}{\sum_{t=1}^{T} \alpha_t}$.

If $f(x_i) = \sum_{t=1}^{T} \alpha_t' h_t(x_i)$, and $H(x_i) = sign[f(x_i)]$ is the predicted class label, the margin of a sample (x_i, y_i) will be

$$\hat{\rho}_t(x_i) = \frac{y_i \sum_{t=1}^{T} \alpha_t h_t(x_i)}{\sum_{t=1}^{T} \alpha_t} \tag{3.1}$$

Theorem 1 (Schapire *et al.* [111]): Given the weighted training error $\phi_t = \sum_{i=1}^{N} L_t(i)[[h_t(x_i) \neq y_i]]$, for any δ, the probability that $\hat{\rho}_t(x_i) \leq \delta$ can be given as,

$$\boldsymbol{P}[\hat{\rho}_t(x_i) \leq \delta] \leq 2^T \prod_{t=1}^{T} \sqrt{\phi_t^{1-\delta}(1-\phi_t)^{1+\delta}} \tag{3.2}$$

When $\delta = 0$, $\boldsymbol{P}[\hat{\rho}_t(x_i) \leq 0]$ represents the probability that a sample is misclassified, which should be minimized.

3.2 Selective Boosting Analysis

3.2.1 Sample Weight Regulator

The AdaBoost techniques conduct greedy learning of hard-to-learn samples, which may reduce the classification of easy-to-learn samples. As a result, the overfitting problem occurs. The accumulated weight of a sample is an effective indicator, which can determine the degree this sample contributes to potential overfitting. Thus it can be used to properly adjust the penalty of this sample so as to reduce overfitting.

Let β_t be the weight of the weak learner h_t in sBoost, the margin can be defined in the same

manner as that in Equation (3.1)

$$\rho_t(x_i) = \frac{y_i \sum_{t=1}^{T} \beta_t h_t(x_i)}{\sum_{t=1}^{T} \beta_t} \tag{3.3}$$

The accumulated weights of samples can be computed as

$$\kappa_t(i) = \frac{\exp(\rho_t(x_i))}{\sum_{i=1}^{N} \exp(\rho_t(x_i))} \tag{3.4}$$

$$\xi_t(i) = \sum_{j=1}^{t} (\beta_t \kappa_t(i)) \tag{3.5}$$

$$\psi_t(i) = \frac{\xi_t(i)}{\max\{\xi_t(i)\}} \beta_t \tag{3.6}$$

Since some noisy samples may induce overfitting, they need to be processed properly in the training process. In the proposed sample weight regulator, the *k*-nearest-neighbor method is adopted to identify noisy samples and evaluate their noise level. Given *K* nearest samples of a target sample in the training data set, if *p* samples of these *K* samples have the same class label as this target sample ($0 \le p \le K$), the level of noise of this sample will be estimated by

$$\theta_i = 1 - \frac{p}{K} \tag{3.7}$$

where the noise level is independent of time step *t*.

The proposed sample weight regulator is given by

$$\varpi_t(i) = \psi_t(i)\theta_i = \frac{\xi_t(i)}{\max\{\xi_t(i)\}}\beta_t\theta_i \tag{3.8}$$

It can be proven that $0 \le \varpi_t(i) \le \beta_t$. Accordingly, the update of the i^{th} sample weight at step t can be formulated as:

$$\begin{aligned} L_{t+1}(i) &= \frac{L_t(i)\exp\left(-\beta_t h_t(x_i) y_i - \varpi_t(i)\right)}{J'_t} \\ &= \frac{1}{N}\frac{\exp\left\{-\sum_{t=1}^{T}\left[y_i\beta_t h_t(x_i) + \varpi_t(i)\right]\right\}}{\prod_{t=1}^{T} J'_t} \end{aligned} \tag{3.9}$$

where J_t' is the cost function of sBoost.

By using the sample weight regulator $\varpi_t(i)$, the noisy samples will be processed according to their noise level. For each sample, the maximum penalty reduction is determined by its accumulated weight, which will be lessened by the sample's noise level. Consequently, the less noisy samples will not have over-reduced penalties to decrease their chance of being correctly classified, and the penalties of noisy samples can be adjusted to reduce overfitting.

Given $0 < \exp\left(-\varpi_t(i)\right) \le 1$, the cost function associated with the sample weight regulator is suggested as:

$$\begin{aligned} J'_t &= \sum_{i=1}^{N} L_t(i)\exp\left(-y_i\beta_t h_t(x_i) - \varpi_t(i)\right) \\ &= \left(\sum_{y_i h_t(x_i)=1} L_t(i)\exp\left(-\varpi_t(i)\right)\right)\exp\left(-\beta_t\right) + \left(\sum_{y_i h_t(x_i)=-1} L_t(i)\exp\left(-\varpi_t(i)\right)\right)\exp\left(\beta_t\right) \\ &= \varepsilon_t \exp\left(-\beta_t\right) + \tau_t \exp\left(\beta_t\right) \end{aligned} \tag{3.10}$$

where $\varepsilon_t = \sum_{y_i h_t(x_i)=1} L_t(i)\exp(-\varpi_t(i))$ and $\tau_t = \sum_{y_i h_t(x_i)=-1} L_t(i)\exp(-\varpi_t(i))$.

It can be seen that $0 < \varepsilon_t < 1$, and $0 < \tau_t < 1$. In addition, $\sum_{i=1}^{N} L_{t+1}(x_i) = 1$ can be derived from Equations (3.9) and (3.10).

Taking derivative of Equation (3.10) with respect to β_t and letting it equal zero, the optimal weights of weak classifier h_t will be

$$\beta_t = \frac{1}{2}\ln\left(\frac{\varepsilon_t}{\tau_t}\right) \tag{3.11}$$

Inserting Equation (3.11) into Equation (3.10), we get

$$\begin{aligned} J_t' &= \varepsilon_t \exp(-\beta_t) + \tau_t \exp(\beta_t) \\ &= \varepsilon_t \exp\left(-\frac{1}{2}\ln\left(\frac{\varepsilon_t}{\tau_t}\right)\right) + \tau_t \exp\left(\frac{1}{2}\ln\left(\frac{\varepsilon_t}{\tau_t}\right)\right) \\ &= 2\sqrt{\varepsilon_t \tau_t} \end{aligned} \tag{3.12}$$

Let

$$\begin{aligned} \rho_t'(x_i) &= \frac{\sum_{t=1}^{T}[y_i \beta_t h_t(x_i) + \varpi_t(i)]}{\sum_{t=1}^{T}\beta_t} \\ &= \frac{y_i \sum_{t=1}^{T}\beta_t h_t(x_i)}{\sum_{t=1}^{T}\beta_t} + \frac{\sum_{t=1}^{T}\varpi_t(i)}{\sum_{t=1}^{T}\beta_t} \end{aligned} \tag{3.13}$$

Since $\boldsymbol{P}[\rho_t(x_i) \le 0]$ represents the probability that a sample is misclassified, the minimization of

$\boldsymbol{P}[\rho_t(x_i) \le 0]$ is equivalent to

$$\mathbf{P}[\rho_t(x_i) \le 0] = \mathbf{P}\left[\rho_t'(x_i) \le \frac{\sum_{t=1}^{T} \varpi_t(i)}{\sum_{t=1}^{T} \beta_t}\right] \le \mathbf{P}[\rho_t'(x_i) \le \gamma] \tag{3.14}$$

where

$$\gamma = \sup_i \left(\frac{\sum_{t=1}^{T} \varpi_t(i)}{\sum_{t=1}^{T} \beta_t} \right) \tag{3.15}$$

The probability of $\rho_t'(x_i) \le \gamma$ can be derived by

$$\sum_{t=1}^{T} [y_i \beta_t h_t(x_i) + \varpi_t(i)] \le \gamma \sum_{t=1}^{T} \beta_t \tag{3.16}$$

$$\exp\left(\gamma \sum_{t=1}^{T} \beta_t - \sum_{t=1}^{T} [y_i \beta_t h_t(x_i) + \varpi_t(i)] \right) \ge 1 \tag{3.17}$$

$$\boldsymbol{P}[\rho'(x_i) \le \gamma] \le \boldsymbol{E}\left[\exp\left(\gamma \sum_{t=1}^{T} \beta_t - \sum_{t=1}^{T} [y_i \beta_t h_t(x_i) + \varpi_t(i)] \right) \right]$$

$$= \frac{1}{N} \exp\left(\gamma \sum_{t=1}^{T} \beta_t \right) \sum_{i=1}^{N} \exp\left(- \sum_{t=1}^{T} [y_i \beta_t h_t(x_i) + \varpi_t(i)] \right)$$

$$= \exp\left(\gamma \sum_{t=1}^{T} \beta_t \right) \left(\prod_{t=1}^{T} J_t' \right) \sum_{i=1}^{N} L_{t+1}(x_i)$$

$$= \exp\left\{ \gamma \sum_{t=1}^{T} \left[\frac{1}{2} \ln\left(\frac{\varepsilon_t}{\tau_t} \right) \right] \right\} \left[\prod_{t=1}^{T} \left(2\sqrt{\varepsilon_t \tau_t} \right) \right]$$

$$= 2^T \prod_{t=1}^{T} \left(\sqrt{\left(\frac{\varepsilon_t}{\tau_t} \right)^r \varepsilon_t \tau_t} \right) \tag{3.18}$$

Since the condition $0 \le \varpi_t(i) \le \beta_t$ is satisfied, it can be derived that $0 \le \frac{\sum_{t=1}^{T} \varpi_t(i)}{\sum_{t=1}^{T} \beta_t} \le 1$, and $0 \le \gamma \le 1$.

From Equation (3.18), it is seen that $0 < \sqrt{\left(\frac{\varepsilon_t}{\tau_t} \right)^r \varepsilon_t \tau_t} < 1$, when $0 < \varepsilon_t < \tau_t < 1$. The larger the *T* is, the smaller the value $\prod_{t=1}^{T} \left(\sqrt{\left(\frac{\varepsilon_t}{\tau_t} \right)^r \varepsilon_t \tau_t} \right)$ becomes. From Equations (3.14) and (3.18), it is easy to derive that $\boldsymbol{P}[\rho_t(x_i) \le 0] \le 2^T \prod_{t=1}^{T} \left(\sqrt{\left(\frac{\varepsilon_t}{\tau_t} \right)^r \varepsilon_t \tau_t} \right)$. Consequently, the more weak learners are incorporated into the ensemble (or the larger the *T* is), the lower the probability in which a sample is misclassified (or the smaller the $\boldsymbol{P}[\rho_t(x_i) \le 0]$ becomes).

The ensemble classifier is formulated as

$$H(x_i) = sign\left\{ \sum_{t=1}^{T} \beta_t' h_t(x_i) \right\} \tag{3.19}$$

The confidence rate is given by

$$C(x_i) = \left| \sum_{t=1}^{T} \beta_t' h_t(x_i) \right| \tag{3.20}$$

3.2.2 Error Correction Method

The sample weight regulator is used in the proposed sBoost to alleviate the penalties of noisy hard-to-classify samples, in order to reduce overfitting. However, the overfitting problem may still occur due to the inconsistency of training data and test data. To address this problem, an error correction method is suggested in the proposed sBoost to detect potential clean samples in the test data, and correctly classify them. To implement the proposed error correction method, each dimension of the training data $x(i)$ is normalized over [0, 1]. Let $\hat{x}$ denote the original data; U^d and L^d are the respective maximum and minimum value of the original training data at dimension d, then

$$x_i^d = \frac{\hat{x}_i^d - L^d}{U^d - L^d}, \quad i = 1, 2, \ldots, N \tag{3.21}$$

where N is the number of training data, and d is the index of dimension. Each test sample $x'(i)$ is also processed using the same way as the training data.

$$x_j'^d = \frac{\hat{x}_j'^d - L^d}{U^d - L^d}, \quad j = 1, 2, \ldots, N_t \tag{3.22}$$

where N_t is the number of test data.

The proposed error correction method can be represented as

$$K_t(j) = \sum_{i \in S_j} \left(y_i / q \right), \quad j = 1, 2, \ldots, N_t \tag{3.23}$$

where $y_i \in \{-1, 1\}$ is the class label of the training data. S_j represents a set that contains q samples in the training data that are nearest neighbors to the test sample y_j'. It is seen that $\left| K_t(j) \right| \le 1$. The error correction method is implemented with the following rules:

(I) If $K_t(j) = 1$, the test sample x_j' is considered as a clean sample with class label y_j' =1.

(II) If $K_t(j) = -1$, the test sample x_j' is considered as a clean sample with class label y_j' = -1.

(III) If $|K_t(j)| < 1$, the class label of the test sample x_j' is determined by boosting using the sample weight regulator.

Potential clean samples (i.e., $|K_t(j)| = 1$) in the test data set are detected using Equation (3.23) and assigned predicted class label according to (I) and (II). Other test samples are classified using boosting with the sample weight regulator. By using this error correction method, the potential clean samples will not be misclassified by the classifier with the potential overfitting problem. Hence the overfitting can be further reduced to improve the classification accuracy.

3.2.3 Implementation of sBoost Technique

The developed sBoost technique includes the boosting and the error correction processes. The proposed boosting strategy with sample weight regulator will be implemented in the following steps:

(a) Initialize the distribution of training data set $L_1(i) = 1/N$, i = 1, 2, …, N.

(b) Train the weak learner $h_t \in \{-1, +1\}$ using training data set with distribution L_t.

(c) Calculate $\varepsilon_t = \sum_{y_i h_t(x_i)=1} L_t(i)\exp(-\varpi_t(i))$; $\tau_t = \sum_{y_i h_t(x_i)=-1} L_t(i)\exp(-\varpi_t(i))$; and the weight of the weak learner h_t, $\beta_t = \frac{1}{2}\ln\left(\frac{\varepsilon_t}{\tau_t}\right)$.

(d) If $0 < \tau_t < \varepsilon_t < 1$ is not satisfied, $L_{t+1}(i) = 1/N$; otherwise, update the weights of the training samples, $L_{t+1}(i) = \frac{L_t(i)\exp(-\beta_t h_t(x_i) y_i - \varpi_t(i))}{J'_t}$, where $J'_t = \sum_{i=1}^{N} L_t(i)\exp(-y_i \beta_t h_t(x_i) - \varpi_t(i))$, i = 1, 2, …, N.

(e) Repeat steps (b) to (d) with t = 1 : T.

(f) Formulate the ensemble classifier as $H(x_i) = sign\left[\sum_{t=1}^{T} \beta'_t h_t(x_i)\right]$, where $\beta'_t = \frac{\beta_t}{\sum_{t=1}^{T} \beta_t}$ is the normalized the weights of weak learner h_t.

After boosting using the sample weight regulator, the errors are further corrected to improve the classification accuracy. Given a test sample x_j', the error correction is undertaken using the following steps:

(a) Determine if x_j' is a potential clean sample using Equation (3.23). If it is, its class label is predicted using rules (I) - (II).

(b) If $|K_t(j)| < 1$, apply rule (III). Predict the class label of x_j' by $H(x_j') = sign\left[\sum_{t=1}^{T} \beta_t' h_t(x_j')\right]$.

3.3 Performance Evaluation

The effectiveness of the proposed sBoost technique is verified in this section by simulation tests using 12 benchmark data sets from KEEL database [112]. The dataset information is summarized in Table 3.1. These data sets range in size from 106 to 19,020, as well as in dimension (features) from 5 to 60. To make a comparison, the classical AdaBoost algorithm, AB in short, the weight decay AdaBoost algorithm, WAB in short [69], the noise detection based AdaBoost algorithm, ND in short [70], and sBoost without error correction mechanism, sBoost-1, are employed. The sBoost using the proposed error correction method is denoted sBoost-2. The AB, WAB, and ND are used to compare the effect of proposed sample weight regulator in sBoost; whereas the sBoost-1 is used to evaluate the effectiveness of the suggested error correction mechanism.

Table 3.1. Data set information.

Data Name	No. of Samples	No. of Features
Phoneme	5404	5
Appendicitis	106	7
Pima	768	8
Breast	277	9
Tic-tac-toe	958	9
Magic	19020	10
Heart	270	13
Australian	690	14
German	1000	20
Twonorm	7400	20
Wdbc	569	30
Sonar	208	60

In each test, the maximum number of weak learners is fixed at 200 for the five classifiers: AB, WAB, ND, sBoost-1 and sBoost-2. The decision tree C4.5 is used as the weak learner. Artificial noise is added to each of the benchmark data sets. Noise level E% (e.g., 0% - 10%) in a data set is realized by randomly selecting E% of total samples in the training data set and changing their class labels [113]. The 2-fold cross-validation approach is conducted 5 times (i.e., 5×2 cross-validations) for each benchmark data set. In each 2-fold cross-validation, the data are randomly split into two data sets for two runs: one set is used for training, and the other set for testing in the first run. The two data sets are then switched for training and testing in the second run. The parameters of the sBoost technique are selected as $K = 5$ and $q = 5$ by trial and error.

To examine the effectiveness of the proposed sBoost technique over these classical classifiers, the differences between successful classification rates (SRs) of ND, sBoost-1 and sBoost-2 and SR of WAB, that is, SR(ND) – SR(WAB), SR(sBoost-1) – SR(WAB) and SR(sBoost-2) – SR(WAB) for these 11 benchmark data sets are demonstrated in Figures 3.1 to 3.3. The SR of the WAB is the minuend of the subtraction. The indices 1 ~ 11 in the horizontal axes represent the data sets of {Phoneme, Appendicitis, Pima, Breast, Magic, Heart, Australian, German, Twonorm, Wdbc, Sonar},

respectively. Since the values of SR(sBoost-2) – SR(WAB) of the Tic-tac-toe data set fall within the range of [0.023, 0.038], much larger than those in other cases, they are not shown in Figures 3.1 to 3.3 for proper view scales. The difference values of SR(AB) with respect to other classifiers (i.e., WAB, ND, sBoost-1 and sBoost-2) are also much larger than those in other cases, for example, SR(sBoost-2) – SR(AB) $\in$ [0.047,0.127]; correspondingly these values are not shown in Figures 3.1 to 3.3.

From Figures 3.1 to 3.3, it is seen that sBoost-1 and sBoost-2 outperform WAB and ND in overall performance of these 11 data sets. Their superior performance over WAB is clear as more weak learners are incorporated into the ensemble, because the sample weight regulator in sBoost-1 and sBoost-2 can effectively adjust noisy sample weights to reduce overfitting. The sBoost-1 and sBoost-2 outperform ND due to their efficient noise-level-based adaptive weight regulation. On the other hand, the sBoost-2 outperforms sBoost-1 in most test scenarios, which can detect and process clean samples more effectively (i.e., labeled).

It can be realized from Figure 3.1(c) that the WAB performs better than the proposed sBoost techniques in the fifth, eighth, and ninth data sets, whereas ND can provide superior performance in the third and sixth data sets. The sBoost techniques reduce the penalty of noisy samples, in order to improve their generalization capability. This may result in misclassification of some noisy samples, and in turn lead to poor classification performance in the premature stage (e.g., with 50 weak learners in this case). Nevertheless, the proposed sBoost techniques surpass both WAB and ND as more weak learners are incorporated into the ensemble as demonstrated in Figure 3.3(c) (e.g., with 200 weak learners in this case), because the sBoost technique provides better generalization capability than WAB and ND.

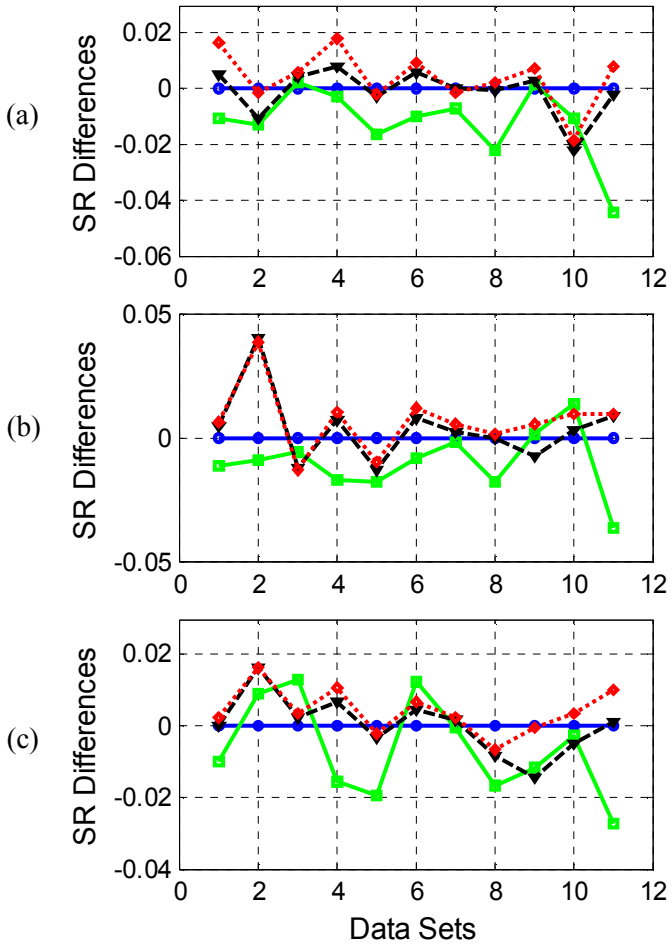

Figure 3.1. The differences of SR using 50 weak learners with different noise levels: (a) 0%; (b) 5%; and (c) 10%. The blue-circle-solid line represents SR(WAB) – SR(WAB); the green-square-solid line represents SR(ND) – SR(WAB); the black-triangle-dashed line represents SR(sBoost-1) – SR(WAB); the red-diamond-dotted line represents SR(sBoost-2) – SR(WAB).

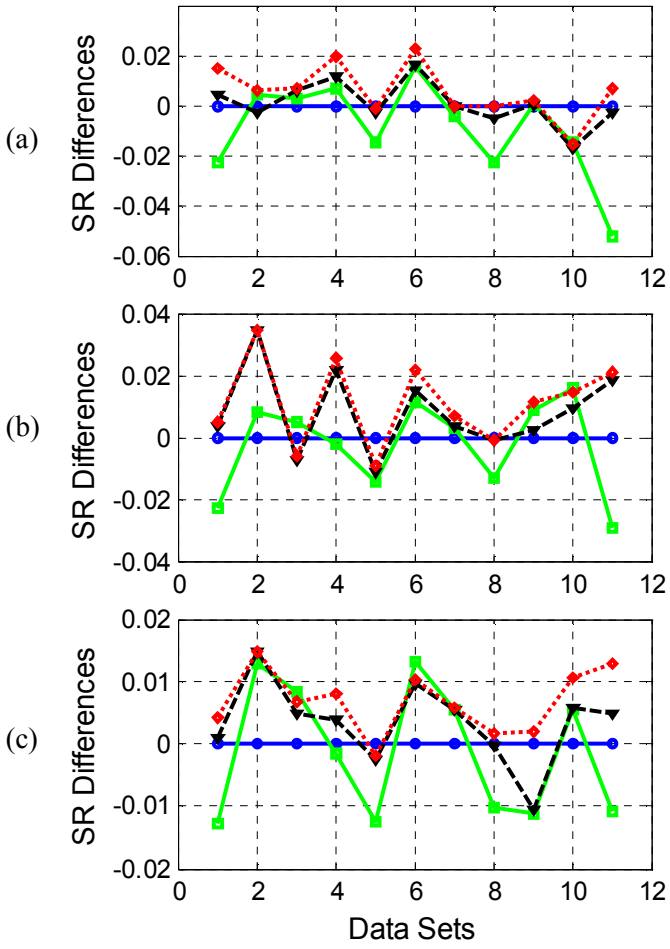

Figure 3.2. The differences of SR using 100 weak learners with different noise levels: (a) 0%; (b) 5%; and (c) 10%. The blue-circle-solid line represents SR(WAB) – SR(WAB); the green-square-solid line represents SR(ND) – SR(WAB); the black-triangle-dashed line represents SR(sBoost-1) – SR(WAB); the red-diamond-dotted line represents SR(sBoost-2) – SR(WAB).

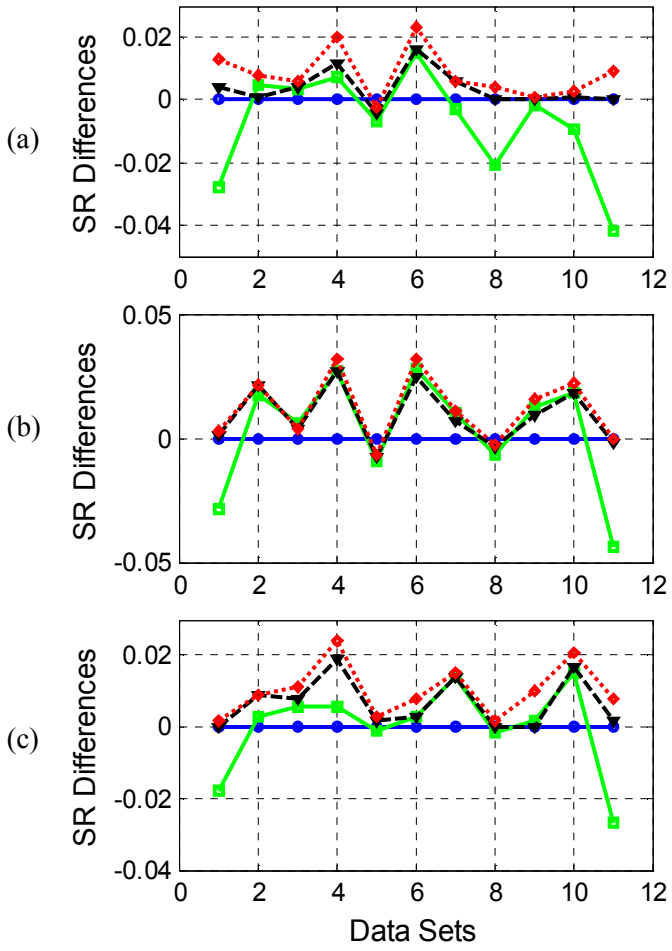

Figure 3.3. The differences of SR using 200 weak learners with different noise levels: (a) 0%; (b) 5%; and (c) 10%. The blue-circle-solid line represents SR(WAB) – SR(WAB); the green-square-solid line represents SR(ND) – SR(WAB); the black-triangle-dashed line represents SR(sBoost-1) – SR(WAB); the red-diamond-dotted line represents SR(sBoost-2) – SR(WAB).

Figure 3.4 shows the averaged test errors of the real-world breast cancer data set using these five boosting techniques. It is seen that both WAB and ND generate lower average test errors than AB as the number of weak learner increases, because of their capacity for addressing the overfitting problem (i.e., the test error increases as the number of weak learner increases). Nevertheless, their averaged test errors still increase somewhat because overfitting is not sufficiently reduced. The sBoost techniques outperform both WAB and ND, without clear overfitting, because their sample weight

regulator can effectively reduce the overfitting problem and improve the classification accuracy. The sBoost-2 overtakes sBoost-1 because the potential clean samples can be correctly classified.

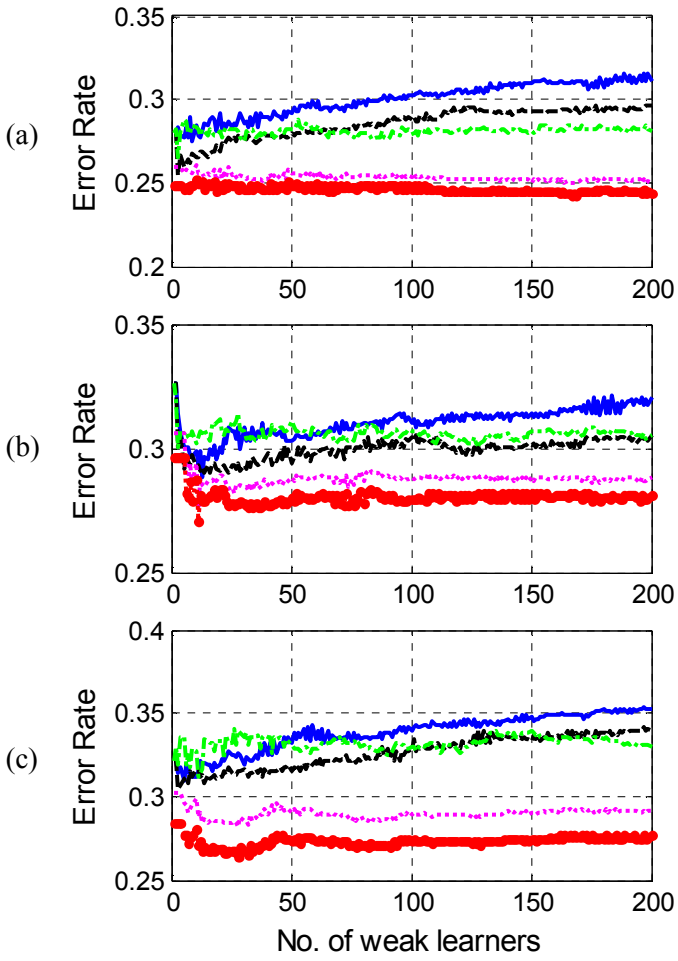

Figure 3.4. The averaged test errors of breast cancer data with different noise levels: (a) 0%; (b) 5%; and (c) 10%. The blue-solid line represents AB; the black-dashed line represents WAB; the green-dash-dotted line represents ND; the magenta-dotted line represents sBoost-1; the red-square-dotted line represents sBoost-2.

Table 3.2 summarizes the averaged test errors of the related techniques over 12 data sets. It is seen that the WAB and ND generate better results than AB because of their strategy to process hard-to-learn samples. sBoost-1 outperforms AB, WAB and ND at all three noise levels, because it can effectively detect noisy samples and reduce their effect on data distributions. The error correction

method in sBoost-2 can further improve classification accuracy compared to sBoost-1, because it can correct some misclassified clean samples due to the overfitting problem.

Table 3.2. Averaged test errors of AB, WAB, ND, sBoost-1, and sBoost-2 using 5×2 cross-validation and 200 weak learners.

Noise	0%					5%					10%				
Data Name	AB	WAB	ND	sB1	sB2	AB	WAB	ND	sB1	sB2	AB	WAB	ND	sB1	sB2
Phoneme	.193	.192	.220	.188	**.179**	.220	.215	.243	.214	**.212**	.260	.253	.271	.253	**.251**
Appendicitis	.192	.189	.184	.188	**.181**	.245	.232	.214	**.211**	**.211**	.294	.292	.289	**.283**	**.283**
Pima	.258	.249	**.232**	.245	.243	.295	.286	**.280**	.282	.282	.320	.310	**.284**	.302	.299
Breast	.299	.292	.285	.280	**.272**	.330	.320	.293	.293	**.288**	.347	.344	.338	.325	**.320**
Tic-tac-toe	.245	**.166**	.278	.192	.191	.269	.221	.252	.222	**.191**	.304	**.242**	.325	.258	.257
Magic	.172	**.165**	.172	.169	.167	.211	**.206**	.216	.214	.213	.245	.242	.243	.240	**.239**
Heart	.207	.199	.178	.183	**.176**	.276	.259	.231	.234	**.227**	.324	.307	**.296**	.304	.299
Australian	.163	.149	.152	**.143**	**.143**	.206	.195	.185	.188	**.184**	.239	.225	.211	.211	**.210**
German	.260	.254	.275	.254	**.250**	.293	**.281**	.288	.285	.284	.312	.308	.310	.308	**.306**
Twonorm	.032	.031	.033	.031	**.030**	.109	.101	.088	.092	**.085**	.160	.150	.148	.150	**.140**
Wdbc	.050	.034	.043	.033	**.031**	.116	.112	.094	.094	**.090**	.172	.161	.146	.144	**.140**
Sonar	.208	.204	.246	.204	**.195**	.269	**.253**	.297	.255	**.253**	.317	.313	.340	.311	**.305**

Note: sB1 represents sBoost-1, and sB2 represents sBoost-2.

The statistical tools, Wilconxon signed-rank test [114] and sign test [115], were also employed to evaluate the pairwise statistical significant differences among these five classifiers, AB, WAB, ND, sBoost-1 and sBoost-2. The comparison results are summarized in Tables 3.3-3.5 as a matrix of the pairwise statistical tests. The numbers of wins, ties and loses are represented as *w*/*t*/*l*. p_s is the *p*-value of the sign test and p_w denotes the *p*-value of Wilcoxon signed-rank test. Symbol ' ✓' is used to indicate that the two classifiers are significantly different at 95% confidence level; whereas '✖' indicates that the two classifiers are significantly different at 90% confidence level.

It is clear from Tables 3.3 to 3.5 that WAB, ND, sBoost-1 and sBoost-2 outperform AB at the three noise levels. Although sBoost-1 has no clear difference over WAB at 0% noise level, its performance is much better than WAB as the noise level increases, which can be attributed to its efficient noise manipulation mechanism. The ND performs better than WAB at 5% and 10% noise level, however,

sBoost techniques outperform ND at all three noise levels because of their enhanced noisy sample processing strategy. On the other hand, the sBoost-2 outperforms the sBoost-1 at all three noise levels due to its effective clean sample correction approach.

Table 3.3. Statistical significant comparisons of five boosting techniques with 0% noise over 12 data sets.

Methods		AB	WAB	ND	sBoost-1	sBoost-2
Mean		0.1899	0.1770	0.1915	0.1758	0.1715
AB	*w/t/l*		12/0/0	7/0/5	12/0/0	12/0/0
	p_s		0.0005 √	0.7744	0.0005 √	0.0005 √
	p_w		0.0005 √	0.9248	0.0005 √	0.0005 √
WAB	*w/t/l*			4/0/8	7/3/2	10/0/2
	p_s			0.3877	0.1797	0.0386 √
	p_w			0.2407	0.2852	0.0503 ×
ND	*w/t/l*				9/0/3	11/0/1
	p_s				0.1460	0.0063 √
	p_w				0.0664 ×	0.0063 √
sBoost-1	*w/t/l*					11/1/0
	p_s					0.0010 √
	p_w					0.0010 √

Table 3.4. Statistical significant comparisons of five boosting techniques with 5% noise over 12 data sets.

Methods		AB	WAB	ND	sBoost-1	sBoost-2
Mean		0.2366	0.2234	0.2235	0.2153	0.2100
AB	*w/t/l*		12/0/0	9/0/3	11/0/1	11/0/1
	p_s		0.0005 √	0.1460	0.0063 √	0.0063 √
	p_w		0.0005 √	0.1294	0.0010 √	0.0010 √
WAB	*w/t/l*			7/0/5	8/0/4	9/1/2
	p_s			0.7744	0.3877	0.0654 ×
	p_w			0.9268	0.0732 ×	0.0107 √
ND	*w/t/l*				6/2/4	11/0/1
	p_s				0.7539	0.0063 √
	p_w				0.3359	0.0015 √
sBoost-1	*w/t/l*					10/2/0
	p_s					0.0020 √
	p_w					0.0020 √

Table 3.5. Statistical significant comparisons of five boosting techniques with 10% noise over 12 data sets.

Methods		AB	WAB	ND	sBoost-1	sBoost-2
Mean		0.2745	0.2622	0.2668	0.2574	0.2541
AB	*w/t/l*		12/0/0	9/0/3	12/0/0	12/0/0
	p_s		0.0005 √	0.1460	0.0005 √	0.0005 √
	p_w		0.0005 √	0.1455	0.0005 √	0.0005 √
WAB	*w/t/l*			7/0/5	8/3/1	11/0/1
	p_s			0.7744	0.0391 √	0.0063 √
	p_w			0.8057	0.0703 ×	0.0176 √
ND	*w/t/l*				8/1/3	10/0/2
	p_s				0.2266	0.0386 √
	p_w				0.1533	0.0210 √
sBoost-1	*w/t/l*					11/1/0
	p_s					0.0010 √
	p_w					0.0010 √

3.4 Summary

A selective boosting technique, sBoost, has been developed in this chapter for more efficient pattern classification. The sBoost can adaptively adjust the effect of noisy samples on the data distribution using the proposed sample weight regulator, in order to reduce the overfitting problem. An error correction method is suggested in sBoost to detect and process potential clean samples in the test data to further improve the classification accuracy. The effectiveness of the developed sBoost classifier was verified by simulation tests using a series of benchmark data sets that are commonly used in this research field. Test results showed that the developed sBoost was an efficient classification tool. It can effectively conduct pattern classification for different applications.

The work presented in this chapter is to achieve the research objective in Section 1.3: *Develop a pattern classification technique to effectively categorize the representative features from the selected technique(s) to provide a more positive assessment of IM health condition.*

Chapter 4
A Model-Based pBoost Predictor

In this chapter, a novel model-based boosting technique, pBoost, is developed to forecast future states of the IM health condition, in order to further enhance the diagnostic reliability [116]. The effectiveness of the proposed pBoost technique will be examined by using simulation tests and real applications.

4.1 The pBoost Predictor

The idea in pBoost technique is to incorporate many AR predictors in the ensemble, each of which is assigned an appropriate weight; correspondingly, the ensemble predictor can outperform each base predictor and become a stronger predictor. The development of the technique is detailed below.

4.1.1 Sample Regulation

Consider the training data sets $z(k)$; k = 1, 2, …, K, where K is the number of samples in the training date set. For s-step-ahead prediction, the training date set can be re-arranged to have the input vector $x(i) = [z(i), z(i + 1), \ldots, z(i + d - 1)]$, and the output $y(i) = z(i + d + s - 1)$, where i = 1, 2, …, N, and $N = (K - d - s + 1)$, and d is the dimension of the input vector $x(i)$.

In the ensemble of the pBoost, the base predictor h_t incorporated at step t aims to classify the training data with distribution L_t. Given the distribution L_t, the update of the distribution at step t+1 will be performed by

$$L_{t+1}(i) = \frac{L_t(i)\exp\left(-\beta_t \left| y_d(i) - p_t(i) \right|\right)}{Z_t} \tag{4.1}$$

where y_d are the desired values, and p_t are the predicted values at step t using the base predictor h_t; β_t is the weight of the base predictor h_t; $Z_t = \sum_{i=1}^{N} L_t(i)\exp\left(-\beta_t \left| y_d(i) - p_t(i) \right|\right)$ is a normalization factor.

In Equation (4.1), the larger the error $|y_d(i) - p_t(i)|$, the smaller the sample penalty. The proposed pBoost method focuses on easy-to-learn samples, and gradually learns more difficult samples through AR model training. Thus the overfitting caused by hard-to-learn samples can be reduced.

By setting the initial distribution $L_1(i) = \frac{1}{N}$, i = 1, 2, …, N, the update of the distribution can also be carried out by

$$L_{t+1}(i) = \frac{\exp\left(-\sum_{t=1}^{T} \beta_t |y_d(i) - p_t(i)|\right)}{N \prod_{t=1}^{T} Z_t} \tag{4.2}$$

The weight β_t of the base predictor h_t will be

$$\beta_t = \frac{1}{2M_t} \ln\left(\frac{M_t + \lambda_t}{M_t - \lambda_t}\right) \tag{4.3}$$

where $\lambda_t = \sum_{i=1}^{N} L_t(i) |y_d(i) - p_t(i)|$; $M_t = \sup|y_d(i) - p_t(i)|$ is the maximum value of $|y_d(i) - p_t(i)|$. It can be proven that $\lambda_t \geq 0$, $\beta_t \geq 0$. The detailed derivation of β_t will be discussed in the Subsection 4.1.2.

The final ensemble predictor is derived as

$$P = \sum_{t=1}^{T} \beta_t' p_t \tag{4.4}$$

where $\beta_t' = \frac{\beta_t}{\sum_{t=1}^{T} \beta_t}$ is the normalized weight of base predictors. The processing diagram of pBoost is illustrated in Figure 4.1.

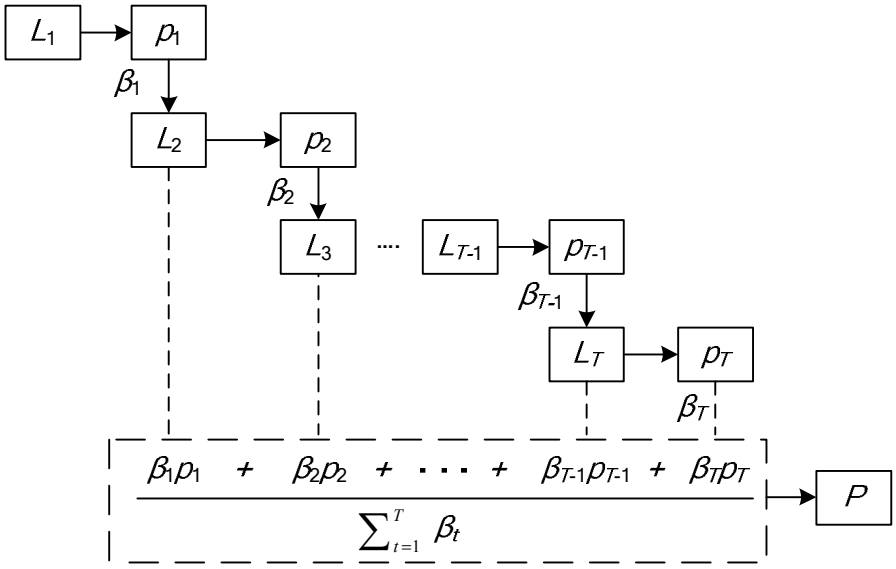

Figure 4.1. Processing diagram of the pBoost.

4.1.2 Derivation of Mean Absolute Error

Given $\lambda_t \geq 0$, $\beta_t' \geq 0$, $\beta_t \geq 0$, $\sum_{t=1}^{T} \beta_t' = 1$, $\sum_{i=1}^{N} L_{T+1}(i) = 1$, and the final ensemble prediction $P(i) = \sum_{t=1}^{T} \beta_t' p_t(i)$, the mean absolute error (MAE) of the training data can be determined by

$$MAE = \frac{1}{N} \sum_{i=1}^{N} \left| y_d(i) - P(i) \right| \tag{4.5}$$

$$= \frac{1}{N} \sum_{i=1}^{N} \left| y_d(i) - \sum_{t=1}^{T} \beta_t' p_t(i) \right| \tag{4.6}$$

$$= \frac{1}{N} \sum_{i=1}^{N} \left| \sum_{t=1}^{T} \beta_t' \left(y_d(i) - p_t(i) \right) \right| \tag{4.7}$$

$$\leq \frac{1}{N} \sum_{i=1}^{N} \sum_{t=1}^{T} \left| \beta_t' \left(y_d(i) - p_t(i) \right) \right| \tag{4.8}$$

$$\leq \frac{1}{N} \sum_{i=1}^{N} \sum_{t=1}^{T} \left| \beta_t \left(y_d(i) - p_t(i) \right) \right| \tag{4.9}$$

$$\leq \frac{1}{N}\sum_{i=1}^{N}\exp\left(-\sum_{t=1}^{T}\left|\beta_t\left(y_d(i)-p_t(i)\right)\right|\right) \tag{4.10}$$

$$= \frac{1}{N}\sum_{i=1}^{N}\exp\left(-\sum_{t=1}^{T}\beta_t\left|y_d(i)-p_t(i)\right|\right) \tag{4.11}$$

$$= \sum_{i=1}^{N}L_{T+1}(i)\prod_{t=1}^{T}Z_t \tag{4.12}$$

$$= \prod_{t=1}^{T}Z_t \tag{4.13}$$

To satisfy inequality (4.9), $\beta_t' \leq \beta_t$ or $\sum_{t=1}^{T}\beta_t \geq 1$ should be satisfied. If T is sufficiently large, then $\sum_{t=1}^{T}\beta_t \geq 1$. The inequality in Equation (4.10) can be simplified as

$$x \leq \exp(-x) \tag{4.14}$$

where $x = \sum_{t=1}^{T}\left|\beta_t\left(y_d(i)-p_t(i)\right)\right| = \sum_{t=1}^{T}\beta_t\left|y_d(i)-p_t(i)\right|$.

To satisfy Equation (4.14), x can be constrained within the range [0, 0.567] as demonstrated in Figure 4.2.

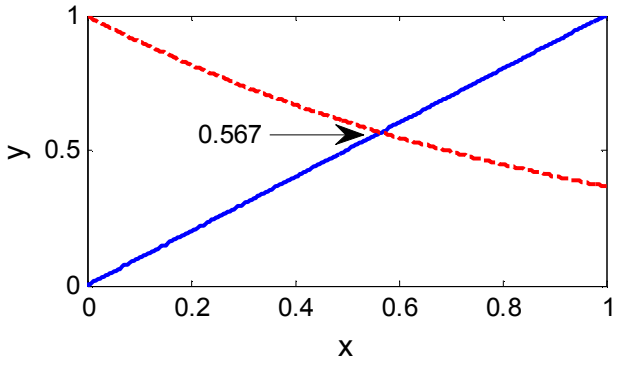

Figure 4.2. Plot of $y = x$ (blue solid line) and $y = e^{-x}$ (red dashed line).

Then $x = \sum_{t=1}^{T} \beta_t \left| y_d(i) - p_t(i) \right|$ should lie within the range [0, 0.567], which can be achieved by properly scaling the training data. Therefore, the minimization of the prediction MAE is equivalent to minimizing $\prod_{t=1}^{T} Z_t$, or minimizing Z_t at each step t. Given the initialization distribution $L_1(i) = \frac{1}{N}$, $Z_t = \sum_{i=1}^{N} L_t(i) \exp\left(-\beta_t \left| y_d(i) - p_t(i) \right|\right)$ will remain positive.

The following two constraints need to be satisfied,

$$\sum_{t=1}^{T} \beta_t \geq 1 \tag{4.15}$$

$$\sum_{t=1}^{T} \beta_t \left| y_d(i) - p_t(i) \right| \in [0, 0.567]. \tag{4.16}$$

Constraint (4.15) can be achieved by scaling $\left| y_d(i) - p_t(i) \right|$. β_t is immune to scaling of $\left| y_d(i) - p_t(i) \right|$, which will be shown in the following subsection. Thus constraints (4.15) and (4.16) can be satisfied simultaneously.

4.1.3 MAE Convergence

Let $u_t = \left| y_d(i) - p_t(i) \right|$ with $u_t \in [0, M_t]$, then $\lambda_t = \sum_{i=1}^{N} L_t(i) u_t(i)$. The upper bound of Z_t can be derived as

$$\begin{aligned} Z_t &= \sum_{i=1}^{N} L_t(i) \exp\left(-\beta_t u_t(i)\right) \\ &\leq \sum_{i=1}^{N} L_t(i) \left(\frac{M_t + u_t(i)}{2M_t} \exp(-\beta_t M_t) + \frac{M_t - u_t(i)}{2M_t} \exp(\beta_t M_t) \right) \\ &= \frac{1}{2} \exp(-\beta_t M_t) + \frac{\lambda_t}{2M_t} \exp(-\beta_t M_t) + \frac{1}{2} \exp(\beta_t M_t) - \frac{\lambda_t}{2M_t} \exp(\beta_t M_t) \end{aligned} \tag{4.17}$$

Let $U = \frac{1}{2}\exp(-\beta_t M_t) + \frac{\lambda_t}{2M_t}\exp(-\beta_t M_t) + \frac{1}{2}\exp(\beta_t M_t) - \frac{\lambda_t}{2M_t}\exp(\beta_t M_t)$; the minimization of the upper bound U with respect to β_t can be achieved by setting $\frac{\partial U}{\partial \beta_t} = 0$. By re-arranging $\frac{\partial U}{\partial \beta_t} = 0$,

$$\beta_t = \frac{1}{2M_t}\ln\left(\frac{M_t + \lambda_t}{M_t - \lambda_t}\right) \tag{4.18}$$

Substituting Equation (4.18) into Equation (4.17), the minimum upper bound of Z_t can be derived as

$$Z_t \le \sqrt{1 - \left(\lambda_t / M_t\right)^2} \tag{4.19}$$

Since $M_t \ge \lambda_t \ge 0$, then $\sqrt{1 - \left(\lambda_t / M_t\right)^2} \le 1$. The upper bound of MAE of the training data can be derived as

$$MAE \le \prod_{t=1}^{T} Z_t \le \prod_{t=1}^{T} \sqrt{1 - \left(\lambda_t / M_t\right)^2} \tag{4.20}$$

Therefore, the more base predictors h_t that are included in the ensemble predictor (or the larger the T), the lower the upper bound of the training MAE will be.

4.1.4 AR Model as Base Predictor

The AR models will be used as base predictors in the proposed pBoost technique because of their simple modeling mechanism, high training efficiency, and ease of implementation. To associate the distribution of training data with the base predictors (i.e., the AR models), these models are trained with the WLSE. The AR model has the form of

$$p_t(i) = \theta_1 z(i-1) + \theta_2 z(i-2) + ... + \theta_{r-1} z(i-r+1) \tag{4.21}$$

where $\{\theta_i\}$ are linear parameters; i = 1, 2, …, r - 1. Equation (4.21) can also be written in the following matrix form,

$$Y = X\theta \tag{4.22}$$

where

$$X = \begin{bmatrix} z(1) & z(2) & \cdots & z(r-1) \\ z(2) & z(3) & \cdots & z(r) \\ \vdots & \vdots & \ddots & \vdots \\ z(N_t - r - s + 1) & z(N_t - r - s + 2) & \cdots & z(N_t - s - 1) \end{bmatrix} \tag{4.23}$$

$$\theta = \begin{bmatrix} \theta(1) \\ \theta(2) \\ \vdots \\ \theta(r-1) \end{bmatrix} \tag{4.24}$$

$$Y = \begin{bmatrix} z(r+s) \\ z(r+s+1) \\ \vdots \\ z(N_t) \end{bmatrix} \tag{4.25}$$

where N_t is the number of system state z. The linear parameters θ of AR models can be trained using the WLSE method [46]:

$$\theta = (X^T W X)^{-1} X^T W Y \tag{4.26}$$

The weight matrix W is represented as

$$W = \begin{bmatrix} L_t(1)^q & \cdots & 0 & 0 \\ \vdots & L_t(2)^q & \cdots & 0 \\ 0 & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & L_t(N)^q \end{bmatrix} \tag{4.27}$$

where $N = N_t - r - s + 1$, and q is a strength factor to characterize the degree of influence to which the distribution $L_t(i)$ has on each training sample.

By adding more AR base predictors to the ensemble, the upper bound of training MAE as well as the training MAE decrease as demonstrated in Equation (4.20). Consequently, the prediction accuracy can be improved by using the ensemble of AR models.

In the boosting algorithms such as the Adaboost method, large penalties are imposed on hard-to-learn samples to ensure that these samples can be learned with more effort. Nevertheless, this will sacrifice the learning accuracy of low weight samples, resulting in the overfitting problem. The overfitting problem easily occurs when relatively strong base learners rather than weak base learners are used in the ensemble. In the proposed pBoost as shown in Equation (4.1), the larger the error is, the lesser the penalty will be assigned to the sample. The proposed pBoost will emphasize easy-to-learn samples, and gradually learn harder-to-learn samples as the capability of the ensemble learner becomes stronger, or as more base learners are incorporated in the ensemble. This sample regulation method can reduce the overfitting problem, especially when relatively strong base learners (i.e., AR models) are used in the ensemble.

4.1.5 Implementation of pBoost Technique

The implementation procedure of the pBoost predictor is summarized as follows:

1) Scale the data within a proper range (e.g., [0, 1]), so that constraints in subsection 4.1.2 can be satisfied.

2) Initialize the distribution of the training data set $L_1(i) = \frac{1}{N}$; $i = 1, 2, \ldots, N$.

3) Train a base predictor h_t (i.e., AR model trained by the WLSE) using training data set with the distribution L_t.

4) Compute the sum of weighted absolute error $\lambda_t = \sum_{i=1}^{N} L_t(i)\left|y_d(i) - p_t(i)\right|$, where y_d are the desired values, and p_t are the predicted values at step t using the base predictor h_t.

5) Calculate the weight of the base predictor h_t, $\beta_t = \frac{1}{2M_t}\ln\left(\frac{M_t + \lambda_t}{M_t - \lambda_t}\right)$, where M_t is the maximum value of $\left|y_d(i) - p_t(i)\right|$.

6) Update the distribution of the training samples, $L_{t+1}(i) = \frac{L_t(i)\exp\left(-\beta_t\left|y_d(i) - p_t(i)\right|\right)}{Z_t}$, where $Z_t = \sum_{i=1}^{N} L_t(i)\exp\left(-\beta_t\left|y_d(i) - p_t(i)\right|\right)$ is a normalization factor.

7) Repeat steps 3) to 6) as $t = 1, 2, \ldots, T$.

8) Formulate the final ensemble predictor $P = \sum_{t=1}^{T} \beta_t' p_t$, where $\beta_t' = \frac{\beta_t}{\sum_{t=1}^{T}\beta_t}$ are normalized weights of base predictors, $t = 1, 2, \ldots, T$.

4.2 Performance Evaluation and Application

To verify the effectiveness of the proposed pBoost predictor, simulation tests are conducted to evaluate its accuracy. Two related predictors are used for comparison: the AdaBoost.R2 predictor [117], and the AdaBoost.RT predictor [118]. The three boosting techniques, AdaBoost.R2, AdaBoost.RT, and pBoost, will use 50 base predictors (AR models in this case). All the AR base predictors are trained by the WLSE. The evaluations are conducted by comparing the error convergence and prediction accuracy of the three boosting techniques. To satisfy constraint conditions as stated in Subsection 4.1.2, the data sets used in this section are scaled to be within the range of [0, 1] in computation. Test results, however, are still shown in their original scale.

4.2.1 Mackey-Glass Data Forecasting

The Mackey-Glass data set [119] is commonly used in the forecasting research field to compare the performance of predictors, due to its specific properties such as chaotic, non-periodic and non-convergence; it is given by:

$$\frac{dx(t)}{dt} = \frac{0.2x(t-\tau)}{1+x^{10}(t-\tau)} - 0.1x(t) \tag{4.28}$$

In this simulation test, a data set is obtained from Equation (4.28) with the initial conditions of τ = 30, $x(0)$ = 1.2, dt = 1 and $x(t) = 0$ for $t < 0$. 500 samples are selected for training, 250 samples for validation, and 250 samples for testing. q = 0.7 and r = 15 are selected in pBoost using cross validation by fixing T = 50. Five-step-ahead prediction is conducted in this Mackey-Glass data tests.

The MAE and root mean square error (RMSE) of these three predictors with respect to the different number of base predictors, are shown in Figures 4.3 and 4.4, respectively. The validation errors and test errors using 50 base predictors are summarized in Table 4.1. It is seen from Figures 4.3 and 4.4 that all three predictors have the same MAE and RMSE error when only one base predictor is used, which is used as the reference of comparison. However, the pBoost outperforms both AdaBoost.R2 and AdaBoost.RT as more base predictors are incorporated in the ensemble. The errors of AdaBoost.R2 and AdaBoost.RT increase as shown in Figures 4.3 and 4.4, because of their ineffective ensemble construction with base predictors. It can be observed that the errors of AdaBoost.R2 remain constant when the number of base predictors is larger than 4; this is due to the fact that the updated data distribution formulated by AdaBoost.R2 has insignificant impact on the AR model training. Since AdaBoost.R2 selects the weighted median as output of the ensemble, the ensemble output will remain unchanged if it always selects those AR models with close predictions. The AdaBoost.RT formulates its ensemble output as the weighted average of base predictors' outputs, and consequently the error of AdaBoost.RT grows slightly as more predictors are involved, because its ensemble learning mechanism cannot effectively process AR models as base predictors. Thus, both AdaBoost.R2 and AdaBoost.RT cannot benefit from increasing the number of AR predictors. The pBoost, on the other hand, is able to reduce both the MAE and RMSE as more AR base predictors are incorporated in the ensemble.

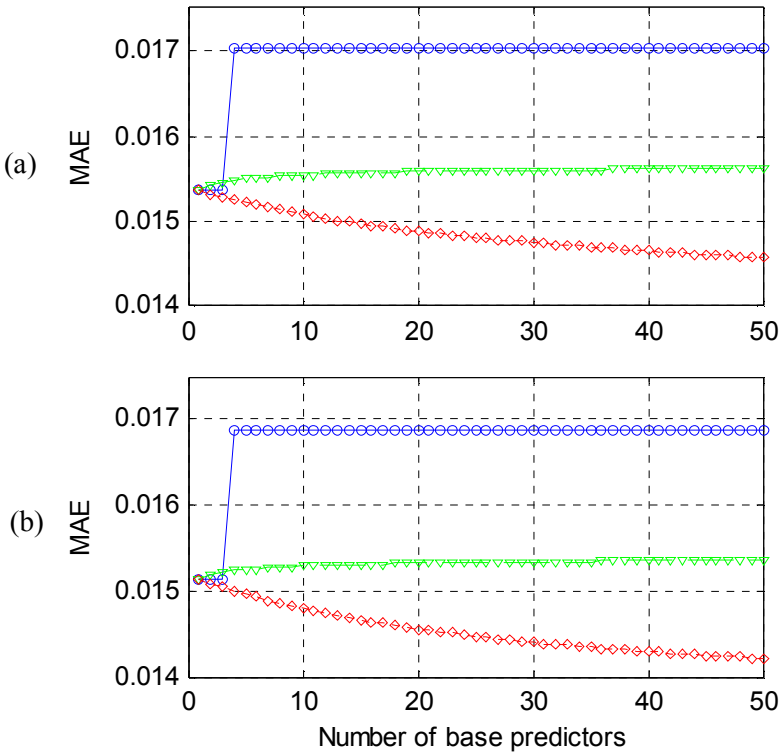

Figure 4.3. The convergence comparison using a Mackey-Glass data for five-step-ahead prediction: (a) validation MAE; (b) test MAE. The blue-circle-solid line represents errors of the AdaBoost.R2; the green-triangle-dashed line represents the errors of the AdaBoost.RT; the red-diamond-dotted line represents the errors of the pBoost.

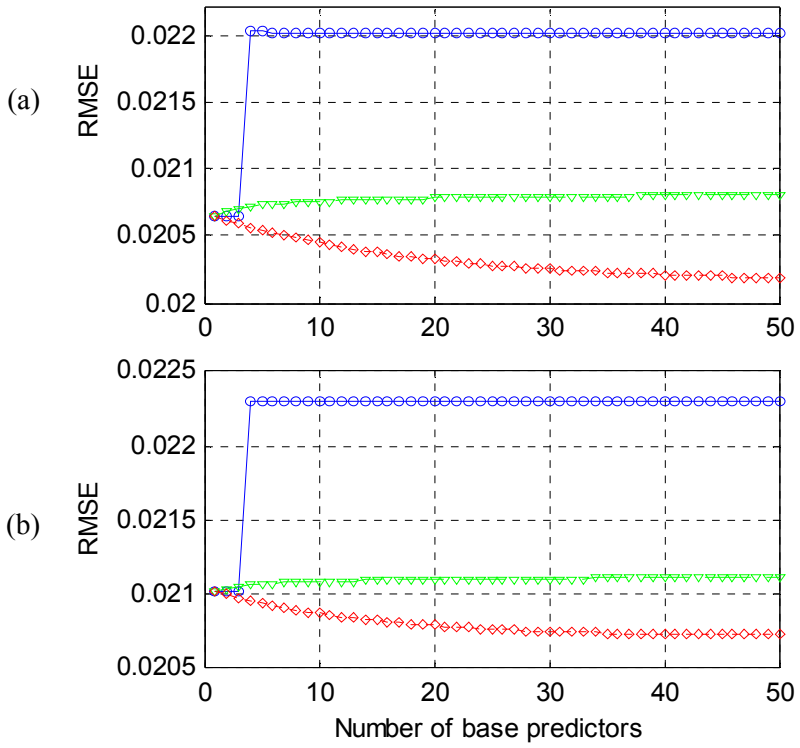

Figure 4.4. The convergence comparison using a Mackey-Glass data for five-step-ahead prediction: (a) validation RMSE; (b) test RMSE. The blue-circle-solid line represents errors of the AdaBoost.R2; the green-triangle-dashed line represents the errors of the AdaBoost.RT; the red-diamond-dotted line represents the errors of the pBoost.

Table 4.1. The five-step-ahead prediction performance of three predictors using 50 base predictors in terms of Mackey-Glass data.

Predictors	AdaBoost.R2 (10^{-2})	AdaBoost.RT (10^{-2})	pBoost (10^{-2})
Validation MAE	1.703	1.562	1.457
Validation RMSE	2.202	2.080	2.018
Test MAE	1.686	1.536	1.421
Test RMSE	2.229	2.111	2.072

To get some insight into the effect of the order of AR, r, and the strength factor q, on the performance of pBoost, simulation tests are also conducted using the Mackey-Glass data. Figure 4.5 shows MAEs of the validation data prediction, with respect to the order of AR models r, the number of base predictors T, and the factor q. The maximum number of base predictors T is set at 100. It is seen from Figure 4.5 that the MAE decreases as the number of base predictors increases. For the same order r of the AR model, the test MAE converges faster with a greater q. The pBoost with AR(15) as base predictor generates lower MAE when compared to AR(10) and AR(20).

Figure 4.6 illustrates the corresponding RMSE of validation data prediction under the same conditions as those in Figure 4.5. With the exception of $q = 0.1$, regardless of r, RMSE decreases first as the number of predictors increases, and then increases. q is the rate of error changes (decrease and increase): the higher the q factor is, the faster the change will occur. In this test, minimum RMSE could be achieved with $q = 0.7$, $r = 15$ when $T = 50$, as shown in Figure 4.6(b).

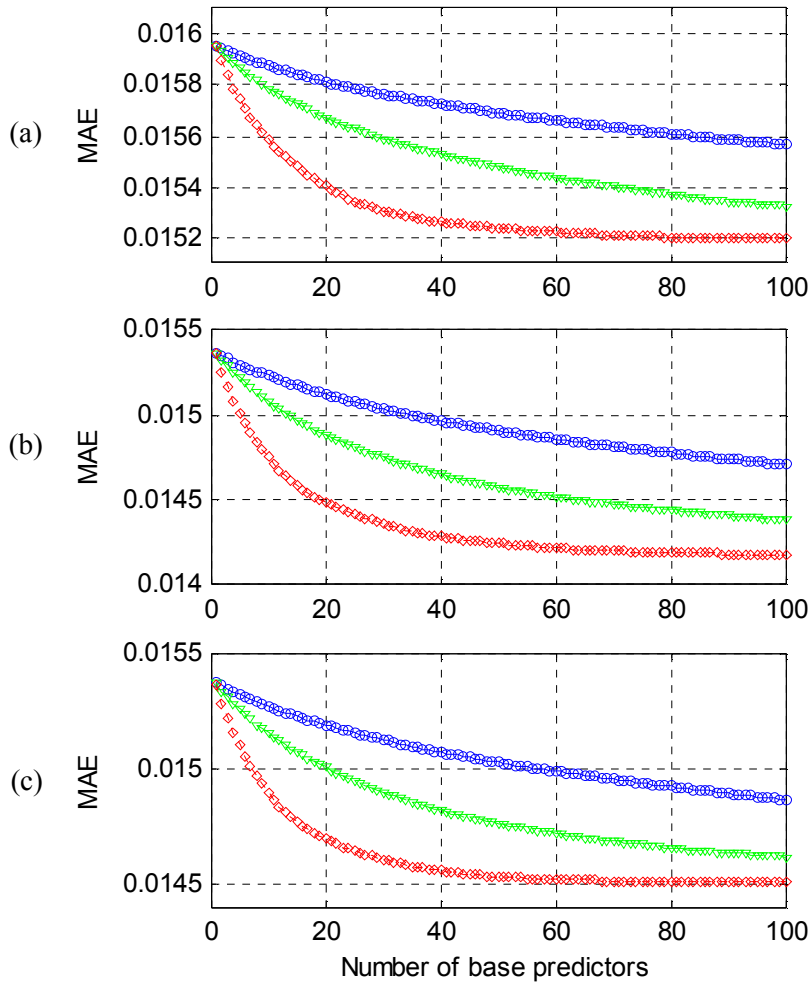

Figure 4.5. MAE of the five-step-ahead Mackey-Glass data prediction using pBoost predictor with respect to (a) AR(10); (b) AR(15); (c) AR(20), with $q = 0.1$ (blue-circle-solid lines), $q = 0.7$ (green-triangle-dashed lines), and $q = 2$ (red-diamond-dotted lines).

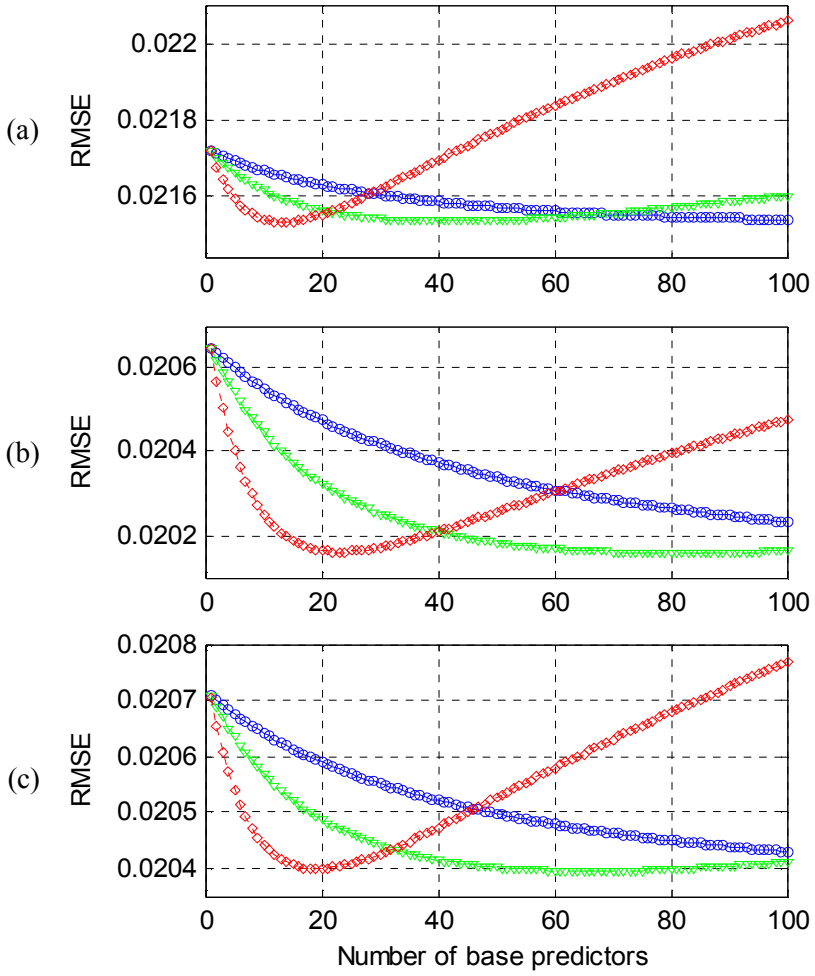

Figure 4.6. RMSE of the five-step-ahead prediction of Mackey-Glass data using pBoost predictor with respect to (a) AR(10); (b) AR(15); (c) AR(20), with q = 0.1 (blue-circle-solid lines), q = 0.7 (green-triangle-dashed lines), and q = 2 (red-diamond-dotted lines).

4.2.2 Sunspot Data Forecasting

Sunspot activity record is another benchmark data set that is commonly used to evaluate predictors in the field of forecasting research and development. 265 data sets are used for testing in this case. The first 200 sunspot data (from years 1749 to 1948) are used for training, 30 data for validation (from years 1949 to 1978) and 35 data (from years 1979 to 2013) for testing. Two-step-ahead prediction is performed for sunspot data forecasting.

Figures 4.7 and 4.8 show the respective MAE and RMSE of the three predictors with $T = 50$, $r = 14$ and $q = 4$, where r and q are selected using cross validation by fixing $T = 50$. Table 4.2 lists the validation errors and test errors of the predictors, when 50 base predictors are used. From Figure 4.7 and Figure 4.8, it can be seen that the pBoost and AdaBoost.RT can decrease the errors; however, the pBoost outperforms the AdaBoost.RT due to its more effective boosting approach using the AR base predictors.

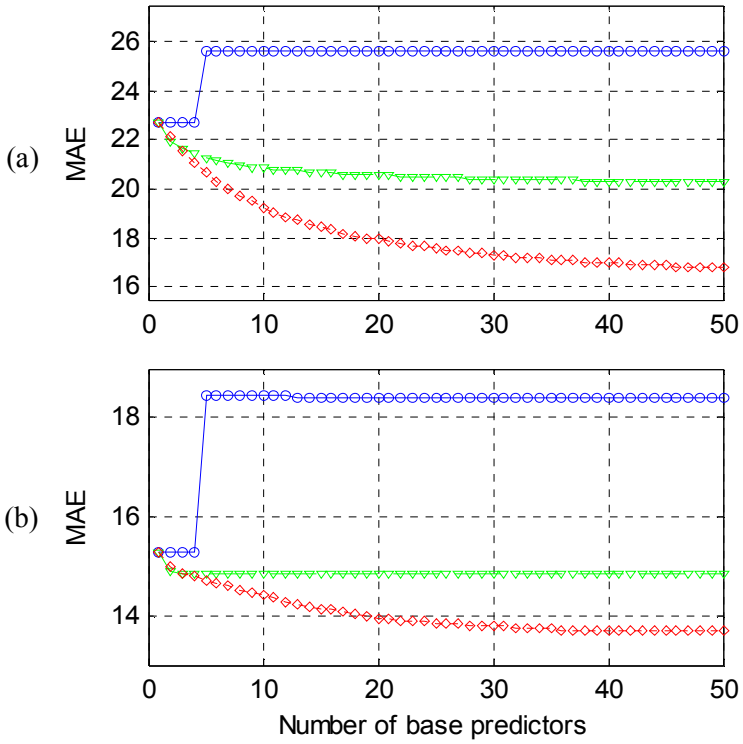

Figure 4.7. The convergence comparison using sun spot data for two-step-ahead prediction: (a) validation MAE; (b) test MAE. The blue-circle-solid line represents errors of the AdaBoost.R2; the green-triangle-dashed line represents the errors of the AdaBoost.RT; the red-diamond-dotted line represents the errors of the pBoost.

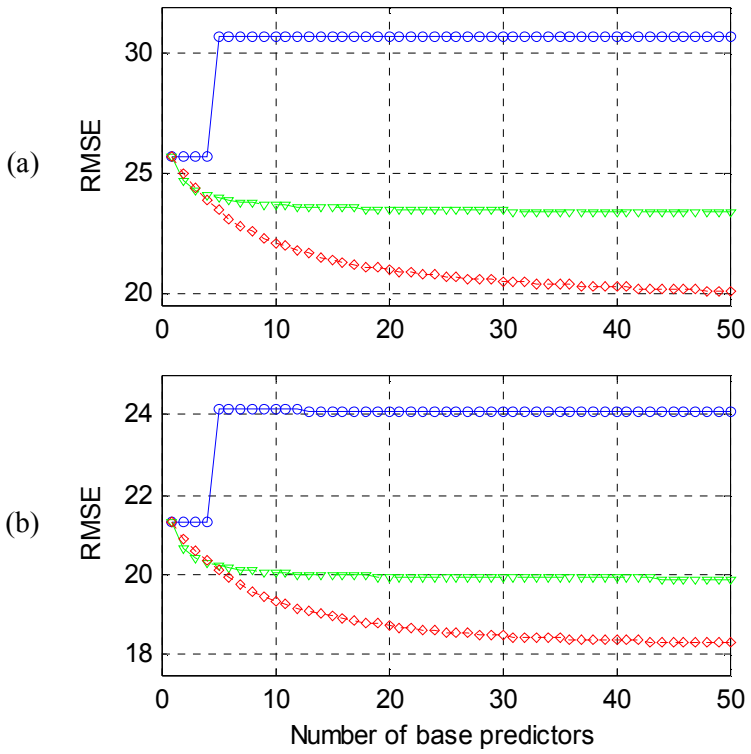

Figure 4.8. The convergence comparison using sunspot data for two-step-ahead prediction: (a) validation RMSE; (b) test RMSE. The blue-circle-solid line represents errors of the AdaBoost.R2; the green-triangle-dashed line represents the errors of the AdaBoost.RT; the red-diamond-dotted line represents the errors of the pBoost.

Table 4.2 The two-step-ahead prediction performance of three predictors using 50 base predictors in terms of Sunspot data.

Predictors	AdaBoost.R2	AdaBoost.RT	pBoost
Validation MAE	25.634	20.260	16.775
Validation RMSE	30.742	23.364	20.073
Test MAE	18.422	14.834	13.696
Test RMSE	24.091	19.894	18.296

4.2.3 Material Fatigue Prognosis

The proposed pBoost predictor is implemented in this work for material fatigue prognosis. Since material fatigue testing is usually time-consuming, an efficient predictor is extremely helpful to estimate the material's dynamic characteristics without undertaking full-scale fatigue testing. For illustration, an aluminum specimen with a thickness of 3 mm was tested using the experimental setup shown in Figure 4.9. Both ends of the specimen were clamped to the testing machine. A small hole was made in the middle of the specimen to facilitate crack propagation. The crack dimension was measured directly by calipers and indirectly using relative voltage measurement. The measurement was taken every 3000 load cycles. The test presented here is based on an indirect measurement.

Figure 4.9. The experimental setup for material fatigue testing.

In material fatigue testing, experiments were conducted under four different testing conditions corresponding to initial crack dimensions. Under each testing condition, the experiment was repeated 6 times to yield six data sets. One data set was randomly selected from these generated data sets. The first 825 data samples of the data set were used for training, 425 data samples for validation, and 400 data samples for testing. Twenty-step-ahead prediction was conducted for material property forecasting, which was also used to compare the performance of the related predictors.

The MAE and RMSE of the predictors against the number of base predictors are shown in Figures 4.10 and 4.11 respectively, with $T = 50$, $r = 20$, and $q = 7$; q and r are selected using cross validation by fixing $T = 50$. The corresponding validation errors and test errors of the three predictors using 50

base predictors are summarized in Table 4.3. It is seen from Figures 4.10 and 4.11 that both AdaBoost.R2 and AdaBoost.RT cannot reduce the MAE and RMSE when additional base predictors (i.e., AR models) are used. This deficiency exists due to ineffective ensemble construction with the base predictors. The pBoost, on the other hand, can improve its performance (i.e., decreasing the errors) by incorporating more AR models in the ensemble. Figure 4.12 demonstrates the prognosis performance of these three predictors with 50 base predictors. The test results are listed in Table 4.3. It is seen that the pBoost outperforms the other two predictors, and has the ability to track the dynamics of the material fatigue property more accurately.

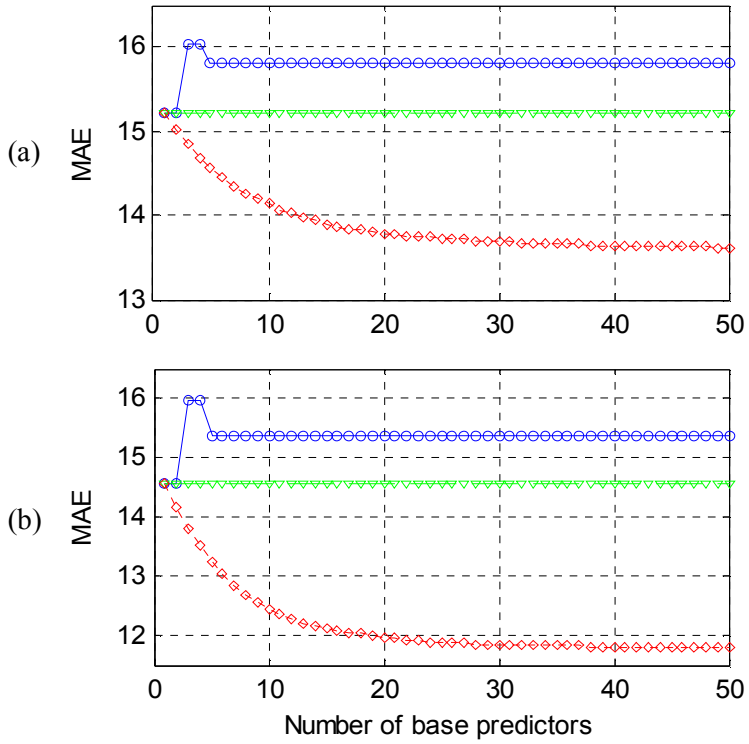

Figure 4.10. The convergence comparison of the related predictors for material fatigue twenty-step-ahead prognosis: (a) validation MAE; (b) test MAE. The blue-circle-solid line represents errors of the AdaBoost.R2; the green-triangle-dashed line represents the errors of the AdaBoost.RT; the red-diamond-dotted line represents the errors of the pBoost.

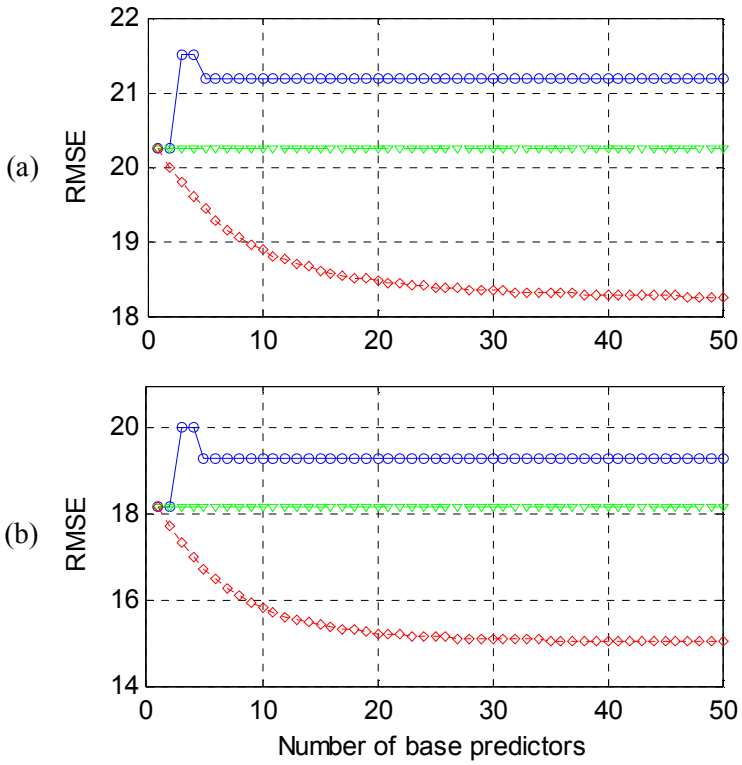

Figure 4.11. The convergence comparison of the related predictors for material fatigue twenty-step-ahead prognosis: (a) validation RMSE; (b) test RMSE. The blue-circle-solid line represents errors of the AdaBoost.R2; the green-triangle-dashed line represents the errors of the AdaBoost.RT; the red-diamond-dotted line represents the errors of the pBoost.

Table 4.3. The twenty-step-ahead prediction performance of three predictors using 50 base predictors in terms of material fatigue data.

Predictors	AdaBoost.R2	AdaBoost.RT	pBoost
Validation MAE	15.819	15.219	13.619
Validation RMSE	21.188	20.231	18.250
Test MAE	15.370	14.552	11.806
Test RMSE	19.315	18.162	15.033

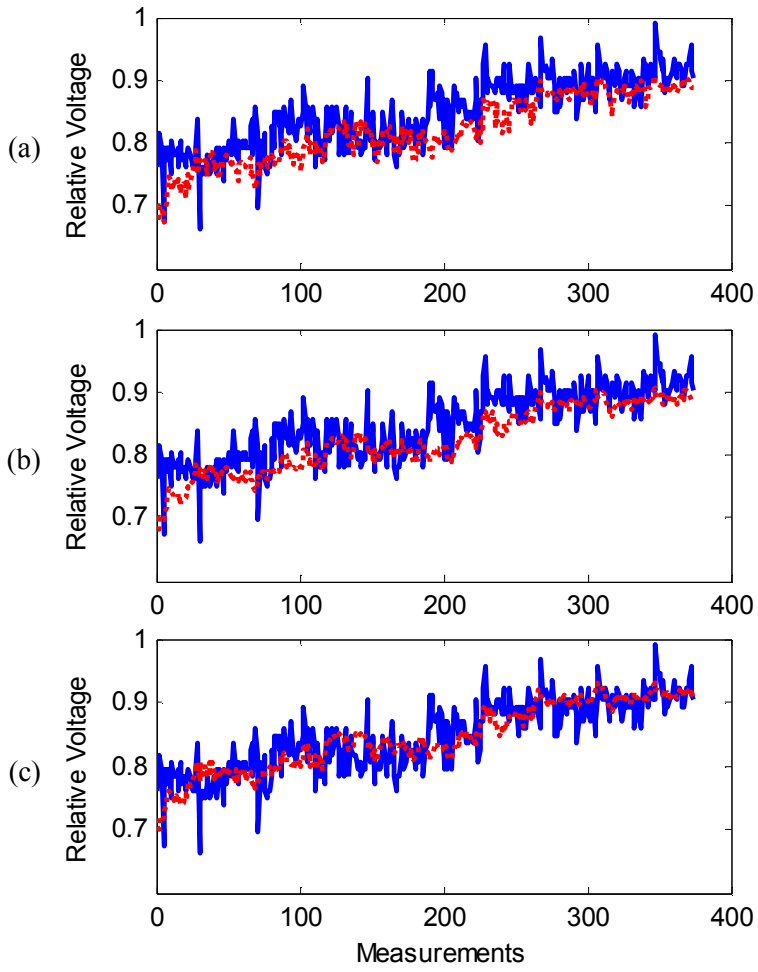

Figure 4.12. The twenty-step-ahead prediction performance of (a) the AdaBoost.R2; (b) AdaBoost.RT; and (c) the pBoost for material fatigue prognosis. The blue solid line represents the real data and the red dotted line represents the predicted values.

4.3 Summary

A new boosting technique, pBoost, has been developed in this chapter for system state prediction. The pBoost can aggregate many base predictors such as AR models, to improve forecasting performance. Different base predictors address the training data with different distributions, and the distribution of training data at each step reflects the prediction performance of the ensemble predictor consisting of all previous base predictors. The effectiveness of the proposed pBoost predictor was

verified using simulations of two benchmark data sets. The pBoost predictor was also implemented for material fatigue prognosis. Test results showed that the pBoost predictor can effectively formulate the ensemble of base predictors to improve forecasting accuracy. It is able to capture the dynamic behavior of the tested system effectively and track its future states accurately. The verification of the proposed pBoost predictor for IM fault indices prediction will be discussed in Chapter 6.

The work presented in this chapter is to achieve the research in Section 1.3: *Develop a model-based predictor to forecast future states of the IM conditions. The forecast information can be used to further improve the reliability of IM condition monitoring.*

Chapter 5
A Data-Driven Fuzzy Neural Network Predictor

As an alternative solution to the model-based pBoost predictor discussed in Chapter 4, a knowledge-based data-driven predictor, using a new evolving fuzzy neural network, eFNN, is developed in this chapter [120]. The effectiveness of the proposed eFNN technique will be verified by using simulation tests and real applications.

5.1 The Evolving Fuzzy Neural Predictor

The developed eFNN predictor applies both linear and nonlinear modeling strategies to characterize properties of multi-dimensional data sets. A novel cumulative clustering algorithm is proposed to evolve fuzzy reasoning rules for nonlinear modeling. The development of the technique is detailed below.

5.1.1 Architecture of the eFNN Predictor

Figure 5.1 describes the network architecture of the proposed eFNN predictor. It is a six-layer feed-forward network. Layer 1 is the input layer. Each input $\lfloor x_{i,t}, x_{i,t-s}, ..., x_{i,t-(p-1)s} \rfloor$ represents a vector from the i^{th} data set with time lags 0 to p-1; s is the time-step; p is the dimension of the input data vector; i = 1, 2,…, m; and m is the dimension of the multiple data sets or the number of inputs to the eFNN predictor.

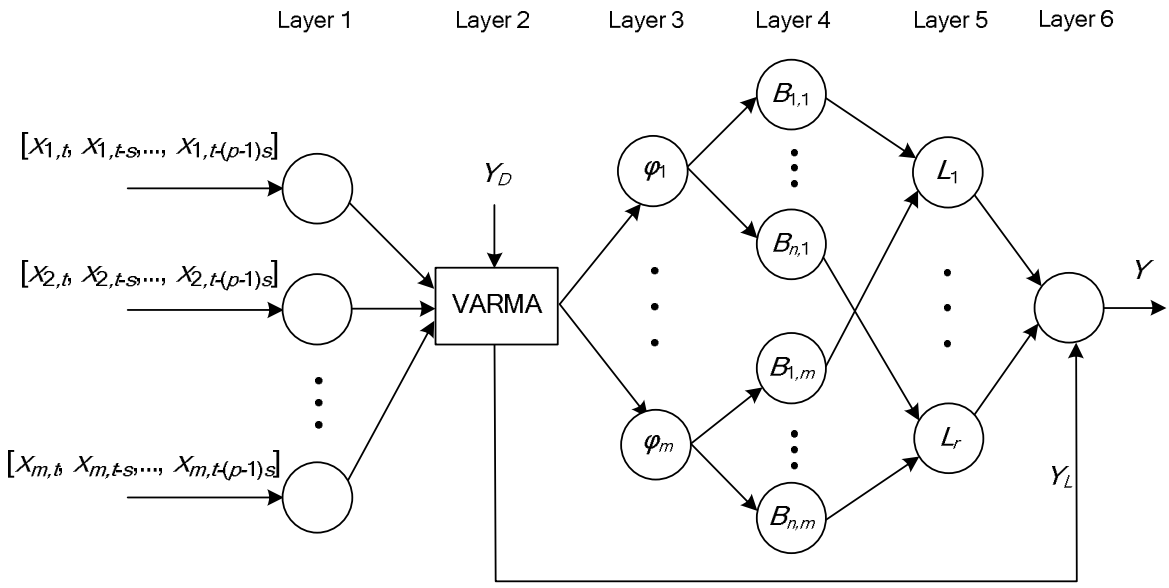

Figure 5.1. Architecture of the eFNN predictor.

In most forecasting applications, there exist both linear and nonlinear correlations between the target data sets and these available data sets. Layer 2 performs VARMA (vector autoregressive-moving-average) filtering to model the linear correlations among the data sets in each dimension. The classic VARMA model generates an output vector which contains m entries; the i^{th} entry corresponds to the predicted value of the i^{th} dimensional data set, $i = 1, 2, \ldots, m$. In terms of the i^{th} entry in the output vector, the VARMA filter can be expressed as:

$$Y_L = \mathbf{\Theta}_1 \begin{bmatrix} x_{1,t} \\ x_{2,t} \\ \vdots \\ x_{m,t} \end{bmatrix} + \mathbf{\Theta}_2 \begin{bmatrix} x_{1,t-s} \\ x_{2,\,t-s} \\ \vdots \\ x_{m,\,t-s} \end{bmatrix} + \cdots + \mathbf{\Theta}_p \begin{bmatrix} x_{1,t-(p-1)s} \\ x_{2,t-(p-1)s} \\ \vdots \\ x_{m,t-(p-1)s} \end{bmatrix} + \Lambda_1 \begin{bmatrix} \varphi_{1,t} \\ \varphi_{2,t} \\ \vdots \\ \varphi_{m,t} \end{bmatrix} + \Lambda_2 \begin{bmatrix} \varphi_{1,t-s} \\ \varphi_{2,\,t-s} \\ \vdots \\ \varphi_{m,\,t-s} \end{bmatrix} + \cdots + \Lambda_q \begin{bmatrix} \varphi_{1,t-(q-1)s} \\ \varphi_{2,t-(q-1)s} \\ \vdots \\ \varphi_{m,t-(q-1)s} \end{bmatrix} \quad (5.1)$$

where $\mathbf{\Theta}_k = \begin{bmatrix} \theta_{k,1} & \theta_{k,2} & \cdots & \theta_{k,m} \end{bmatrix}$ are the linear AR parameters, $k = 1, 2, \ldots, p$, and $\Lambda_l = \begin{bmatrix} \lambda_{l,1} & \lambda_{l,2} & \cdots & \lambda_{l,m} \end{bmatrix}$ are the linear MA parameters; $l = 1, 2, \ldots, q$. Y_L is the predicted value of the i^{th} dimensional data set using VARMA.

The linear filtering output Y_L is forwarded to the output node in Layer 6. $\varphi_{i,j}$ is the linear filtering error of the i^{th} dimensional data set at time instance j. To conduct an s-step-ahead forecasting, $\varphi_{i,j}$ can be determined as $\varphi_{i,j} = Y_D - Y_L$, where Y_D and Y_L are the desired and estimated values of the VARMA filter in the i^{th} dimensional data set at time instant j, respectively.

To simplify representation, the linear estimation (or filtering) errors $\{\varphi_{1,t}, \varphi_{2,t}, \cdots, \varphi_{m,t}\}$ are represented as $\{\varphi_1, \varphi_2, \cdots, \varphi_m\}$, which are the inputs in Layer 3.

Layer 4 is the fuzzy rule layer. Gaussian functions are selected as membership functions (MFs), $B_{j,i}$, to formulate the fuzzy operation. Given the input $\mathbf{\Phi} = [\varphi_1, \varphi_2, \cdots, \varphi_m]$, the firing strengths η_j can be derived by using the fuzzy product T-norm:

$$\eta_j = \exp\left(-\frac{1}{2} \frac{\left\| \mathbf{\Phi} - \boldsymbol{\mu}_j \right\|^2}{\sigma_j^2} \right)$$

$$= \exp\left(-\frac{1}{2} \frac{(\varphi_1 - \mu_{j,1})^2 + (\varphi_2 - \mu_{j,2})^2 + \cdots (\varphi_m - \mu_{j,m})^2}{\sigma_j^2} \right) = \prod_{i=1}^{m} B_{j,i}(\varphi_i) \quad (5.2)$$

and

$$B_{j,i}(\varphi_i) = \exp\left(-\frac{1}{2} \frac{(\varphi_i - \mu_{j,i})^2}{\sigma_j^2} \right) \quad (5.3)$$

where $\boldsymbol{\mu}_j = [\mu_{j,1}, \mu_{j,2}, ..., \mu_{j,m}]$ is the center of the j^{th} cluster, which will be derived using a clustering algorithm, as discussed in Subsection 5.1.2. $\mu_{j,i}$ (j = 1, 2,…, n, and i = 1, 2,…, m) and σ_j are the respective center and spread of the Gaussian MF $B_{j,i}$. n is the number of nodes for each input φ_i.

Layer 5 is the rule layer. Each node in this layer is formulated by

$$L_j = a_{j,1} x_1 + a_{j,2} x_2 + ... + a_{j,m} x_m + b_j \quad (5.4)$$

where L_j denotes a first order TS model, in which $a_{j,i}$ (j = 1, 2, …, n; i = 1, 2, …, m) are the linear parameters, and b_j is the bias in L_j.

Layer 6 is the output layer. The eFNN output Y is formulated as,

$$Y = Y_L + \frac{\sum_{j=1}^{n} \eta_j L_j}{\sum_{j=1}^{n} \eta_j} \tag{5.5}$$

The fuzzy rules in Layers 4 and 5 are generated by the use of an evolving clustering paradigm that will be discussed in Subsection 5.1.2.

5.1.2 The Adaptive Clustering Algorithm

A cumulative evolving clustering (CEC) algorithm is proposed in this work, to adaptively evolve the fuzzy reasoning rules (clusters) represented in Layers 4 and 5 in Figure 5.1. The inputs to the evolving fuzzy (EF) network are the linear estimation error vectors $\mathbf{\Phi} = [\varphi_1, \varphi_2, \cdots, \varphi_m]$, and the j^{th} fuzzy rule can be formulated as

$$R_j : \text{IF } (\varphi_1 \text{ is } B_{j,1}) \text{ and } (\varphi_2 \text{ is } B_{j,2}) \text{ and, …, and } (\varphi_m \text{ is } B_{j,m}), \text{ THEN } (Z \text{ is } L_h) \tag{5.6}$$

where j = 1, 2,…, n; n is the number of fuzzy rules generated; Z is the clustering index; h = 1, 2,…, r; and r is the number of first order TS models ($r \le n$). The generated j^{th} cluster is an m-dimensional cluster with center $\mathbf{C}_j = [C_{j,1}, C_{j,2}, \cdots, C_{j,m}]$, and radius R_j.

Figure 5.2 schematically illustrates the clustering process of the CEC algorithm. The normalized Euclidean distance between the new input $\mathbf{\Phi}$ and the center of the j^{th} cluster, $\mathbf{C}_j$, is defined as

$$d_j = \frac{1}{\sqrt{m}} \left\| \mathbf{\Phi} - \mathbf{C}_j \right\| \tag{5.7}$$

R_j' is the intermediate spread with $R_j' = \left(\frac{1}{2} + \frac{M - N_j}{2M}\right) R_j$, where M is the total number of input values for clustering, and N_j is the number of input values in cluster j.

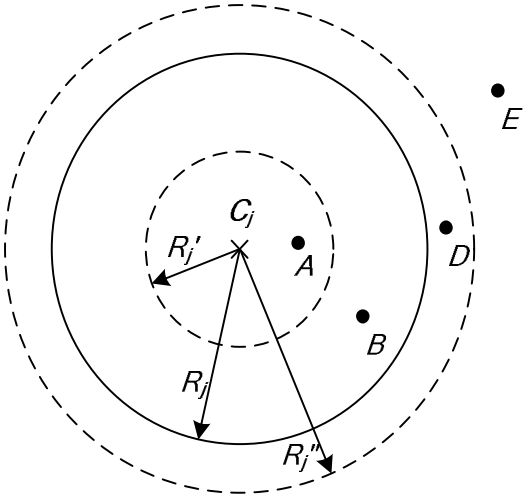

Figure 5.2. Schematic representation of a cluster j. C_j is the center and R_j is the radius of the jth cluster. R_j' and R_j'' are intermediate and extended radii of the jth cluster, respectively. *A*, *B*, *D* and *E* represent different states of clustering.

The CEC clustering process is summarized as follows:

Step 1: Initialization: When the first input $\mathbf{\Phi}$ (i.e., linear estimation error vector) is formulated after VARMA filtering, it becomes the center of the first cluster, $\boldsymbol{C}_1$. The initial radius of this cluster is set as R_0 ($R_0 = 0.01$, in this case). The upper boundary of the radius is denoted by R_U. If R_U is small, more clusters will be generated, and vice versa.

Step 2: Cluster formulation: If a new input $\mathbf{\Phi}$ falls in more than one cluster, only the center and radius of the closest cluster will be updated using the following rules:

a) If $d_j \le R_j'$ (e.g., state *A* in Figure 5.2), the center of the cluster is updated as:

$$\mathbf{C}_{j\_new} = \left(\mathbf{C}_{j\_old} \times N_j + \mathbf{\Phi}\right)/(N_j + 1) \tag{5.8}$$

where N_j is the number of input values in cluster *j*.

b) If $R_j' < d_j \le R_j$ (e.g., state *B* in Figure 5.2), the center and radius of this cluster remain unchanged.

c) Otherwise, if $R_j < d_j \le \left(2R_U - 2R_j'\right)$ (e.g., state *D* in Figure 5.2), the center of the cluster remains unchanged, but the radius R_j is updated as $R_j = R_j'' = \frac{1}{2} d_j + R'$.

d) If $R_j \le \left(2R_U - 2R_j'\right) < d_j$ (e.g., state *E* in Figure 5.2), a new cluster is created. The new input value $\mathbf{\Phi}$ becomes the center of the new cluster and the radius is initialized as R_0.

e) If $\left(2R_U - 2R_j'\right) < R_j < d_j$, a new cluster is created with the same setting as in d).

In general, the data with $d_j \le R_j'$ will have higher MF degree, and will be used to determine the cluster center. The data with $R_j < d_j \le \left(2R_U - 2R_j'\right)$ represent the potential spread of the cluster, and will be used to update the radius R_j of the cluster. If the input data satisfies $R_j' < d_j \le R_j$, the parameters of the cluster remain unchanged.

Step 3: Structure recognition: The center of the j^{th} cluster $\mathbf{C}_j$ will be the center of the j^{th} firing strength (i.e., $\boldsymbol{\mu}_j$ in Equation (5.2)). Assume that the center $\mathbf{C}_j$ corresponds to the MF degree 100% (i.e., $\eta_j = 1$), and the input values $\mathbf{\Phi}_R$ with $d_j = R_j$ are assigned a MF degree of 1% (i.e., $\eta_j = 0.01$).

Substituting $\eta_j = 0.01$ in Equation (5.2) yields

$$\exp\left(-\frac{1}{2}\frac{\left\|\mathbf{\Phi}_R - \boldsymbol{\mu}_j\right\|^2}{\sigma_j^2}\right) = 0.01 \tag{5.9}$$

Then inserting Equation (5.7) into Equation (5.9) gives the following equation:

$$\exp\left(-\frac{1}{2}\frac{mR_j^2}{\sigma_j^2}\right)=0.01 \tag{5.10}$$

The spread σ_j can be derived by rearranging Equation (5.10):

$$\sigma_j=\frac{\sqrt{m}\times R_j}{\sqrt{-2\ln(0.01)}} \tag{5.11}$$

When an input value with $d_j \le R_j$, its MF degree can be derived as

$$\begin{aligned}\eta_j &= \exp\left(-\frac{1}{2}\frac{\left\|\mathbf{\Phi}-\boldsymbol{\mu}_j\right\|^2}{\sigma_j^2}\right)\\ &= \exp\left(-\frac{1}{2}\frac{md_j^2}{\sigma_j^2}\right)\\ &= \exp\left[\left(\frac{d_j}{R_j}\right)^2\ln(0.01)\right]\end{aligned} \tag{5.12}$$

Since $\left(\frac{d_j}{R_j}\right)^2 \le 1$ and $\ln(0.01)$ is a negative value, the MF degree $\eta_j=\exp\left[\left(\frac{d_j}{R_j}\right)^2\ln(0.01)\right]\ge 0.01$.

When d_j takes the extremely small value (i.e., $d_j=0$), the MF degree $\eta_j=1$. Therefore the MF degree η_j will be within the range of [0.01, 1].

When projecting the *m*-dimensional cluster onto each dimension, the corresponding MF center and radius will be center $\mu_{j,i}$ and spread σ_j of $B_{j,i}$ in Equation (5.3), respectively. The proposed CEC technique is a constrained evolving algorithm, which is dependent on two factors: the input $\mathbf{\Phi}$, and the upper boundary R_U. If R_U remains constant (a general case), the clusters will be evolved based only on input information $\mathbf{\Phi}$.

The proposed CEC updates the center of a cluster by considering the information of both previous cluster center position (i.e., Equation (5.8)) and the newest input sample. Thus the updated cluster center of the CEC is less sensitive to outliers. The CEC adjusts the cluster center using an evolving mechanism, $R_j' = \left(\frac{1}{2} + \frac{M - N_j}{2M}\right) R_j$. Those new clusters (containing only a few samples) will have more opportunities to update their centers using new accommodated samples, so as to optimize cluster center position. The centers of the clusters with many samples will be less sensitive to new accommodated samples, especially those new samples which belong to this cluster, but are far away from the cluster center. Therefore the cluster center can be less affected by outliers.

5.1.3 Training Strategy

The parameters of the developed eFNN predictor will be optimized by appropriate training as illustrated in Figure 5.3. To catch linear correlations of the *m*-dimensional data sets, the parameters in the VARMA filter are optimized online by the use of the recursive least square estimate (RLSE).

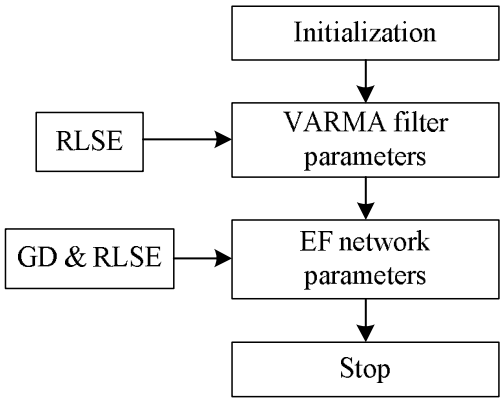

Figure 5.3. Flowchart of the hybrid training process of the developed eFNN predictor.

A hybrid training strategy will be used to optimize parameters in the EF network (Layers 3-5 in Figure 5.1). The GD algorithm is employed to update the nonlinear parameters in nodes $B_{j,i}$ in Layer 4, whereas the RLSE is utilized to adaptively tune the linear parameters in L_j in Equation (5.4). According to our previous research in system training [86], a hybrid training strategy can reduce the search dimension in comparison with a single training method (e.g., the GD), reduce trapping in local optima, and improve convergence of the training process.

Specific training processes are summarized as follows:

1) The initial values of parameters in nodes L_j, (j = 1, 2,…, n) are initialized over the interval [0, 1].

2) The parameters $\theta_{k,i}$ and $\lambda_{l,i}$ (k = 1, 2, ..., p; l = 1, 2, ..., q; i = 1, 2, …, m) in the VARMA filter are optimized online by using the RLSE.

3) After training of the VARMA filter parameters, the nonlinear parameters in nodes $B_{j,i,}$ (j = 1, 2,…, n; i = 1, 2, …, m) are optimized by using a GD algorithm, and linear parameters in L_j are updated by RLSE adaptively.

In the training process, only one training epoch is needed to update the VARMA filter parameters. After the linear correlation information is filtered out, the estimation error data set $\{\varphi_1, \varphi_2, ..., \varphi_m\}$ will retain less regulated information. This means that fewer clusters can be formulated, which can simplify the eFNN predictor structure and speed up training convergence, as will be discussed in Subsection 5.2.

5.2 Performance Evaluation

The effectiveness of the proposed eFNN predictor is verified in this section using a forecasting simulation example of a multi-dimensional financial data set. Next, it is implemented and tested for IM system state prognosis. To simplify the discussion, the developed eFNN predictor using the proposed CEC algorithm is designated as eFNN-CEC.

To make a comparison, the related predictors based on an enhanced fuzzy filtered neural network (EFFNN) [86], an evolving neuro-fuzzy (eNF) scheme [121], and the dynamic evolving neural-fuzzy inference system (DENFIS) [106] are employed for testing. The EFFNN predictor is a four-layer feed-forward NN with the same number of nodes from Layer 1 to Layer 3 (i.e., 20-20-20-1). The eNF

predictor is an evolving NN with three input nodes. The DENFIS is a data-driven NN using a clustering technique.

To evaluate the effectiveness of the proposed evolving CEC algorithm in the eFNN predictor, an evolving clustering method (ECM) suggested in [103] is implemented in the eFNN predictor to replace the CEC algorithm, designated here as eFNN-ECM in short. That is, the only difference between the eFNN-CEC and the eFNN-ECM is associated with evolving clustering algorithms. The maximum number of training epochs of the predictors (i.e., EFFNN, eNF, eFNN-ECM, and eFNN-CEC) is set at 1000.

5.2.1 Exchange Rate Forecasting

In this test, a six-dimensional currency exchange rate data set is used to examine the performance of the proposed eFNN-CEC predictor. The data set consists of daily exchange rates of the Canadian dollar versus the US dollar (the first dimensional data set), the European euro versus the US dollar (the second dimensional data set), the British pound versus the US dollar (the third dimensional data set), the Australian dollar versus the US dollar (the fourth dimensional data set), the Hong Kong dollar versus the US dollar (the fifth dimensional data set) and the New Zealand dollar versus the US dollar (the sixth dimensional data set). All were collected simultaneously over the period between January 1, 2010 and May 31, 2012 [122]. The tests were performed on each dimensional data set. The first one third of the data set in each dimension was used for training, and the remainder was used for testing.

Figures 5.4 to 5.9 illustrate the three-step-ahead forecasting performance of these six data sets using the related predictors, and the related results are summarized in Tables 5.1 to 5.6. It is seen that the eNF predictor is superior to the EFFNN predictor in terms of both forecasting accuracy and the running time, because of its evolving reasoning mechanism. The DENFIS achieves less prediction error than the eNF as indicated in Tables 5.1, 5.3, 5.4 and 5.6, respectively, but it generates more clusters and hence requires more running time. It is seen from Tables 5.2 and 5.5 that the eNF outperforms the DENFIS in terms of number clusters generated, prediction errors and running time, because of its more advanced evolving mechanism. The eFNN related predictors (i.e., eFNN-CEC and eFNN-ECM) yield less forecasting error than those based on the DENFIS, the eNF and the EFFNN, because eFNN undertakes more efficient linear and nonlinear correlation modeling (e.g., eFNN-ECM generates 75% and 67% smaller errors than the eNF and the DENFIS, respectively, as

demonstrated in Table 5.1). The error in percentage is calculated as $E(\%) = 100 \times \frac{E_B - E_A}{E_B}$, where E_A and E_B are the mean square errors of predictor A and predictor B, respectively. The developed CEC evolving method in eFNN-CEC is more efficient than the classical ECM in eFNN-ECM (e.g., eFNN-CEC generates 42% smaller error than eFNN-ECM in Table 5.4).

On the other hand, eFNN-CEC generates the fewest clusters in these tests because of its linear filtering operation and effective clustering algorithms, which takes the least running time correspondingly. The eFNN-CEC predictor outperforms other predictors in capturing and tracking the dynamic behaviors of the underlying systems in this test.

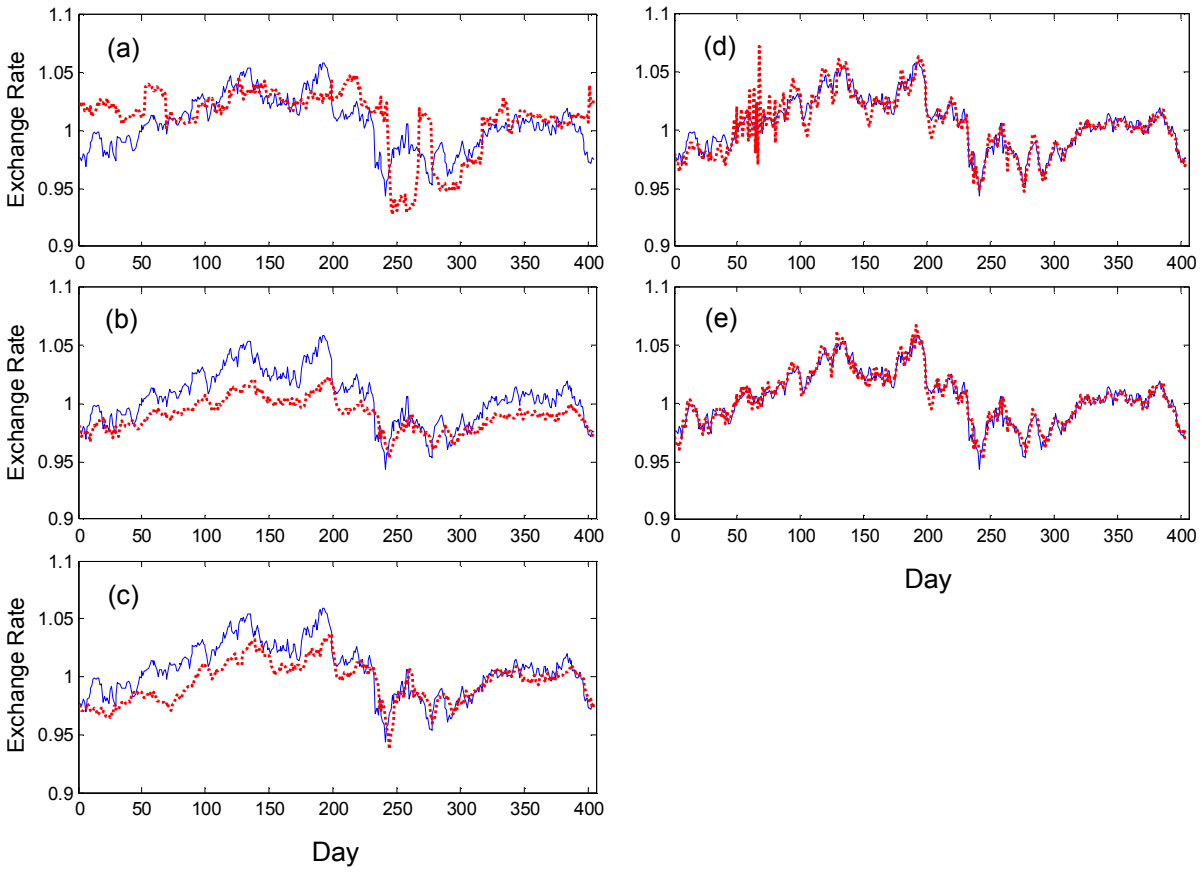

Figure 5.4. Comparison of three-step-ahead forecasting results of daily Canadian dollar/US dollar exchange rate data. The blue solid line is the real data to estimate; the red dotted line is the forecasting results using different predictors: (a) EFFNN; (b) eNF; (c) DENFIS; (d) eFNN-ECM; (e) eFNN-CEC.

Table 5.1. Forecasting results (three-step-ahead) of Canadian dollar/US dollar exchange rate.

Forecasting Schemes	No. of Clusters	MSE (10^{-4})	Running Time (s)
EFFNN	20	6.072	27.211
eNF	6	3.723	22.524
DENFIS	10	2.797	25.979
eFNN-ECM	3	0.918	17.872
eFNN-CEC	2	0.421	16.013

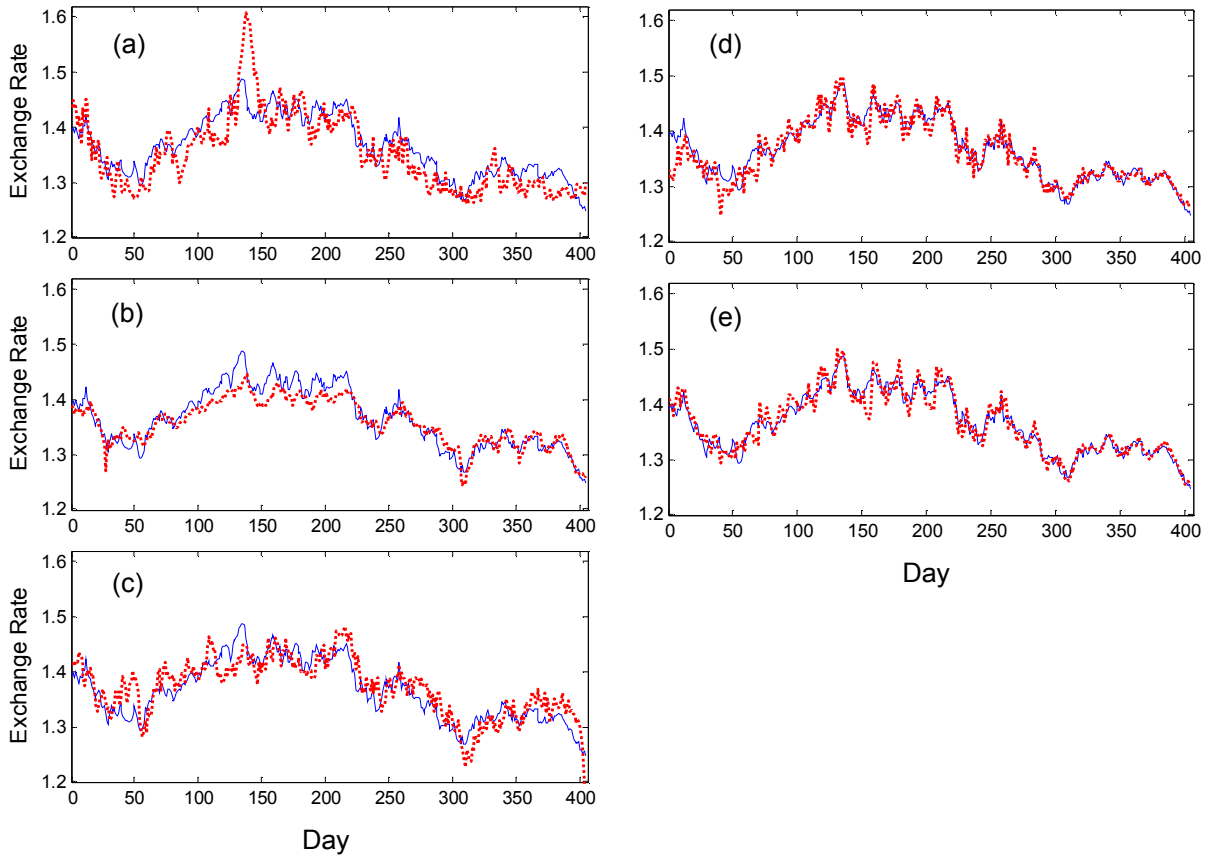

Figure 5.5. Comparison of three-step-ahead forecasting results of daily European euro/US dollar exchange rate data. The blue solid line is the real data to estimate; the red dotted line is the forecasting results using different predictors: (a) EFFNN; (b) eNF; (c) DENFIS; (d) eFNN-ECM; (e) eFNN-CEC.

Table 5.2. Forecasting results (three-step-ahead) of European euro/US dollar exchange rate.

Forecasting Schemes	No. of Clusters	MSE (10^{-4})	Running Time (s)
EFFNN	20	15.000	27.422
eNF	8	4.836	25.835
DENFIS	10	8.843	26.634
eFNN-ECM	4	3.990	21.261
eFNN-CEC	3	2.689	18.813

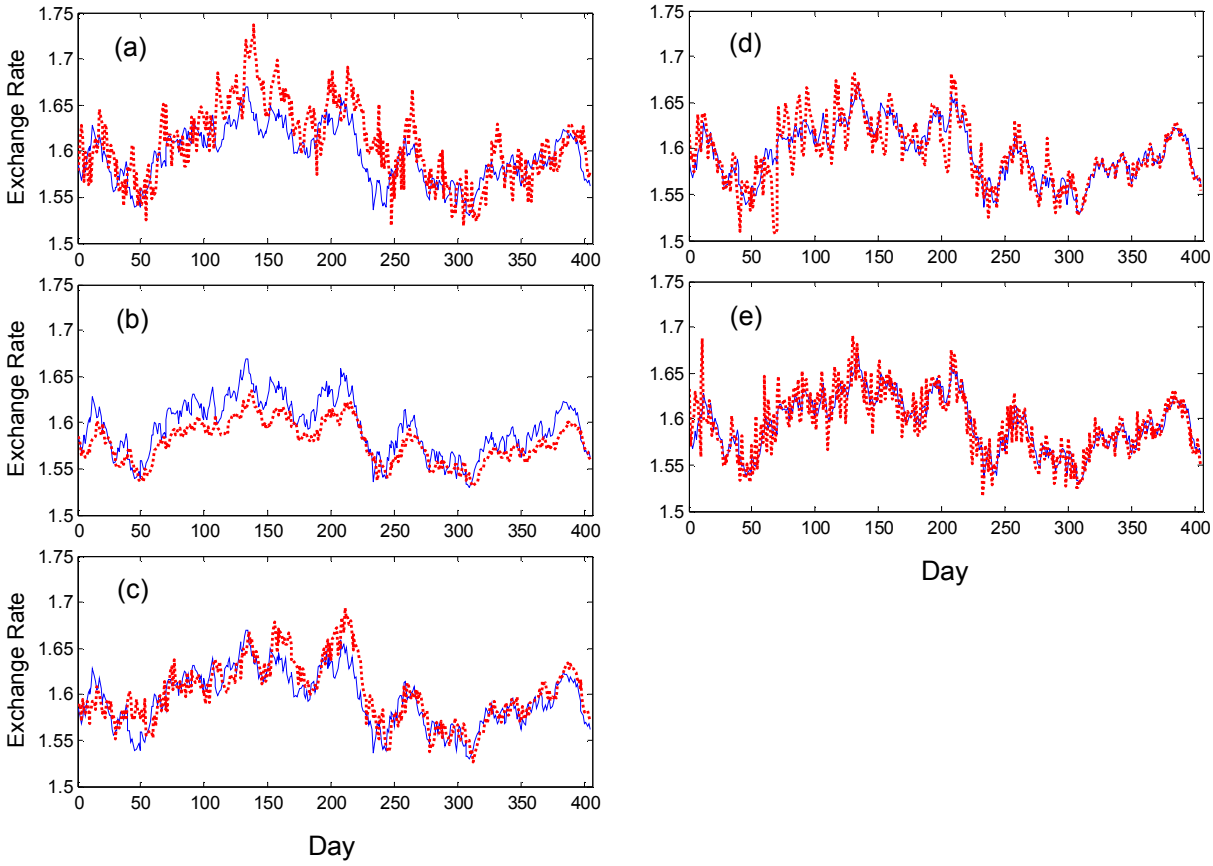

Figure 5.6. Comparison of three-step-ahead forecasting results of daily British pound/US dollar exchange rate data. The blue solid line is the real data to estimate; the red dotted line is the forecasting results using different predictors: (a) EFFNN; (b) eNF; (c) DENFIS; (d) eFNN-ECM; (e) eFNN-CEC.

Table 5.3. Forecasting results (three-step-ahead) of British pound/US dollar exchange rate.

Forecasting Schemes	No. of Clusters	MSE (10^{-4})	Running Time (s)
EFFNN	20	9.281	26.113
eNF	8	4.579	22.492
DENFIS	12	3.656	24.956
eFNN-ECM	3	3.382	17.319
eFNN-CEC	2	3.141	16.593

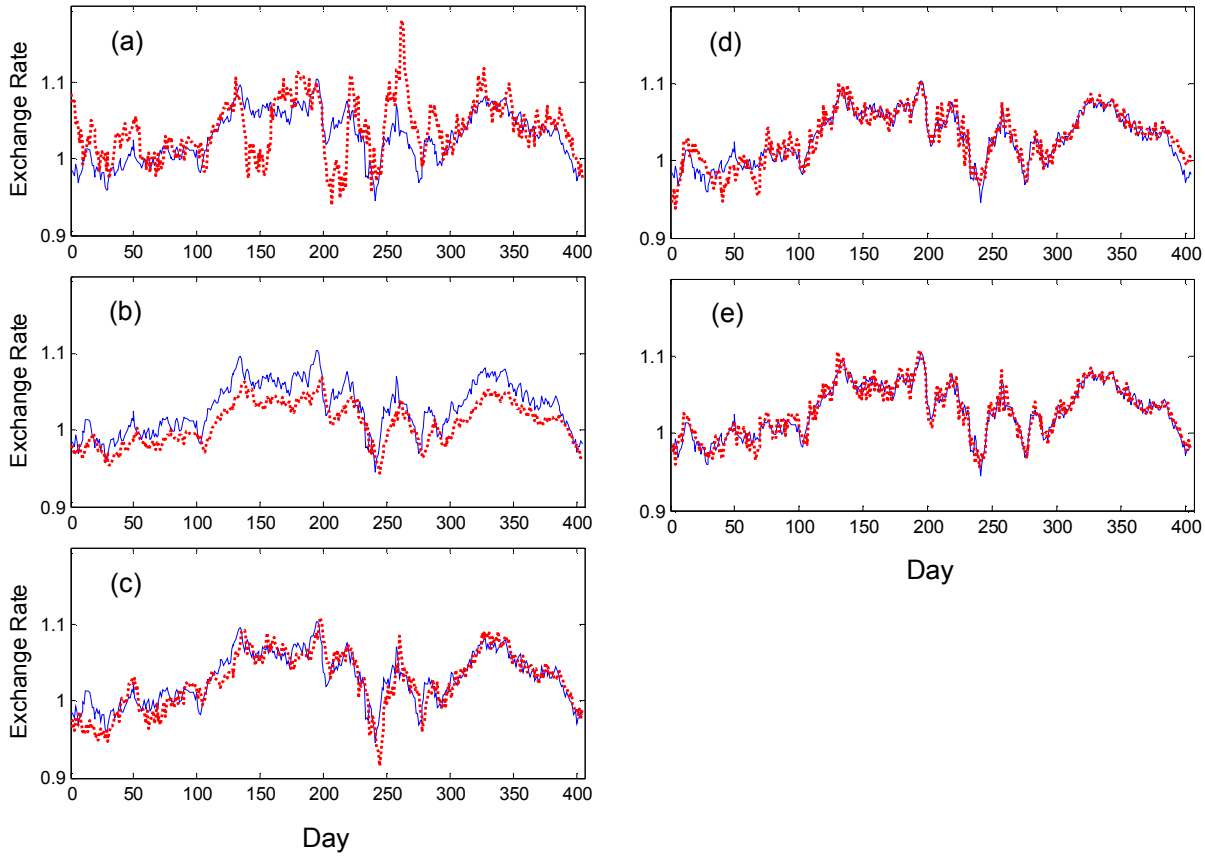

Figure 5.7. Comparison of three-step-ahead forecasting results of daily Australian dollar/US dollar exchange rate data. The blue solid line is the real data to estimate; the red dotted line is the forecasting results using different predictors: (a) EFFNN; (b) eNF; (c) DENFIS; (d) eFNN-ECM; (e) eFNN-CEC.

Table 5.4. Forecasting results (three-step-ahead) of Australian dollar/US dollar exchange rate.

Forecasting Schemes	No. of Clusters	MSE (10^{-4})	Running Time (s)
EFFNN	20	14.000	26.362
eNF	7	6.448	24.804
DENFIS	10	3.502	25.462
eFNN-ECM	3	2.201	16.973
eFNN-CEC	2	1.279	16.109

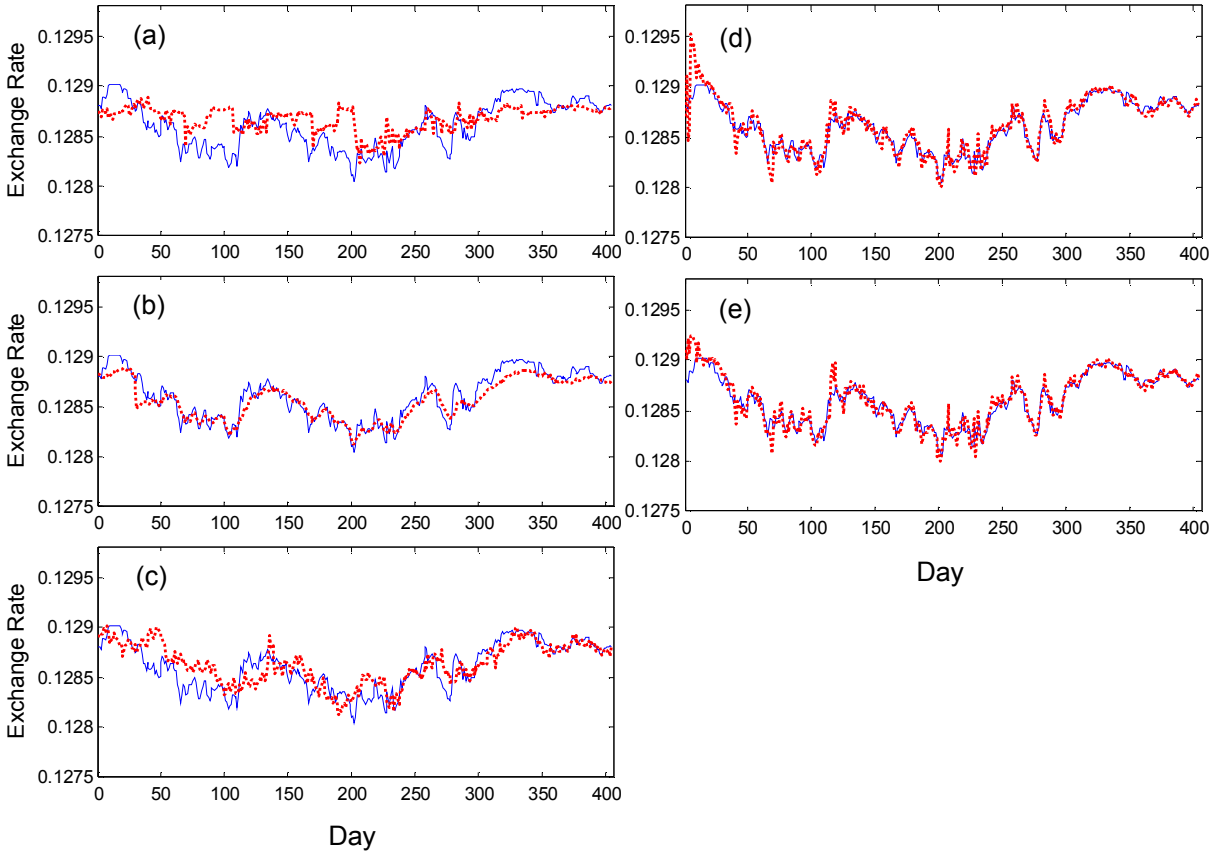

Figure 5.8. Comparison of three-step-ahead forecasting results of daily Hong Kong dollar/US dollar exchange rate data. The blue solid line is the real data to estimate; the red dotted line is the forecasting results using different predictors: (a) EFFNN; (b) eNF; (c) DENFIS; (d) eFNN-ECM; (e) eFNN-CEC.

Table 5.5. Forecasting results (three-step-ahead) of Hong Kong dollar/US dollar exchange rate.

Forecasting Schemes	No. of Clusters	MSE (10^{-8})	Running Time (s)
EFFNN	20	4.436	27.594
eNF	8	0.926	25.481
DENFIS	10	2.321	26.687
eFNN-ECM	4	0.883	17.521
eFNN-CEC	2	0.738	15.596

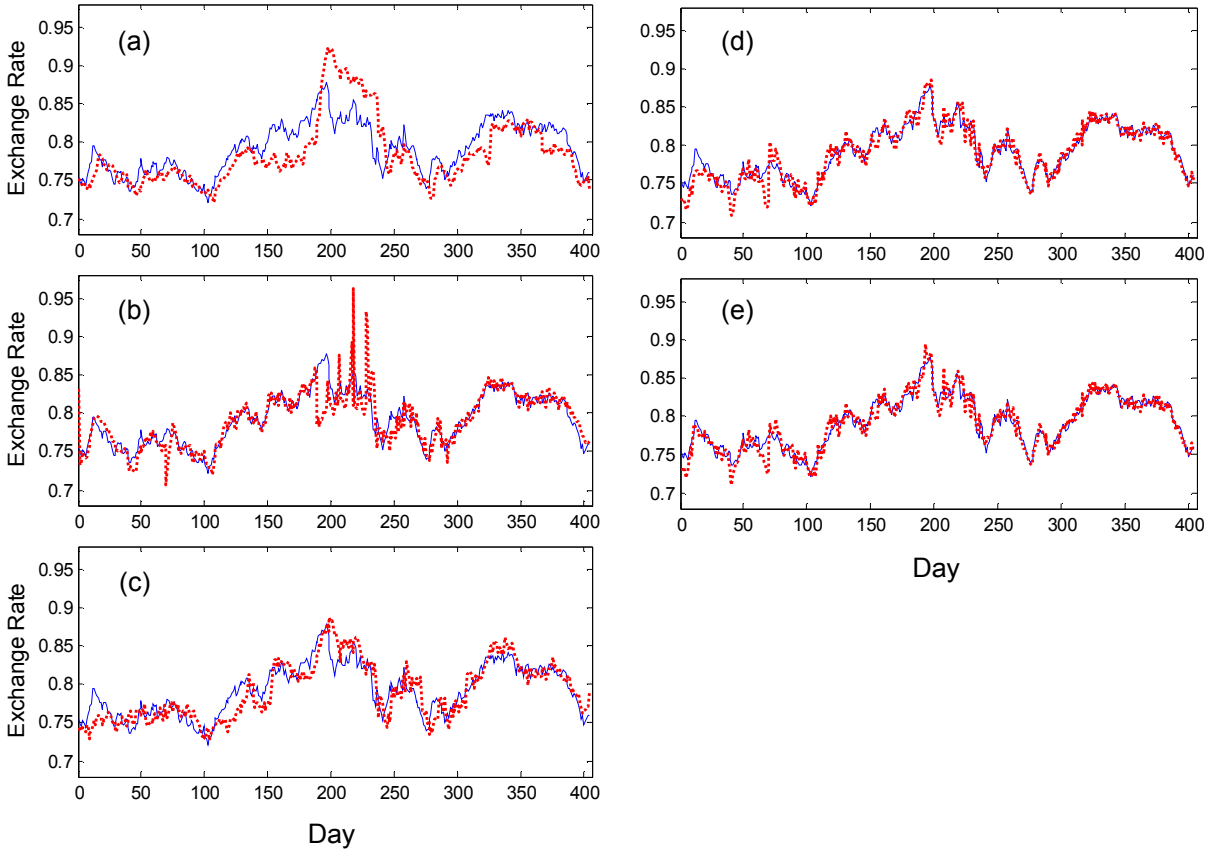

Figure 5.9. Comparison of three-step-ahead forecasting results of daily New Zealand dollar/US dollar exchange rate data. The blue solid line is the real data to estimate; the red dotted line is the forecasting results using different predictors: (a) EFFNN; (b) eNF; (c) DENFIS; (d) eFNN-ECM; (e) eFNN-CEC.

Table 5.6. Forecasting results (three-step-ahead) of New Zealand dollar/US dollar exchange rate.

Forecasting Schemes	No. of Clusters	MSE (10^{-4})	Running Time (s)
EFFNN	20	8.463	26.524
eNF	7	3.951	24.479
DENFIS	8	3.332	24.941
eFNN-ECM	4	1.411	17.893
eFNN-CEC	2	1.141	16.416

5.2.2 Induction Motor System State Prognosis

The developed eFNN-CEC predictor was implemented for IM system state prognosis. An efficient predictor is helpful in modeling IM dynamic characteristics, predicting its future system states, and improving its performance. The experimental setup used in this test was given in Subsection 2.3.1.

This test is to forecast future states of stator current signals that will be used for IM health condition monitoring. During the test, stator current signals were collected at a sampling frequency of 10 kHz. Two stator current signals were used to form a two-dimensional data set for IM system state forecasting. The first dimensional data set is a stator current residual signal from phase 1 by filtering out supply frequency components, and the second dimensional data set is a stator current signal from phase 2. The first 300 data were used for training, and the remaining 600 data were used for testing.

Figures 5.10 and 5.11 demonstrate the respective four-step-ahead forecasting results of the signal residual and the stator current signal using the related predictors, whereas the results are summarized in Tables 5.7 and 5.8. The unit of the stator current signal is in Amperes (A). After calculating the mean square error of the predicted values, the unit becomes A^2 as being used in Tables 5.7 and 5.8. It can be seen that the eNF outperforms the EFFNN, with 29% smaller error in Table 5.7. The DENFIS generates more clusters and would take longer running time than the eNF, but the DENFIS is more accurate than eNF because more fuzzy rules are used to model the data characteristics. The eFNN predictors outperform predictors based on the EFFNN, the eNF and the DENFIS because the eFNN can employ both linear and nonlinear modeling mechanisms. Moreover, the eFNN-CEC predictor

creates the fewest clusters (only three in this case) compared to eFNN-ECM (four to five clusters), eNF (six to eight clusters) and DENFIS (twelve to thirteen clusters). It is seen from Tables 5.7 and 5.8 that the proposed eFNN-CEC predictor provides the highest forecasting accuracy in comparison with eFNN-ECM, eNF, DENFIS and EFFNN predictors. In addition, it can be seen from Tables 5.8 that the eFNN-CEC generates 43% smaller error than the second best predictor, eFNN-ECM. The eFNN-CEC predictor can catch the dynamic behavior of the tested IM system quickly and accurately.

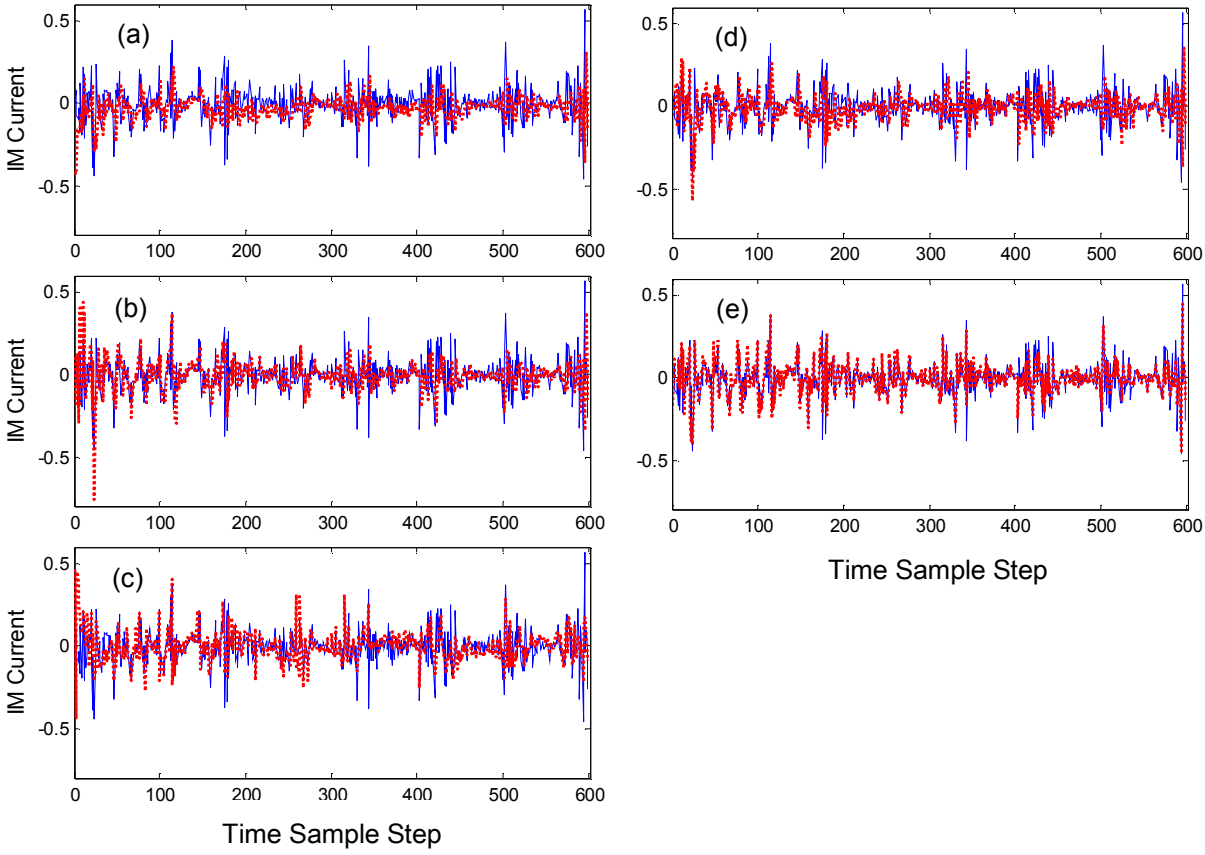

Figure 5.10. Comparison of four-step-ahead forecasting results of the signal residual of an IM. The blue solid line is the real data to estimate; the red dotted line is the forecasting results using different predictors: (a) EFFNN; (b) eNF; (c) DENFIS; (d) eFNN-ECM; (e) eFNN-CEC.

Table 5.7. Forecasting results (four-step-ahead) of the signal residual of an IM (from phase 1).

Forecasting Schemes	No. of Clusters	MSE (A^2)	Running Time (s)
EFFNN	20	0.021	41.593
eNF	8	0.015	38.988
DENFIS	13	0.011	40.124
eFNN-ECM	5	0.008	25.553
eFNN-CEC	3	0.002	20.704

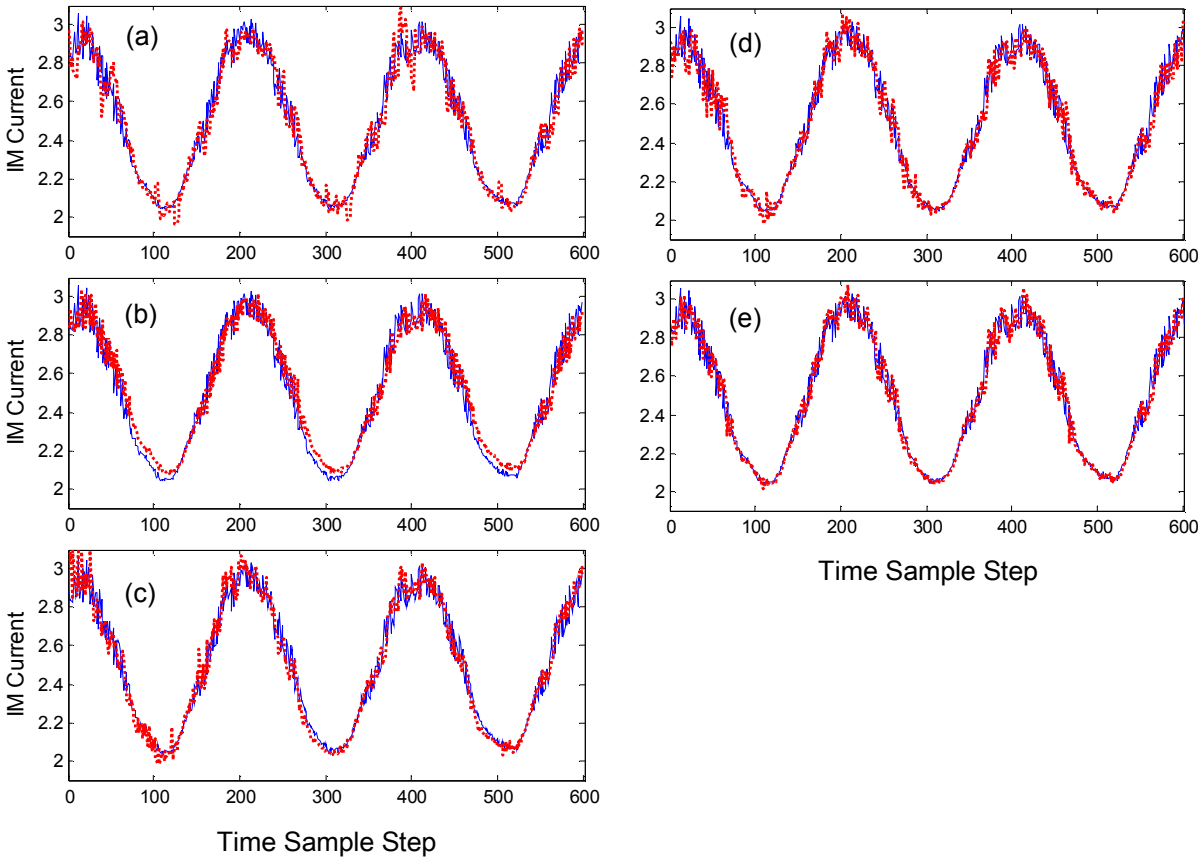

Figure 5.11. Comparison of four-step-ahead forecasting results of the stator current signal of an IM. The blue solid line is the real data to estimate; the red dotted line is the forecasting results using different predictors: (a) EFFNN; (b) eNF; (c) DENFIS; (d) eFNN-ECM; (e) eFNN-CEC.

Table 5.8. Forecasting results (four-step-ahead) of the stator current signal of an IM (from phase 2).

Forecasting Schemes	No. of Clusters	MSE (A^2)	Running Time (s)
EFFNN	20	0.018	39.051
eNF	6	0.017	34.129
DENFIS	12	0.008	37.478
eFNN-ECM	4	0.007	20.732
eFNN-CEC	3	0.004	16.046

The IM data is a two dimensional data set. If the proposed eFNN-CEC technique is used, the residual IM data after VARMA filtering can be represented as $\mathbf{\Phi} = [\varphi_1, \varphi_2]$ in Equation (5.2), which is then fed to EF network for nonlinear modeling. The proposed CEC technique is used to generate clusters from the input patterns $\mathbf{\Phi}$, so as to adaptively construct the structure of EF network as well as Gaussian MFs $B_{j,i}$ in Equation (5.3). The distributions of input patterns $\mathbf{\Phi}$, and the corresponding trained Gaussian MFs are shown in Figures 5.12 and 5.13, respectively. The parameters $\mu_{j,1}$, $\mu_{j,2}$ and σ_j of the Gaussian MFs in Equation (5.3) corresponding to Figures 5.12 and 5.13, are given in Tables 5.9 and 5.10, respectively. From Figures 5.12 and 5.13, it is seen that the derived three Gaussian MFs can capture the distribution of the input patterns effectively. Compared to other predictors with more generated clusters (i.e., Gaussian MFs), the proposed eFNN-CEC can conduct accurate prediction with less running time as illustrated in Tables 5.7 and 5.8.

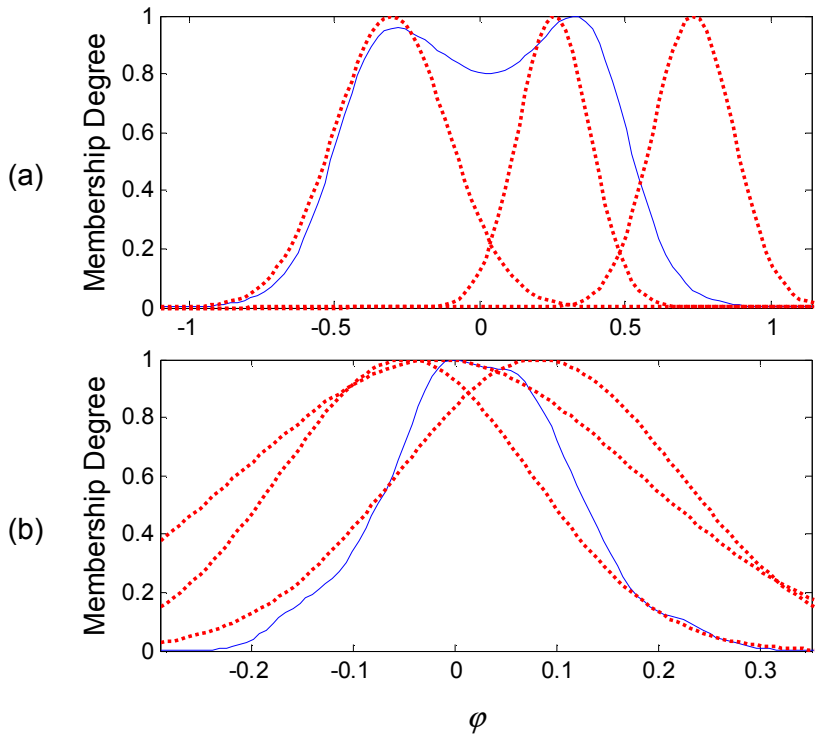

Figure 5.12. The distribution of (a) input patterns φ_1; and (b) input patterns φ_2, associated with the corresponding Gaussian MFs, for the signal residual of an IM. The blue solid line represents the distribution of the input patterns; the red dotted line represents the Gaussian MFs.

Table 5.9. The parameters of the trained Gaussian MFs for the signal residual of an IM.

Parameters	$\mu_{j,1}$ (Fig. 13a)	$\mu_{j,2}$ (Fig. 13b)	σ
Cluster 1 (j = 1)	0.254	-0.048	0.124
Cluster 2 (j = 2)	0.734	0.082	0.139
Cluster 3 (j = 3)	-0.303	-0.016	0.197

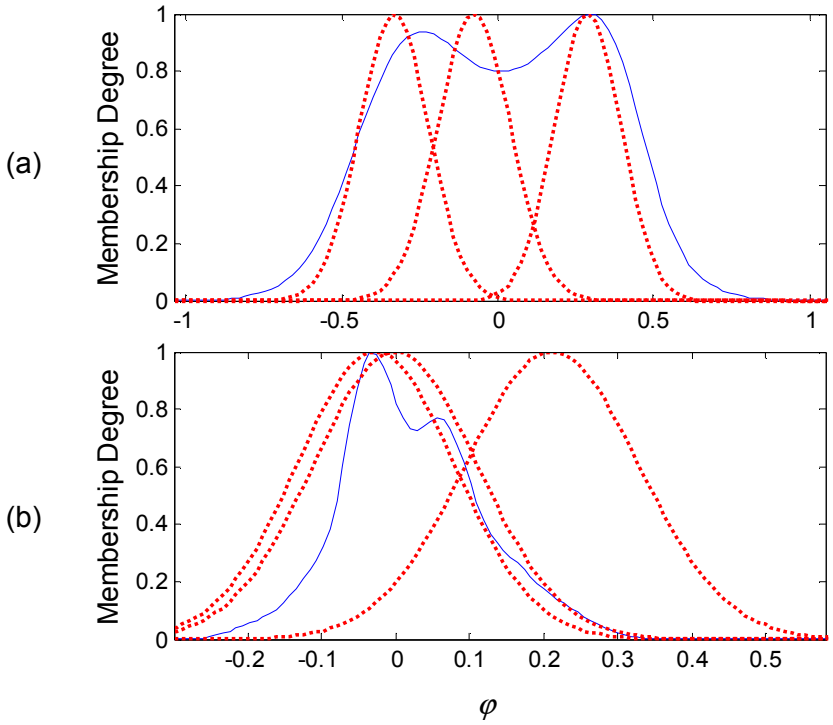

Figure 5.13. The distribution of (a) input patterns φ_1; and (b) input patterns φ_2, associated with the corresponding Gaussian MFs, for the stator current signal of an IM. The blue solid line represents the distribution of the input patterns; the red dotted line represents the Gaussian MFs.

Table 5.10. The parameters of the trained Gaussian MFs for the stator current signal of an IM.

Parameters	$\mu_{j,1}$ (Fig. 14a)	$\mu_{j,2}$ (Fig. 14b)	σ
Cluster 1 (j = 1)	0.288	-0.028	0.107
Cluster 2 (j = 2)	-0.333	-0.001	0.111
Cluster 3 (j = 3)	-0.077	0.211	0.117

5.3 Summary

An evolving fuzzy neural network, eFNN, technique has been developed in this work for multi-dimensional system state forecasting. It can integrate the strength of both the VARMA filter and

nonlinear network modeling in dealing with multi-dimensional data sets. A novel evolving clustering algorithm, CEC, is proposed to adaptively generate fuzzy reasoning clusters and adjust the eFNN network structure. The effectiveness of the proposed eFNN predictor and the new clustering algorithm was verified by simulation test using a multi-dimensional financial data set. The new predictor was also implemented for the induction motor, IM, system state prognosis. Test results showed that the developed eFNN predictor is an accurate forecasting tool, and can capture the dynamic behavior of the tested system quickly and accurately. The CEC is also an effective evolving technique for network structure formulation.

The work presented in this chapter is to achieve the research in Section 1.3: *Develop a knowledge-based data-driven predictor to forecast future states of the IM conditions. The forecast information can be used to further improve the reliability of IM condition monitoring.*

Chapter 6
An Integrated Monitoring System for IM Health Condition Monitoring

In this chapter an integrated fuzzy diagnostic (IFD) system is developed to effectively integrate the diagnostic information and the prognostic information in order to achieve a more accurate assessment of the IM health conditions. Its effectiveness is verified based on a series of testing corresponding to most common IM defects: broken rotor bar(s) and pitting of the bearing.

6.1 Decision-Making Process

The proposed integrated monitoring system consists of two modules: feature extraction and decision-making, as illustrated in Figure 6.1. In feature extraction, appropriate signal processing techniques are used to extract representative features from collected stator current signals and generate fault indicators (or indices). Since different IM defects require different fault detection techniques for analysis, the related index formulation will be discussed in the following subsections corresponding to each defect detection condition.

The decision-making module is to process these representative features for IM health condition monitoring. As discussed in the Chapter 5, the eFNN predictor is sensitive to the sequence of input data; it usually generates a more complex reasoning structure compared with pBoost predictor. Correspondingly, its error convergence will be slower, which may result in degraded training accuracy in online IM monitoring operations, since online monitoring operations require prompt data processing in a short time period. Although the accuracy of the pBoost predictor may be a little lower than the eFNN due to its predefined model of the predictor, it has a compact structure and faster error convergence. Therefore, the pBoost predictor is used in this case to estimate future states of IM fault indices.

The diagnostics is performed component-by-component in the IM. The decision-making module is composed of sBoost classifier, pBoost predictor and IFD system. The sBoost classifier aims to map the fault indices extracted from the related signal processing techniques into different IM health condition categories. The pBoost predictors are used to forecast the future states of each of the IM fault indices. The predicted fault indices indicate the future trends of these indices. If the current fault

indices are contaminated by noise caused by factors such as sudden load disturbances or transient interference from other machines, the trends of these fault indices may provide a different perspective associated with historical (previous) records, which can provide useful information to enhance the reliability of the IM health condition monitoring. This decision-making process is performed by the proposed IFD scheme with a novel confidence-rate-based fuzzy reasoning mechanism as revealed in the following section.

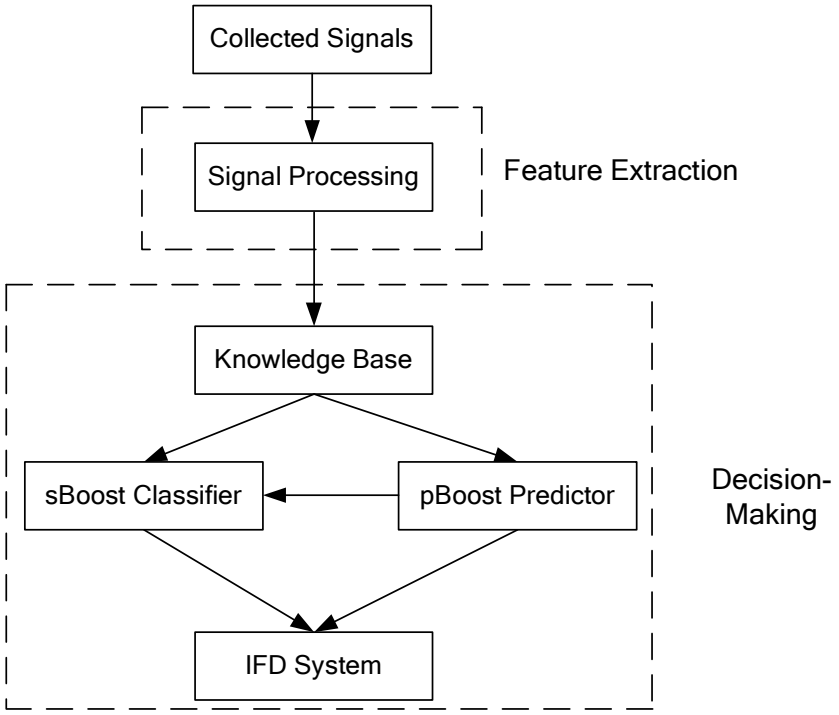

Figure 6.1. The diagram of the proposed integrated monitoring system.

6.2 IM Broken Rotor Bar Fault Diagnosis

6.2.1 Monitoring Indices for IM Rotor Bar Fault Detection

In IM rotor bar condition monitoring, a fault detection technique (an index) from a different information domain could provide a different perspective of the IM health condition. In this work,

fault indices from three information domains: the time domain, the frequency domain, and the time-frequency domain, will be employed for advanced processing in order to improve diagnostic accuracy.

Consider three well-accepted time domain statistics: energy, skewness and kurtosis, as defined in Equations (6.1), (6.2) and (6.3) respectively:

$$\text{Energy: } \sum_{i=1}^{N} x_i^2 \tag{6.1}$$

$$\text{Skewness: } \frac{\sum_{i=1}^{N}(x_i - E_x)^3}{(N-1)\vartheta_x^3} \tag{6.2}$$

$$\text{Kurtosis: } \frac{\sum_{i=1}^{N}(x_i - E_x)^4}{(N-1)\vartheta_x^4} - 3 \tag{6.3}$$

where x_i are the data samples in a data set, i = 1, 2, 3, …, N; and N is the size of a data set; E_x is the mean, and ϑ_x is the standard deviation of the data set. The "-3" in Equation (6.3) is used to make the kurtosis of the normal distribution equal to zero.

A series of tests were conducted using these time-domain techniques (i.e., energy, skewness, and kurtosis) for IM broken rotor bar fault diagnosis. The results are summarized in Table 6.1, and the test data are described in the following subsection. It is seen from Table 6.1 that the skewness of the notch-filtered stator current signal can provide the highest accuracy among these three techniques, which will be selected as the time domain index in this case. The elliptic notch filter is applied to suppress the supply frequency component in each data set and reduce its interference in fault diagnosis [123]. Given a stator current signal, x_i, i = 1, 2, 3, …, N, with length of N. The selected skewness index will be:

$$z_1 = \frac{\sum_{i=1}^{N}(x_i - E_x)^3}{(N-1)\vartheta_x^3} \tag{6.4}$$

where E_x is the mean, and ϑ_x is the standard deviation of the stator current signal.

Table 6.1. Comparison of the related time domain indices for IM broken rotor bar fault diagnosis.

Diagnostic Indices	False Alarms	Missed Alarms	Overall Accuracy
Energy	52	51	65.7%
Skewness	28	34	79.3%
Kurtosis	33	45	74.0%

Among the techniques in frequency domain analysis, the proposed spectrum synch technique in Chapter 2 was shown to outperform other related techniques, and thus will be used as the fault index in the frequency domain in this work. The two first-order fault harmonics of IM broken rotor bar fault in Equations (2.10) and (2.11) with $k = 1$ are synchronized to generate the fault index. In IM broken rotor bar fault related harmonics, these two first-order fault harmonics usually have a more pronounced peak than higher harmonics. Correspondingly, the fault index in the frequency domain will be

$$z_2 = \chi_s \tag{6.5}$$

where χ_s is derived from Equation (2.9).

Among the IM rotor bar fault detection techniques in the time-frequency domain, as discussed in Chapter 1, short time Fourier analysis employs the window function with a fixed length; consequently the signal can only be examined with a fixed time resolution and frequency resolution. The Wigner-Ville distribution generates cross terms that may mask the fault features. Although the wavelet packet analysis can generate more detailed decomposition for some advanced signal analysis, it is usually difficult to recognize the real representative features from the map with redundant and misleading information, for fault detection in complex IM systems. The wavelet transform (WT) decomposes the signal into different time and frequency resolutions, without cross terms. Although the resolution of the WT is lower than the wavelet packet analysis in terms of the decomposition at the detail end, it

applies compact filter bank implementation and the resulting features are relatively easy to explain. Correspondingly, the discrete WT will be used as the time-frequency domain technique in this case.

The fundamental frequency component in the stator current signal is suppressed by using an elliptic notch filter [123]. Then the discrete WT is employed to extract fault related sub-bands [21]. The two sub-bands containing f_{bl1} and f_{br1} are selected, and the root-mean-square (RMS) value of their corresponding decomposed signals is used as a fault index, where f_{bl1} and f_{br1} are the fault frequency components in Equations (2.10) and (2.11) with $k = 1$. Correspondingly, the fault index in the time-frequency domain will be:

$$z_3 = \sqrt{\frac{1}{L_{bl} + L_{br}} \left(S_{bl}^2 + S_{br}^2 \right)} \tag{6.6}$$

where S_{bl} and S_{br} represent the decomposed signals corresponding to the sub-band containing f_{bl1} and the sub-band containing f_{br1} respectively. L_{bl} and L_{br} denote the length of S_{bl} and the length of S_{br} respectively.

6.2.2 Data Preparation

To test the performance of the proposed integrated monitor for IM broken rotor bar fault diagnostics, the stator current data were collected using the experimental setup as described in Subsection 2.3.1. The supply frequency was set at 50 Hz and the sampling frequency f_s = 10 kHz.

For IM broken rotor bar fault diagnosis, one hundred data sets were collected from a healthy IM corresponding to each of the load conditions: light-load condition (approximately 10% of rated power), medium-load condition (approximately 50% of the rated power) and heavy-load condition (approximately 100% of rated power). One hundred data sets were collected from an IM with broken rotor bar fault corresponding to each of these three load conditions. In total, 600 data sets were collected, whereby half were used for training, and the remaining for testing.

6.2.3 Fault Indices Prediction

The forecasting of fault indices could explore the time series properties of each fault index, and synthesize the corresponding prognostic information into fault diagnosis, in order to further improve

the diagnostic accuracy and reliablity. If z_1', z_2', and z_3' are the predicted values of the fault indices of z_1, z_2, and z_3, respectively, the implementation of pBoost predictor for fault index prediction takes the following steps:

1) Generate a series of a fault index (i.e., z_1, z_2, or z_3) to construct the training data set with size N_t.

2) Initialize the distribution of the training data set $L_1(i) = \frac{1}{N_t}$; i = 1, 2, …, N_t.

3) Train an AR model h_t with the training data set and the distribution L_t at step t as discussed in Subsection 4.1.4.

4) Compute the sum of weighted absolute error $\lambda_t = \sum_{i=1}^{N_t} L_t(i)\left|y_d(i) - p_t(i)\right|$, where y_d are the desired values, and p_t are the predicted values at step t using the AR predictor h_t.

5) Calculate the weight of the AR predictor h_t using Equation (4.3).

6) Update the distribution of the training samples using Equation (4.1).

7) Repeat steps 3) to 6) as t = 1, 2, …, T.

8) The predicted fault index (i.e., z_1', z_2', or z_3') is calculated using Equation (4.4).

6.2.4 Performance Evaluation

To verify the proposed integrated monitoring system for IM broken rotor bar fault detection, six methods were employed for comparisons:

Method #1: The fault diagnosis based on time-domain fault index z_1 only.

Method #2: The fault diagnosis based on frequency-domain fault index z_2 only.

Method #3: The fault diagnosis based on time-frequency-domain fault index z_3 only.

Method #4: The fault diagnosis only based on the sBoost classifier using fault indices z_1, z_2, and z_3.

Method #5: The fault diagnosis only based on the sBoost classifier using fault indices z_1', z_2', and z_3' that are predicted by pBoost predictors.

Method #6: The fault diagnosis based on the proposed integrated monitoring system using z_1, z_2, z_3, z_1', z_2', and z_3'.

Figure 6.2 demonstrates the training error convergence and test error convergence of *Method* #4. The convergences of both training error and test error of *Method* #5 are illustrated in Figure 6.3. It is seen that the training errors can converge to zero effectively in the training process. The test error also converges as the number of base learners increases.

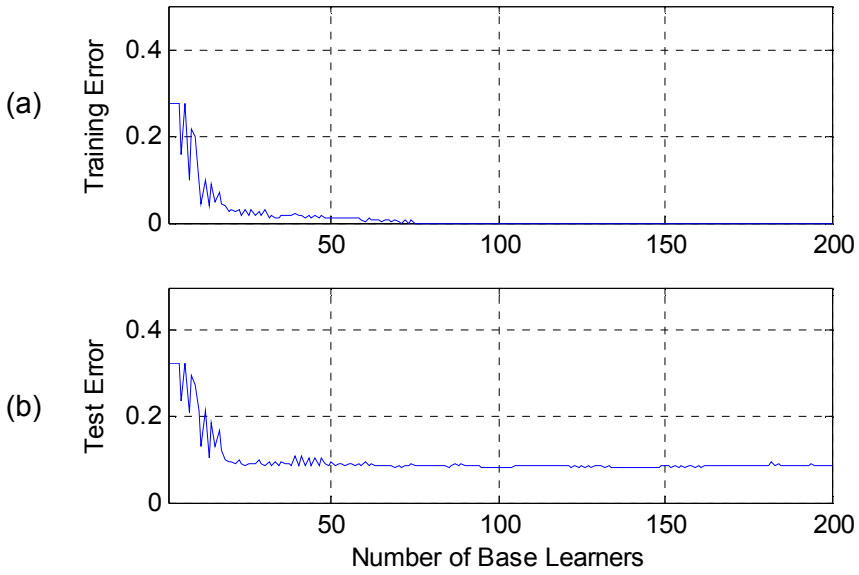

Figure 6.2. The (a) training error and (b) test error using *Method* #4 in IM broken rotor bar fault diagnosis.

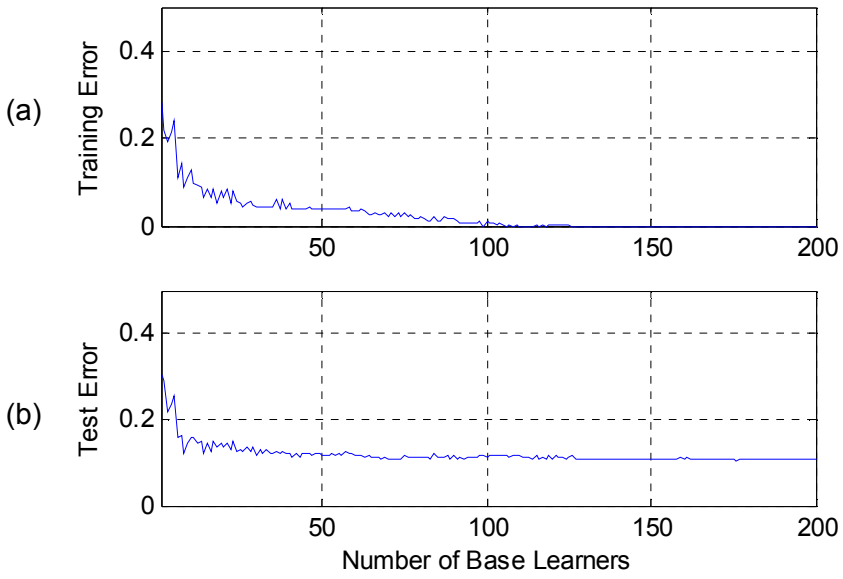

Figure 6.3. The (a) training error and (b) test error using *Method* #5 in IM broken rotor bar fault diagnosis.

Method #6 employs the proposed IFD system to conduct fuzzy inference based on the diagnostic results from *Method* #4 and *Method* #5. Let M_1 and M_2 represent healthy state and damaged state of the IM component (rotor bar in this case), respectively. Given the fault index vector $z = \{z_1, z_2, z_3\}$, and the predicted fault index vector $z' = \{z_1', z_2', z_3'\}$, the fuzzy reasoning mechanism is given as follows:

$\mathcal{R}_1$: IF ($z \subset M_1$) AND ($z' \subset M_1$) THEN (IM component is *healthy*)

$\mathcal{R}_2$: IF ($z \subset M_2$) AND ($z' \subset M_2$) THEN (IM component is *damaged*)

$\mathcal{R}_3$: OTHERWISE (IM component is *possibly damaged*)

If the IM is damaged ($\mathcal{R}_2$), an alarm signal is triggered for repair operations. If the IM is possibly damaged ($\mathcal{R}_3$), the following two rules should be performed:

$\mathcal{R}_4$: IF ($\mathcal{R}_3$ is satisfied) AND $\dfrac{H(z)C(z)+H(z')C(z')}{2} < 0$ THEN (IM component is *healthy*)

$\mathcal{R}_5$: IF ($\mathcal{R}_3$ is satisfied) AND $\dfrac{H(z)C(z)+H(z')C(z')}{2} \geq 0$ THEN (IM component is *damaged*)

where $H(\cdot) \in \{-1, 1\}$ indicates a fault index vector that belongs to one of the health condition categories (damaged or healthy) of the IM component, which can be calculated using Equation (3.19). $H(\cdot) = 1$ indicates the damaged IM state; $H(\cdot) = -1$ represents the healthy IM state. $C(\cdot) \in [0, 1]$is the confidence rate, which can be calculated using Equation (3.20). It indicates to what extent a fault index vector belongs to a health condition category. The larger the $C(\cdot)$ is, the higher the probability that a fault index vector will belong to a specific health condition category. For example, $H(z)C(z) \geq 0$ indicates the IM component is damaged based on the index vector z with confident rate $C(z)$.

Rules $\mathcal{R}_4$ and $\mathcal{R}_5$ are used to trigger the integrated monitor for intense processing actions. To make a clear decision, the monitoring interval becomes much shorter than that in the general monitoring

operations. Data sets will be collected in the following steps and the corresponding fault indices are computed. When $\mathcal{R}_1$ is satisfied in the following steps, then IM component is healthy. When $\mathcal{R}_2$ is satisfied in the following steps, then IM component is damaged. Otherwise, the similar monitoring process rolls over until $\mathcal{R}_1$ or $\mathcal{R}_2$ is satisfied.

Each signal processing technique (e.g. the proposed SS technique) may handle IM fault detection at certain load range, and an intelligent system is needed to improve the accuracy of fault detection in a broader load range. Table 6.2 summaries the test results using these six methods. It is seen that *Methods* #4 and #5 can diagnose IM broken rotor bar fault more accurately than *Methods* #1, #2 and #3, because *Methods* #1, #2 (the proposed SS technique) and #3 only extract fault indices from the respective time domain, the frequency domain, and the time-frequency domain with certain limitations such as load range, and cannot provide robust description of the IM fault states. The utilization of fault indices from three domains can assess IM health conditions more accurately because the proposed sBoost classifier can adaptively resolve decision conflicts (i.e., noisy samples) caused by fault indices from different domains using the proposed sample weight regulator. The fuzzy rules $\mathcal{R}_1$, $\mathcal{R}_2$, $\mathcal{R}_4$, and $\mathcal{R}_5$ construct the IFD system, which is used in *Method* #6 to conduct primary fault diagnosis. *Method* #6 outperforms *Method* #4 and *Method* #5, because IFD system can effectively integrate the information from both the classifier and predictor by using the confident-rate-based fuzzy inference system.

Table 6.2. Diagnostic testing results of IM broken rotor fault using different methods.

Diagnostic Scheme	False Alarms	Missed Alarms	Overall Accuracy
Method #1	28	34	79.3%
Method #2	12	35	84.3%
Method #3	21	35	81.3%
Method #4	8	17	91.7%
Method #5	12	20	89.3%
Method #6	0	3	99.0%

6.3 IM Bearing Outer Race Fault Diagnosis

6.3.1 Monitoring Indices for IM Bearing Fault Detection

Similar to the analysis of IM rotor bar condition monitoring in the previous subsection, three fault detection techniques (or indices) from different information domains: the time domain, the frequency domain, or the time-frequency domain, are selected to improve the robustness of bearing fault diagnosis.

In time domain analysis, the technique will be selected from three time domain methods: energy, skewness, and kurtosis, as defined in Equations (6.1), (6.2) and (6.3), respectively. A series of tests were conducted using these techniques, and the test results of these techniques for IM bearing fault diagnosis are summarized in Table 6.3. It is seen that the skewness of the notch-filtered stator current signal is the most accurate, and will therefore be utilized as the fault index in this case. The supply frequency component is suppressed by using an elliptic notch filter [123]. Let x_i in Equation (6.2) denote the stator current signal, $i = 1, 2, 3, \ldots, N,$ with length of N. The skewness index will be:

$$z_4 = \frac{\sum_{i=1}^{N} (x_i - E_x)^3}{(N-1)\vartheta_x^3} \tag{6.7}$$

where E_x is the mean, and ϑ_x is the standard deviation of the stator current signal.

Table 6.3. Comparison of the related time domain indices for IM bearing fault diagnosis.

Diagnostic Indices	False Alarms	Missed Alarms	Overall Accuracy
Energy	48	63	63.0%
Skewness	51	19	76.7%
Kurtosis	49	50	67.0%

The spectrum synch technique that was proposed in Chapter 2 will be used as the fault index in the frequency domain. The local bands containing 7th to 14th IM bearing fault harmonics (i.e., m = 7, 8, …, 14 in Equation (1.5)) are synchronized to generate the fault index z_2 in the frequency domain, which has the same representation as Equation (6.5). These fault harmonics are selected because they are less affected by the supply frequency component (50Hz) and have relatively higher peak magnitudes.

To derive the fault index in the time-frequency domain, the fundamental frequency component in stator current signal is suppressed by using an elliptic notch filter [123]. Based on the analysis in Subsection 6.2.1, the discrete WT is employed to extract fault related sub-bands from the stator current signal [21]. The sub-bands containing f_{cl1} and f_{cr1} respectively are selected and the RMS value of their corresponding decomposed signatures is used for computation; $f_{cl1} = \left| f_p - f_v \right|$ and $f_{cl1} = \left| f_p + f_v \right|$ are two of the first-order fault harmonics given in Equation (1.5) with m = 1. Correspondingly, the fault index in the time-frequency domain will be given as:

$$z_5 = \sqrt{\frac{1}{L_{cl} + L_{cr}} \left(S_{cl}^2 + S_{cr}^2 \right)} \tag{6.8}$$

where S_{cl} and S_{cr} represent the decomposed signals corresponding to sub-bands containing f_{cl1} and f_{cr1} respectively. L_{cl} and L_{cr} are the lengths of S_{cl} and of S_{cr}, respectively.

6.3.2 Performance Evaluation

To test the performance of the proposed integrated monitor for IM bearing defect diagnosis, the stator current data were collected with the sampling frequency f_s = 10 kHz, and the supply frequency was fixed at 50 Hz. One hundred data sets were collected from a healthy IM corresponding to each of the load conditions: light-load condition, medium-load condition and heavy-load condition. One hundred data sets were collected from an IM with bearing defect corresponding to each load test condition. In total, 600 data sets were collected, whereby half were used for training, and the remaining for testing.

To verify the proposed integrated monitoring system for IM bearing fault detection, six methods were employed for comparisons. The predicted fault indices z_4', z_2', and z_5' were generated in the same procedure as discussed in Subsection 6.2.3.

Method #1: The fault diagnosis based on time-domain fault index z_4 only.

Method #2: The fault diagnosis based on frequency-domain fault index z_2 only.

Method #3: The fault diagnosis based on time-frequency-domain fault index z_5 only.

Method #4: The fault diagnosis only based on the sBoost classifier using fault indices z_4, z_2, and z_5.

Method #5: The fault diagnosis only based on the sBoost classifier using fault indices z_4', z_2', and z_5' that are predicted by pBoost predictors.

Method #6: The fault diagnosis using the proposed integrated monitoring system using z_4, z_2, z_5, z_4', z_2', and z_5'.

Figure 6.4 illustrates the training error convergence and test error convergence of *Method* #4. The convergence of both training error and test error of *Method* #5 is illustrated in Figure 6.5. It is seen that the training error can effectively converge to zero; the test error converges as the number of base learners increases.

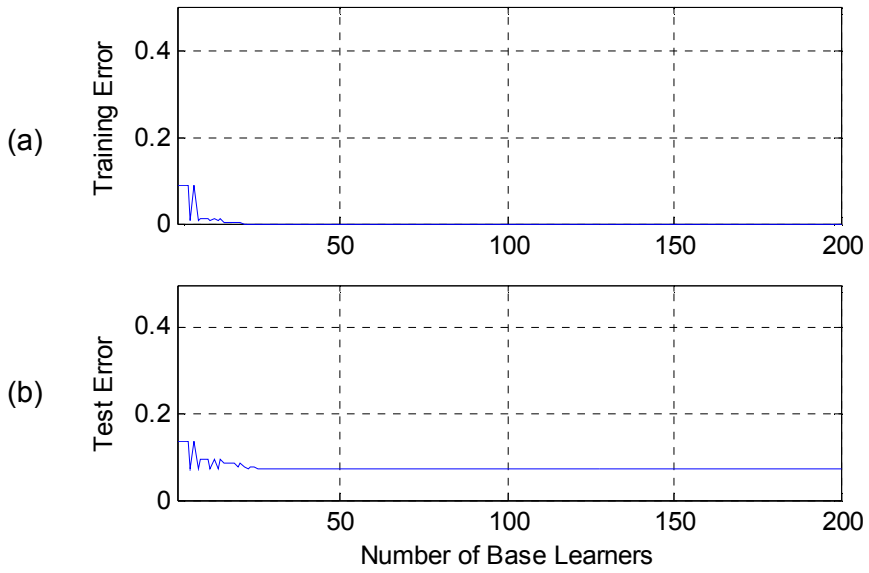

Figure 6.4. The (a) training error; and (b) test error using *Method* #4 in IM outer race bearing defect diagnosis.

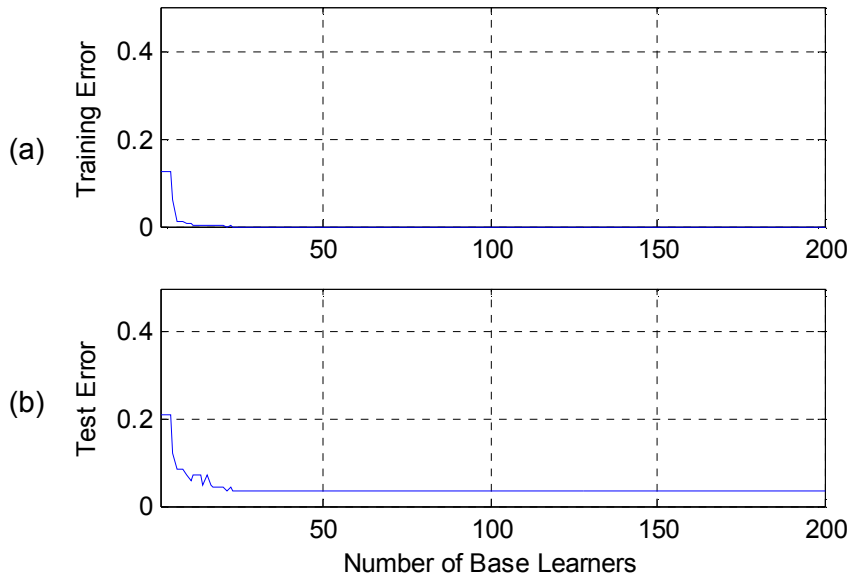

Figure 6.5. The (a) training error; and (b) test error using *Method* #5 in IM outer race bearing defect diagnosis.

The test results using these six methods are summarized in Table 6.4. It is seen that *Method* #4 and *Method* #5 can diagnose IM bearing defect more accurately than *Methods* #1 to #3, because the integrated information can help capture the IM health states more accurately. The sample weight regulator in the proposed sBoost classifier can adaptively resolve decision conflicts (i.e., noise samples) caused by fault indices in different domains and improve diagnostic accuracy. *Method* #6 employs IFD system composed of fuzzy rules $\boldsymbol{\mathcal{R}}_1$, $\boldsymbol{\mathcal{R}}_2$, $\boldsymbol{\mathcal{R}}_4$, and $\boldsymbol{\mathcal{R}}_5$ to conduct primary IM fault diagnosis. It outperforms *Methods* #4 and #5 due to its effective integration of the information from both the classifier and the predictor by using the confident-rate-based fuzzy inference system.

Table 6.4. Diagnostic testing results of IM bearing defect using different methods.

Diagnostic Scheme	False Alarms	Missed Alarms	Overall Accuracy
Method #1	51	19	76.7%
Method #2	25	24	83.7%
Method #3	32	29	79.7%
Method #4	15	6	93.0%
Method #5	7	3	96.7%
Method #6	0	0	100%

6.4 Summary

An intelligent monitoring system is proposed in this chapter for IM health condition monitoring, especially for IM broken rotor bar fault and bearing outer race defect. The proposed integrated monitor is composed of two modules: feature extraction and decision-making. The proposed IFD scheme integrates diagnostic information from sBoost classifier and prognostic information from pBoost predictor to improve monitoring accuracy of IM health conditions.

The work presented in Chapters 6 is to achieve the research goal in Section 1.3: *Propose a new diagnostic scheme to effectively integrate the diagnostic and prognostic information for IM health condition monitoring, and conduct automatic diagnostic decision-making.*

Chapter 7
Conclusions and Future Work

7.1 Conclusions

The IMs are commonly used in various industrial facilities, and a reliable condition monitoring system can be used to diagnose the IM fault at its early stage, so as to prevent malfunction of the driven machinery, and to improve productivity. Several IM fault detection techniques have been proposed in the literature for IM health condition monitoring, but each has its own merits and limitations. It still remains a challenging task to accurately recognize the IM fault due to reasons such as insignificant fault features under some load conditions. An intelligent monitoring system was developed in the thesis to provide more reliable IM fault diagnostics. The developed intelligent monitor consists of two modules: feature extraction and decision-making. Feature extraction is a process to extract fault-related representative features from stator current signals. The decision-making module consists of pattern classification, system state prediction and the fuzzy inference system. The diagnostic information is retrieved by mapping fault features to IM health condition categories using the pattern classification technique. The prognostic information is provided by estimating the future states of fault features (indices) using the system state prediction technique. Finally, the fuzzy inference system is utilized to integrate the diagnostic information and prognostic information, in order to provide a more accurate assessment of the IM health conditions.

A novel fault detection technique, the spectrum synch technique, was proposed in Chapter 2 to extract representative features associated with the early IM defects from the IM stator current spectrum. The local bands of the IM fault frequency components are synchronized to form a fault information spectrum, in which the fault features are enhanced and the unrelated high amplitude frequency components are mitigated. A central kurtosis method is proposed to effectively extract useful information from the fault information spectrum to generate an IM fault index. The effectiveness of the proposed spectrum synch technique was demonstrated through experimental tests on IM with broken rotor bars and IM with pitted bearing outer race. The experiments were conducted under different operating conditions (i.e., different supply frequencies and different load conditions). Test results showed the superiority of the proposed spectrum synch technique over the commonly used techniques in the frequency domain, the power spectral density, and the envelope analysis.

A new pattern classification technique, the selective boosting classifier, was developed in Chapter 3 to categorize the fault indices from selected fault detection techniques in order to diagnose the IM fault. The traditional boosting techniques suffer from the overfitting problem, which can degrade the classification accuracy. The proposed selective boosting, sBoost, classifier can adaptively process noisy data based on the noise level of each sample, so as to enhance the classification performance. An error correction mechanism was also employed to further improve the classification accuracy by examining the class label distribution in the neighborhood of each sample. The effectiveness of the sBoost classifier was verified by using 12 benchmark examples from the literature. Test results showed the proposed sBoost classifier to be an effective classification tool. It could categorize the patterns with different characteristics effectively and accurately.

A novel system state prediction technique, pBoost predictor, was proposed in Chapter 4 to forecast the future states of the IM health conditions. A base learner is adaptively incorporated into the ensemble at each step to improve the performance of the ensemble predictor, and the resultant ensemble predictor outperforms all the base learners. Each base learner addresses a particular data distribution, which is updated based on the performance of the ensemble at the previous step. If the base learner is relatively strong or has relatively good performance (e.g., AR predictor), the ensemble predictor is prone to suffering from the overfitting problem. An advanced sample weight regulation mechanism was suggested to reduce the overfitting problem. The effectiveness of the proposed pBoost predictor was verified by simulation tests and a real-world application. The test results showed that the pBoost predictor was an effective tool to conduct system state prediction. The prognostic information can be used to further improve the reliability of the IM health condition monitoring.

A knowledge-based evolving fuzzy neural network (i.e., eFNN) predictor was developed in Chapter 5 to predict future state of the system. In the proposed eFNN, the linear properties of the data are modeled by a vector autoregressive-moving-average (i.e., VARMA) system and the nonlinear properties of the data are characterized by evolving NN system. A novel clustering technique was proposed to adjust the structure of evolving NN system, in order to capture the characteristics of the input data more accurately. Since the linear properties of the data are filtered out by VARMA, less structured information exists in the remaining data. Thus, fewer clusters are generated by the evolving

NN system, and the simplified structure of the predictor can further facilitate reasoning and training operations.

An integrated monitoring system was developed in Chapter 6 to synthesize the information from both sBoost classifier and pBoost predictor for IM health condition monitoring. In the proposed integrated monitor, a confidence-rate-based reasoning mechanism is proposed to address uncertain IM fault diagnostic decisions, in order to improve the accuracy of fault diagnosis. The effectiveness of the developed integrated monitor was verified by experimental tests corresponding to the common IM faults (e.g., IM broken rotor bar fault and IM bearing outer race defects) under different load conditions (i.e., light-load, medium-load and heavy-load). The test results demonstrated that the proposed integrated monitoring system is a reliable IM fault diagnosis tool, and it can provide a more accurate assessment of IM health conditions.

7.2 Ideas for Future Work

Future research would be needed to cover the following topics:

1) The experiments on IM in this work were limited to broken bars and defects on the outer race of the bearing. The motor manufacture was responsible to induce these faults, and in the manufacturer's assessment, those were the common faults and they represented what would be considered early failure. In future research, more fault types and of varying severity could be examined under a wider range of operating conditions.

2) Only the stator current signals were utilized in the current work. It would be tempting to test the developed system with signals coming from different sensors, specifically, the current sensors and the vibration sensors.

3) The developed sBoost, pBoost, eFNN techniques were demonstrated successfully in this work that these techniques can be extended to a wider range of machinery diagnostics. As a matter of fact in the course of this work, other techniques were also developed, see Appendices B-D, that would be more appropriate for other applications rather than the narrow field of IM health condition monitoring.

4) Test the developed system on IM in industrial environment.

Appendix A
Characteristic Frequencies of Rolling Element Bearings

To derive the rolling element bearing characteristic frequencies, the outer race of a bearing is considered fixed and the inner race rotates with the shaft, which is common in most applications. Thus the linear speed of outer race V_o is zero and the rotating frequency of the inner race, f_i, equals to the shaft rotating speed.

Considering there is no sliding between the rotating parts, the cage linear speed V_c can be given by,

$$V_c = \frac{V_o + V_i}{2} = \frac{V_i}{2} = \frac{\omega_i r_i}{2} \tag{A.1}$$

where V_i, ω_i and r_i are the linear speed, angular speed and radius of the inner race respectively. From Figure 1.3, the following can be derived

$$r_c = \frac{D}{2};\ \ r_i = \frac{D}{2} - \frac{d\cos(\theta)}{2};\ \ r_o = \frac{D}{2} + \frac{d\cos(\theta)}{2};\ \ r_r = \frac{d}{2} \tag{A.2}$$

where d is the ball diameter, D is the pitch diameter, and θ is the contact angle.

From Equations (A.1) and (A.2), the cage angular speed can be given by

$$\omega_c = \frac{1}{2}\left[\omega_i\left(1 - \frac{d\cos(\theta)}{D}\right)\right] \tag{A.3}$$

Provided $\omega_c = 2\pi f_c$, the cage rotating frequency can be expressed in *Hz* as,

$$f_{cbr} = \frac{1}{2}\left[f_i\left(1 - \frac{d\cos(\theta)}{D}\right)\right] \tag{A.4}$$

where f_i is the inner race rotating frequency or shaft speed.

The outer race defect frequency f_{obr} can be calculated from the relative angular speed between the cage and outer race.

$$\begin{aligned} f_{obr} &= N(\omega_c - \omega_o) \\ &= N\omega_c \\ &= \frac{N}{2}\left[f_i\left(1 - \frac{d\cos(\theta)}{D}\right)\right] \end{aligned} \tag{A.5}$$

where N is number of rolling elements.

The inner race defect frequency f_{ibr} can be evaluated from the relative angular speed between the cage and inner race:

$$\begin{aligned} f_{ibr} &= N(\omega_i - \omega_c) \\ &= N\left\{ f_i - \frac{1}{2}\left[f_i\left(1 - \frac{d\cos(\theta)}{D}\right)\right]\right\} \\ &= \frac{N}{2} f_i\left(1 + \frac{d\cos(\theta)}{D}\right) \end{aligned} \tag{A.6}$$

The rolling element defect frequency f_{rbr} can be defined as the rolling element frequency rotation around its own center. The ball angular speed corresponding to its own center is given as follows,

$$\omega_r = \frac{V_r}{r_r} \tag{A.7}$$

$$V_r = (\omega_i - \omega_c) r_i \tag{A.8}$$

$$\omega_r = \frac{(\omega_i - \omega_c) r_i}{r_r} \tag{A.9}$$

The rolling element defect frequency f_{rbr} can then be calculated from,

$$f_{rbr} = \frac{D}{2d} f_i \left[1 - \left(\frac{d \cos\theta}{D} \right)^2 \right] \tag{A.10}$$

Appendix B
The Conjugate Levenberg - Marquardt Training Technique

A new training technique, a Conjugate Levenberg-Marquardt (CLM) method, was proposed to improve forecasting accuracy, which was published in [86]: "D. Li, W. Wang, and F. Ismail, Enhanced fuzzy-filtered neural network for material fatigue prognosis, Applied Soft Computing, vol. 13, no. 1, pp. 283-291, 2013". Although this technique is not used for this IM fault diagnosis, it is fast in convergence, which is especially useful for nonlinear parameter optimization.

The classical conjugate gradient (CG) methods are iterative algorithms that aim to generate a search direction that conjugates to all previous descent directions based on the current steepest descent direction. However they only focus on deriving a descent direction from the combination of the last descent direction and current steepest descent direction. As a result, it is possible that the steepest descent direction is not able to provide satisfactory search efficiency no matter what formulation is applied to manipulate the last descent direction and current steepest descent direction. To address this issue, the proposed CLM applies Levenberg-Marquardt (LM) direction instead of the commonly used steepest descent direction to improve training convergence and efficiency.

It is known that the LM direction is an efficient search direction that varies between Newton's direction and steepest descent direction. It is more efficient than the first-order steepest descent algorithm. However the second-order search ability of LM method may be mitigated when LM direction deviates from Newton's direction. Meanwhile its specific formulation with only one adjustable parameter λ as demonstrated in Equation (B.2) will restrict its ability in exploring better descent directions. This may result in slow convergence. By integrating second-order CG formulation and LM direction, the proposed CLM technique can enhance the second-order search power of LM while a broader space (λ in Equation (B.7) and step length of the derived CLM direction) can be further exploited for optimal search directions. Thus CLM is expected to outperform LM method in training efficiency.

Given a symmetric matrix $Q \subset n \times n$, and two n-dimensional vectors d_j and d_k that are mutually conjugate with respect to Q, a new direction, d_k, that conjugates to all previous descent directions d_j ($j<k$) with respect to Q will be

$$d_k = u_k - \sum_{j=0}^{k-1} \frac{d_j^T Q u_k}{d_j^T Q d_j} d_j \tag{B.1}$$

In CG, the u_k is a set of negative steepest descent directions $-g_k$, but they may not be optimal. The CLM technique will employ the direction l_k to replace $-g_k$:

$$l_k = (H_k + \lambda I) g_k \tag{B.2}$$

where g_k is the gradient; I is an identity matrix; H_k is the Hessian matrix in the k^{th} iteration; and λ is a positive quantity used to determine the direction of l_k . When λ approaches zero, l_k can be considered as a Newton's direction. When λ is close to positive infinity, l_k approximates the steepest descent direction. Substituting $u_k = l_k$ in Equation (B.1) yields

$$d_k = l_k - \sum_{j=0}^{k-1} \frac{d_j^T Q l_k}{d_j^T Q d_j} d_j \tag{B.3}$$

Reorganizing Equation (B.3), we get

$$\begin{aligned} d_k &= l_k + \beta_k d_{k-1} \\ &= (H_k + \lambda I) g_k + \beta_k d_{k-1} \end{aligned} \tag{B.4}$$

where

$$\beta_k = \frac{l_k^T (g_k - g_{k-1})}{d_{k-1}^T (g_k - g_{k-1})} \tag{B.5}$$

d_{k-1} can be derived from Equation (B.3) by switching k to k-1,

$$d_{k-1} = l_{k-1} - \sum_{j=0}^{k-2} \frac{d_j^T Q l_{k-1}}{d_j^T Q d_j} d_j \tag{B.6}$$

Substituting Equation (B.6) into Equation (B.5) yields

$$\beta_k = \frac{l_k^T (g_k - g_{k-1})}{g_{k-1}^T g_{k-1}}$$

$$= \frac{g_k^T (H_k + \lambda I)(g_k - g_{k-1})}{g_{k-1}^T g_{k-1}} \qquad \text{(B.7)}$$

Once β_k is obtained from (B.7), the new CLM direction can be derived from Equation (B.4). By using Equation (B.2) instead of $-g_k$, the search efficiency can be improved. The verification of the proposed CLM method is provided in [86].

Appendix C
The Laplace Particle Swarm Training Technique

A novel Laplace Particle Swarm (LPS) technique is proposed for global optimization, which is published in [91]: "D. Li, W. Wang, and F. Ismail, Fuzzy neural network technique for system state forecasting, IEEE Transactions on Cybernetics, vol. 43, no. 5, pp. 1481-1494, 2013". Although efficiency of the proposed LPS for global optimization is verified in [91], it is time-consuming and is not suitable for this real-time IM condition monitoring application.

1. The Classical Particle Swarm Method

Before the discussion of the proposed LPS technique, a brief description will be given to the classical particle swarm (PS) method.

The classical PS method optimizes the objective function by simultaneously maintaining several candidate solutions in the search space. The accuracy of each candidate solution is represented by a fitness grade. The higher the fitness grade is, the better the candidate solution will be in optimizing the objective function. A particle starts with one candidate solution that is further evolved following certain update rules at each time step. It also records the highest fitness grade to which this particle has achieved thus far. The candidate solution corresponding to this highest fitness grade is referred to as the local best candidate solution. Among all of the local best candidate solutions, the one with the highest fitness is called the global best candidate solution [143]. In each iteration, the increment of the candidate solution in particle *i* is updated by

$$v_i(t+1) = \alpha v_i(t) + \beta_1 \gamma_1 [\tilde{x}_i(t) - x_i(t)] + \beta_2 \gamma_2 [\hat{x}(t) - x_i(t)] \tag{C.1}$$

where $v_i(t)$ is the velocity of particle *i* at time *t* and $x_i(t)$ is the position of particle *i* (candidate solution) at time *t*. α, β_1 and β_2 are user-defined parameters $\{0 \le \alpha \le 1.2,\ 0 \le \beta_1 \le 0,\ 0 \le \beta_2 \le 2\}$. γ_1 and γ_2 are random parameters over [0, 1]. $\tilde{x}_i(t)$ is the local best candidate solution of particle *i* at time *t*, and $\hat{x}(t)$ is the global best candidate solution at time *t*. Consequently each particle will be optimized by

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{C.2}$$

A lower and upper boundaries of the particle, x_{min} and x_{max}, are set to prevent a particle from moving beyond the search space. If the particle is beyond the boundaries, the velocity will be updated by

$$v_i(t) = \tau v_i(t) \tag{C.3}$$

where $\tau \in [0,1]$ is a coefficient used to adjust velocity [143].

To conduct the PS, the fitness of each particle will be evaluated first, and then local best candidate solutions and the global best candidate solution are updated. Finally, the velocity and position of each particle are updated using Equations (C.1) and (C.2), respectively [143].

Although the classical PS has some merits such as low computation complexity and easy programming [144,145], it has several limitations. For instance, it cannot search the parameter space comprehensively because the search direction of the i^{th} particle only depends on the local best candidate solution $\tilde{x}_i(t)$ and the global best candidate solution $\hat{x}(t)$, but that direction may not cover other search directions.

2. The Proposed Laplace Particle Swarm Technique

To overcome the aforementioned problems in the classical PS methods, the proposed LPS will diversify search directions so as to improve training efficiency. In order to cover the entire search space, the search direction of the i^{th} particle will be directed by the particle's local best candidate solution $\tilde{x}_i(t)$, the global best candidate solution $\hat{x}(t)$ and a random local best candidate solution $\tilde{x}_j(t)$, where $i \neq j$. The search step length is adaptively determined by a random number over the following Laplace distribution:

$$F(x) = \begin{cases} \frac{1}{2}\exp\left(\frac{x-\eta}{\omega}\right) & if\ x \leq \sigma \\ 1 - \frac{1}{2}\exp\left(\frac{\eta - x}{\omega}\right) & if\ x > \sigma \end{cases} \tag{C.4}$$

where η and ω denote the center and slope of the Laplace cumulative distribution function, respectively.

Let u be a uniformly distributed random number over [0, 1] and η = 0. The random number λ derived from Laplace distribution will be

$$\lambda = \begin{cases} \omega \ln(2u_i) & if\ u < 0.5 \\ -\omega \ln(2-2u_i) & if\ u \geq 0.5 \end{cases} \tag{C.5}$$

To guarantee convergence of the algorithm, a constraint function ξ is introduced to mitigate the impact of Laplace coefficient λ,

$$\xi = \left(1 - \frac{I_c}{I_t}\right)^b \tag{C.6}$$

where I_c and I_t are the indices of the current iteration and total number of iterations, respectively; $b \in [0.5,5]$ is the strength factor.

Given a particle $x_i(t)$ and a randomly selected local best candidate solution $\tilde{x}_j(t)$, $i \neq j$, the update expression of the velocity in the LPS will be

$$v_i(t+1) = \lambda\xi[\tilde{x}_j(t) - x_i(t)] + \beta_1\gamma_1[\tilde{x}_i(t) - x_i(t)] + \beta_2\gamma_2[\hat{x}(t) - x_i(t)] \tag{C.7}$$

where $x_i(t)$ can be updated using Equation (C.2). The verification of the proposed LPS technique is provided in [91].

Appendix D
A Mutated Particle Filter Technique for System State Estimation

The mutated particle filtering (MPF) technique is proposed for system state estimation, which is published in [135]: "D. Li, W. Wang, and F. Ismail, A mutated particle filter technique for system state estimation and battery life prediction, IEEE Transactions on Instrumentation and Measurement, vol. 63, no. 8, pp. 2034-2043, 2014". Although the proposed MPF can accurately estimate the system states, it is relatively time-consuming and is not used for this online IM health condition monitoring application.

1. Overview

In system state estimation/prognosis, the system's internal states are usually inaccessible to sensors, and the state estimation has to be inferred from noisy measurements. To make inferences about the characteristics of the measured system, a dynamic state space model is required, which usually consists of a transition model and a measurement (sensor) model. The transition model performs system state prediction, and the measurement model links the predicted states to noisy measurements. To carry out system state estimation, a posterior probability density function (pdf) has to be formulated based on system models and noisy measurements. The Kalman filter can be applied for state estimation of linear Gaussian systems; however, it lacks the capacity to address systems with nonlinear/non-Gaussian properties [124]. Although some advanced Kalman filter techniques can be used to model nonlinear systems, they apply a suboptimal implementation of the sequential Bayesian estimation framework for Gaussian random variables [125]. On the other hand, although the grid-based filter can approximate optimal Bayesian recursion by using a uniform grid to explore the entire state space, its computational complexity limits its practical applications [126].

Particle filters (PFs) can be used to model systems with nonlinear and non-Gaussian characteristics, which have been utilized in system state tracking [127,128] and prediction applications [129,130]. A PF is a recursive Monte Carlo-based method that constructs pdf using a set of random particles with associated weights. Among these available PF techniques, Sampling Importance Resampling PF (SIR-PF) is the fundamental PF technique, in which resampling is used to avoid the situation that most particles' weights are close to zero. However, SIR-PF conducts system state estimation without considering current measurement, which may degrade its performance. Although Auxiliary PF (APF)

can generate a new state estimate conditioned on current measurement with the use of some statistical indicators to characterize transition density, it suffers from the loss of diversity among particles after resampling because resampled particles are generated based on a discrete distribution, rather than a continuous one [124]. Regularized PF (RPF) can draw particles from continuous distribution to improve particle diversity [131], however its estimated continuous distribution may not be accurate if there are insufficient particles locating at high likelihood area of the distribution. Regularized auxiliary PF (RAPF) can diversify the particles drawn by sampling particles from a continuous distribution to improve system state estimation [132]; however, the high likelihood area of its posterior pdf may not be fully represented by particles after certain filtering iterations since some gaps could exist in the high likelihood area of the estimated posterior pdf. These gaps lack particles to represent them, and the information of the particles over these gaps cannot be delivered to the following iterations, which, in turn, would degrade inaccurate system state estimation. The unscented PF applies an unscented Kalman filter to generate an importance proposal distribution to improve estimation accuracy [133]; however it may still suffer from inaccurate distribution representation caused by insufficient high-weight particles. The Rao-blackwellished PF can marginalize some system states and conducts particle filtering only to remaining system states, so as to improve computation efficiency [134]; although it could speed up the filtering process, its distribution representation is not accurate.

To tackle the aforementioned problems, the MPF technique is proposed in this work to explore both the extended area of the prior distribution and the entire distribution of the posterior pdf to improve state estimation of dynamic systems [135]. In the proposed MPF, a novel mutation approach is proposed to diversify the particles and exploit the entire posterior pdf space, and a new particle selection mechanism is suggested to search the high likelihood area of the posterior pdf. The proposed MPF technique is implemented for battery remaining useful life (RUL) prediction.

2. The Mutated Particle Filter Technique

To properly introduce the proposed MPF technique, a brief introduction of the related RAPF technique is given first in Subsection D.2.1, and then the proposed mutation method and particle selection scheme in the MPF technique are discussed in Subsection D.2.2.

2.1. Regularized Auxiliary Particle Filter

To formulate a RAPF, a state space model with system states x_t and observations y_t is considered. Given an initial density $p(x_0)$, the states x_t evolve over time as a partially-observed first-order Markov process based on probability transition density $f(x_t|x_{t-1})$, where t = 1, 2, …, T, and T is the number of observations. The measurements y_t are conditionally independent, and are obtained through conditional probability density $f(y_t|x_t)$. Once the density $f(x_{t-1}|y_{1:t-1})$ at time (t–1) is projected forward in time, the prior density of the state at time instant t can be estimated as

$$f(x_t|y_{1:t-1}) = \int f(x_t|x_{t-1}) f(x_{t-1}|y_{1:t-1}) dx_{t-1} \tag{D.1}$$

Applying the Bayesian rule, the updated posterior density can be derived as

$$f(x_t|y_{1:t}) = \frac{f(y_t|x_t) f(x_t|y_{1:t-1})}{f(y_t|y_{1:t-1})} \tag{D.2}$$

where the normalizing factor is given by

$$f(y_t|y_{1:t-1}) = \int f(y_t|x_t) f(x_t|y_{1:t-1}) dx_t \tag{D.3}$$

Equation (D.2) provides an optimal recursive solution to system state estimation. If the related system is linear Gaussian, Kalman filter techniques can provide formal recursion for the density function of the system. However, nonlinear non-Gaussian systems do not have closed-form solutions, because the multi-dimensional integrals in Equations (D.1) and (D.3) are usually intractable. Hence, the pdf, $f(x_t|y_{1:t})$, can be evaluated using some suboptimal filters such as the RAPF method.

In the RAPF [132], the first stage posterior density $f(x_{t-1}|y_{1:t})$ is characterized by a set of particles (i.e., random support points) x_{t-1}^i and their associated weights π_{t-1}^i; i = 1, 2, …, N, where N is the

number of particles. The posterior density $f(x_{t-1}|y_t)$ can be estimated with the following expression [135]

$$f(x_{t-1}|y_t) \approx \sum_{i=1}^{N} \pi_{t-1}^{i} f(y_t|\mu_t^i) R_l(x_{t-1} - x_{t-1}^i) \text{ and } \sum_{i=1}^{N} \pi_{t-1}^{i} = 1 \tag{D.4}$$

where μ_t^i can be the mean, the mode, or a random draw associated with the density $f(x_t|x_{t-1})$. $R_l(\cdot)$ is the re-scaled kernel function given by

$$R_l(x) = l^{-n} R(x/l) \tag{D.5}$$

where $l > 0$ is the scalar kernel bandwidth and n is the dimension of the state vector x.

The kernel $R_l(\cdot)$ and bandwidth l are chosen to minimize the mean integrated square error between the true posterior density and the corresponding regularized empirical representation in Equation (D.4). When all the samples have the same weight [131], the optimal kernel would be the Epanechnikov kernel

$$R^* = \begin{cases} \frac{n+2}{2c_n}(1 - \|x\|^2) & \text{if } \|x\| < 1 \\ 0 & \text{otherwise} \end{cases} \tag{D.6}$$

where c_n is the volume of the n-dimensional unit hypersphere given by

$$c_n = \begin{cases} 2 & n = 1 \\ \pi & n = 2 \\ \vdots & \vdots \\ 2\pi c_{n-2}/n & n = n \end{cases} \tag{D.7}$$

According to the density estimation theory as stated in [136, 137], when the underlying density is Gaussian with a unit covariance matrix, the optimal bandwidth l^* can be determined as [131]

$$l^* = \left[8c_n^{-1}(n+4)\left(2\sqrt{\pi}\right)^n \right]^{1/(n+4)} N^{1/(n+4)} \tag{D.8}$$

where N is the number of particles.

In dealing with an arbitrary underlying density, it is necessary to assume that the density is Gaussian. Then its covariance matrix can be replaced by the empirical covariance matrix D. If M is the square root matrix of the empirical covariance matrix D such that $MM^T = D$, the kernel function in Equation (D.5) can be rewritten as

$$R_l(x) = (\det M)^{-1} l^{-n} R\left(M^{-1}x / l\right) \tag{D.9}$$

where det(M) is the determinant of matrix M.

Once x_{t-1}^j are derived using Equation (D.4), the second stage particles x_t^j can be evaluated through $f\left(x_t | x_{t-1}\right)$; the second stage weights are formulated as

$$\pi_t^j \propto \frac{f\left(y_t | x_t^j\right)}{f\left(y_t | \mu_t^j\right)} \tag{D.10}$$

where $\mu_t^j = E\left(x_t | x_{t-1}^j\right)$. In this scenario, the superscript $j \in \{1, 2, ..., N\}$ represents the index of particles after resampling, while the superscript $i \in \{1, 2, ..., N\}$ in Equation (D.4) denotes the index of particles before resampling.

2.2. The Proposed Mutation Particle Filter Technique

The RAPF in [132] employs a single statistic value μ_t^i (e.g., the mean value) to characterize $f(x_t|x_{t-1})$ in the APF. However, the approximation of $f(x_t|x_{t-1})$ may not be accurate when the process noise is large, which would lead to inaccurate estimation [124]. The large process noise problem could be tackled by using more particles to characterize $f(x_t|x_{t-1})$ rather than a single statistic value. The classical PF techniques, as well as the RAPF, only evaluate $f(x_t|x_{t-1})$ based on prior particles from the last iteration. If these particles cannot represent the prior distribution accurately or the prior distribution deviates from the posterior distribution after passing through system state models, most generated posterior particles may have very low weights, which will result in sample impoverishment. To tackle this problem, the proposed MPF technique employs the mutated particles derived from prior particles to explore a wider range of the prior distribution, rather than the range constrained by some prior particles. Its purpose is to approximate the mapping between prior pdf and posterior pdf more accurately, and exploit the posterior pdf space more comprehensively. A particle selection mechanism is suggested next to make posterior particles carry high weights.

Figure D.1(a) illustrates a typical posterior distribution with sample impoverishment. It can be seen that almost half of the generated posterior particles (blue crosses) have very low weights, and part of the high likelihood area lacks sufficient particles. Figure D.1(b) will be described in subsection D.2.2.

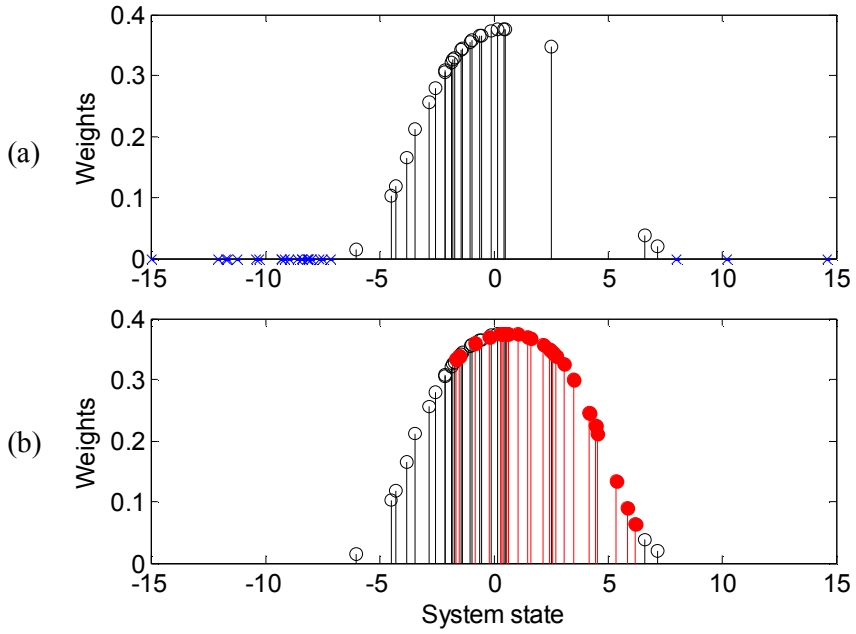

Figure D.1. Illustration of the posterior distribution of a system state: (a) without using the mutation method; blue crosses represent those posterior particles with low weight factors. (b) Using the mutation method; high weight posterior particles (red solid circles) are generated from those prior particles corresponding to low weight posterior particles.

1) The Mutation Mechanism:

The proposed mutation method is described as follows:

Given the particles at time instance t, $\{x_t^1, x_t^2, \cdots, x_t^N\}$, the respective upper and lower boundaries of these particles can be computed by

$$U\left(x_t^i\right) = max\left(x_t^1, x_t^2, \cdots, x_t^N\right) + \lambda_t \tag{D.11}$$

$$L\left(x_t^i\right) = min\left(x_t^1, x_t^2, \cdots, x_t^N\right) - \lambda_t \tag{D.12}$$

where λ_t is a positive constant, which defines the extended searching space of the prior distribution to accommodate mutated particles at time instant t.

For each particle in the particle set $\{x_t^1, x_t^2, \cdots, x_t^N\}$, a mutated particle will be derived as:

$$q = (U - x_t^i)/(x_t^i - L) \tag{D.13}$$

$$\gamma = \begin{cases} q - q\left(1 - \dfrac{r}{q}\right)^b & q \geq r \\ q + (1-q)\left(1 + \dfrac{r-q}{1-q}\right)^b & \text{otherwise} \end{cases} \tag{D.14}$$

$$\phi_t^i = (1-\gamma)L + \gamma U \tag{D.15}$$

where ϕ_t^i is the auxiliary position around x_t^i, and $r \in$ [0, 1] is a random number.

Equations (D.13)-(D.15) represent a mutation mechanism [138] that generates a random number ϕ_t^i over the feasible range of the particle set $\{x_t^1, x_t^2, \cdots, x_t^N\}$ with an approximate uniform distribution. To enable the mutated particle to deviate from the neighborhood of the original particle x_t^i, a mutated particle will be generated as

$$\hat{x}_t^i = U + L - x_t^i - \eta(\phi_t^i - x_t^i) \tag{D.16}$$

where $\eta \in [0,1]$ is a random number, $\hat{x}_t^i$ is the mutated particle, and $b \in [0.5, 1]$ is a strength factor that estimates the variance of the location of the mutated particles. The larger the factor *b* is, the wider the area in which the mutated particles may appear.

Randomly generate 10^4 possible data of the auxiliary particles ϕ_t^i and their corresponding 10^4 data of mutated particles $\hat{x}_t^i$, from a fixed value of original particle $x_t^i \in$ [0, 1] ($x_t^i = 0.2$ and 0.8 in Figure D.2). The distributions of ϕ_t^i and $\hat{x}_t^i$ can be estimated using kernel density estimation as shown in Figure D.2. It can be seen from Figure D.2 that the auxiliary particle ϕ_t^i has an approximately

uniform distribution, irrespective of the value of the original particle x_t^i; on the other hand, the mutated particle $\hat{x}_t^i$ has higher probability, located in the remote area from the original particle x_t^i.

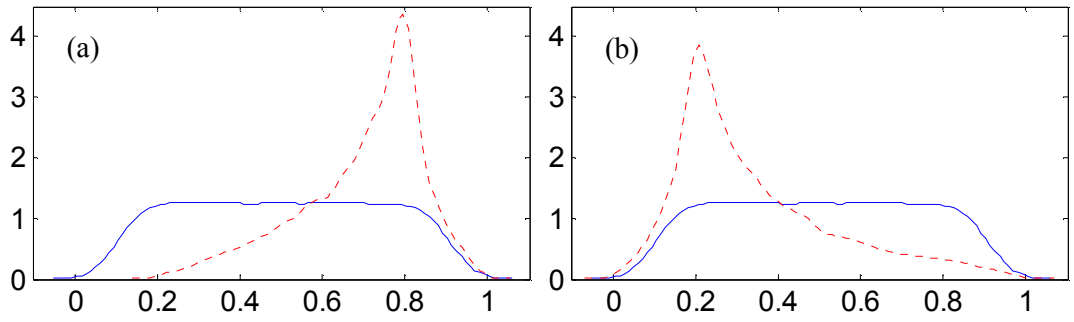

Figure D.2. The distribution of auxiliary particles ϕ_t^i and mutated particles $\hat{x}_t^i$, with the original particle (a) $x_t^i = 0.2$, and (b) $x_t^i = 0.8$. The blue solid line represents the distribution of ϕ_t^i, and the red dotted line represents the distribution of $\hat{x}_t^i$.

In using the proposed mutation approach, if one particle x_t^i lies in a low-value area of particle space, its mutated counterpart will be located in the high value region in a probabilistic form, and vice versa. The mutated particles $\{\hat{x}_t^1, \hat{x}_t^2, \cdots, \hat{x}_t^N\}$ with associated weights $\hat{\pi}_t^i$ can be used to enrich the representation of the high likelihood area of the density function $f(x_t|y_{1t})$, so as to enhance the posterior pdf and improve the system state estimation.

2) The Selection Scheme:

When posterior particles are generated, those particles with very low weights usually have less contribution to the system state estimation, while the particles with high weight may not be sufficient to represent the posterior pdf accurately. To enhance the posterior pdf representation, a threshold ξ is introduced to characterize the contribution of different particles. After the weights of all posterior particles are normalized, the particles with weights larger than (or equal to) ξ will be accepted, while those with weights less than ξ will be replaced by their mutated particles. The threshold ξ is usually a constant, which can be selected based on particular applications (e.g., $\xi = 0.01$ in this case). If a

mutated prior particle generates posterior particles with weights less than ξ, the above process will be repeated until the weight factor of the resulting new particle becomes greater than (or equal to) ξ.

By using the proposed mutation method and particle selection mechanism, the prior particles corresponding to those posterior particles with low weights can be replaced by their mutated counterparts. Consequently, the representation of the high likelihood area in the posterior distribution can be improved as demonstrated in Figure D.1(b).

3) Implementation of the MPF Technique:

In the implementation, N prior particles and their mutated N particles are used to explore the posterior pdf. By testing the derived posterior particles using the proposed selection scheme, each posterior particle with a weight less than the threshold ξ will be replaced by a posterior particle with higher weight by using the mutation mechanism. The following summarizes the implementation procedure of the proposed MPF technique:

(a) draw samples x_t^i from $f\left(x_t \middle| x_{t-1}^i\right)$; $i = 1, 2, \ldots, N$;

(b) compute the mutated particles of x_t^i, $\hat{x}_t^i$; $i = 1, 2, \ldots, N$, using Equations (D.11) – (D.16);

(c) calculate weights $\pi_t^i \propto f\left(y_t \middle| x_t^i\right)$ of these $2N$ particles (i.e., x_t^i and $\hat{x}_t^i$) and conduct normalization: $\sum_{i=1}^{2N} \pi_t^i = 1$;

(d) apply the selection scheme to the weights π_t^i. If $\pi_t^i < \xi$, a new mutated particle $\hat{x}_t^i$ will be generated from its corresponding prior particle x_t^i; this process is repeated until the weight of the generated posterior particle $\pi_t^i \geq \xi$.

(e) calculate the empirical covariance matrix D_t of $\left\{x_t^i, \pi_t^i\right\}_{j=1}^{2N}$ and the root-square matrix M_t using Cholesky decomposition $M_t M_t^T = D_t$;

(f) carry out resampling by using appropriate resampling algorithms (e.g., multinomial resampling [139]);

(g) draw ε from Epanechnikov kernel and calculate l^* using Equation (D.8). Then compute new particles [132]: $X_t^i = x_t^i + l^* M_t \varepsilon$.

The mutation method at steps (b) and (d) aims to explore an extended range of prior distribution so as to improve posterior distribution approximation. The selection scheme at step (d) will replace those posterior particles with very low weights by appropriate posterior particles. Consequently, these particles will explore the high likelihood area of the posterior pdf more thoroughly.

3. Performance Evaluation

The effectiveness of the proposed MPF technique will be evaluated in this section by simulation tests based on a benchmark model. Then it will be implemented for battery RUL prediction. The related SIR-PF and RAPF techniques will be used for comparison.

3.1. Testing Using a Benchmark Model

The following is a benchmark model that is commonly used in PF testing because of its specific properties such as nonlinear and non-Gaussian [140]:

$$x_k = \frac{1}{2} x_{k-1} + \frac{25 x_{k-1}}{1 + x_{k-1}^2} + 8\cos[1.2(k-1)] + \omega_k \tag{D.17}$$

$$y_k = \frac{1}{20} x_k^2 + \upsilon_k \tag{D.18}$$

where ω_k and υ_k are Gaussian white noise signals with zero means. The following conditions are used in this testing: the number of time steps $k = 50$, the variance of the measurement noise $\upsilon_k = 1$, the variance of the process noise $\omega_k = 10$, and the initial state $x_0 = 0.1$. Fifty particles (i.e., $N = 50$) will be used in the related PFs (i.e., SIR-PF, RAPF and MPF). In the proposed MPF, λ_t is selected as the value of standard deviation of the data; the threshold ξ is selected based on application requirements ($\xi = 0.01$ in this case); the strength factor b is determined by a trail and error process ($b = 0.8$ in this case).

To examine the parameter sensitivity of these three PFs, three different particle numbers of 50, 100 and 150, are used for simulation tests with the variance of the process noise ω_k = 1 and strength factor b = 0.8. In general, the more particles used, the higher the estimation accuracy of the PFs (but, of course, the longer time they will consume in modeling). It is seen from the simulation results listed in Table D.1 that the averaged mean of the root mean squares error (RMSE) becomes smaller as the number of particles increases in all three PFs; however the proposed MPF gives the least averaged variance when particle numbers N = 50. On the other hand, Table D.2 summarizes the simulation results of the proposed MPF with different values of strength factor b, when the process noise ω_k = 1 and the particle number N = 50. It is seen that the proposed MPF gives the best performance when b = 0.8 compared to b = 0.7 and 0.9, respectively.

Table D.1. The averaged mean and standard deviation of RMSE with different particle numbers.

Particle Numbers	Averaged mean of RMSE			Averaged standard deviation of RMSE		
	SIR-PF	RAPF	MPF	SIR-PF	RAPF	MPF
50	4.3906	3.9788	3.6303	1.1448	0.8020	0.6062
100	3.5109	3.2854	3.1531	0.9641	0.7854	0.7611
150	3.3006	3.2785	3.1393	1.6458	0.9979	0.6110

Table D.2. The averaged mean and standard deviation of RMSE with different strength factor using the proposed MPF.

Strength factor b	Averaged mean of RMSE	Averaged standard deviation of RMSE
0.7	3.7514	0.6361
0.8	3.6303	0.6062
0.9	3.7190	0.6919

Figure D.3 illustrates the test results of 30 random runs using the same observation data set, as the multinomial resampling is implemented at each time step. It can be seen that the RAPF generates less variance of the estimation error than the SIR-PF, because of its more effective particle-diversifying mechanism. However, the proposed MPF provides more accurate estimation (i.e., with less variance of the estimation error) than the RAPF, because its mutation mechanism can enrich the particle species and capture the distribution more comprehensively and accurately.

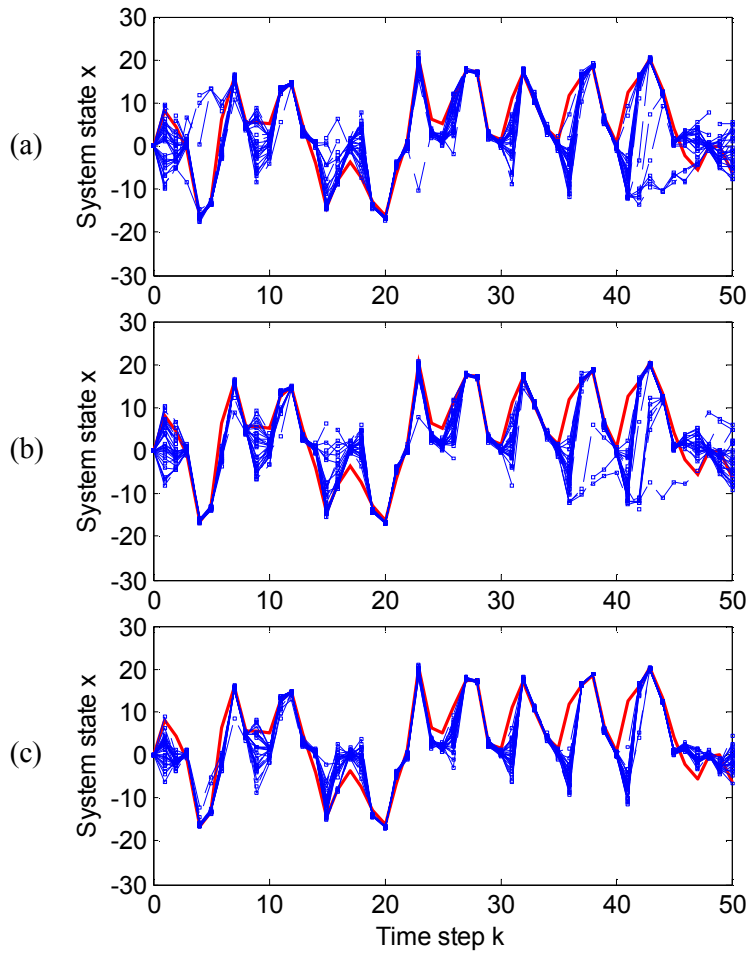

Figure D.3. Performance comparison of the related PF techniques over 30 runs by: (a) SIR-PF; (b) RAPF; (c) MPF. The red solid line represents the true states; the blue dotted lines represent the estimated states at different runs.

To further verify the effectiveness of the proposed MPF, three test scenarios corresponding to different variances ω_k = 1, 4, and 10, are tested. In each scenario, 100 data sets in total are randomly generated using Equations (D.17) and (D.18). For each data set, these three PF techniques (i.e., SIR-PF, RAPF and MPF) are tested over 100 runs. The RMSE between the true states and the estimated states are computed for each run. For comparison, the mean and standard deviation of the RMSE over 100 runs in each data set are calculated and illustrated in Figures D.4-D.6. The results summarized in Table D.3; it can be seen that the averaged mean and standard deviation of RMSE become larger as the process noise increases in all three PF estimation scenarios. The RAPF provides lower averaged mean and standard deviation of the RMSE than those of the SIR-PF, because of its diversified particle representation. The proposed MPF, however, provides the best estimation accuracy (i.e., with the lowest mean and the smallest variance) compared with both SIR-PF and RAPF. This is because the MPF can approximate the distribution more accurately, and explore the distribution more thoroughly.

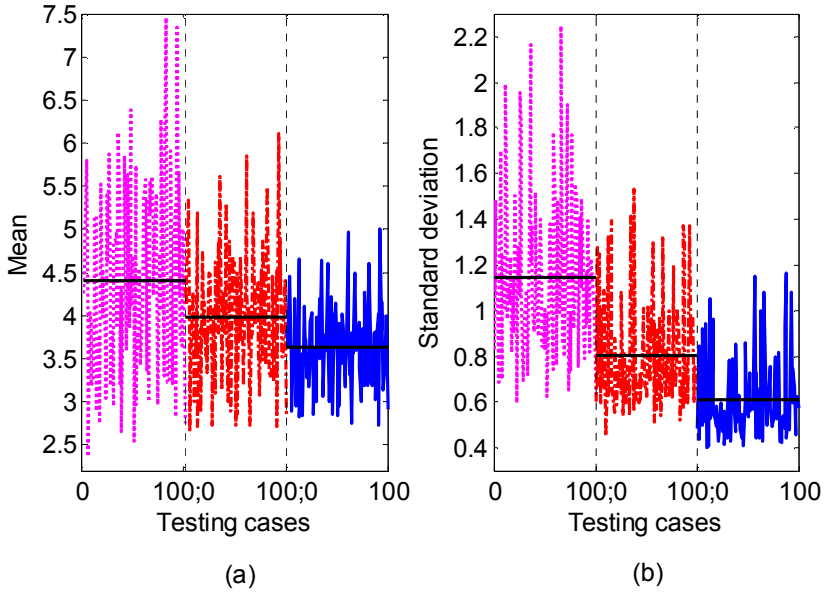

Figure D.4. Test results of three PFs when the variance of the process noise is 1: (a) mean of RMSE, (b) standard deviation of RMSE. The pink dashed line represents the results of SIR-PF; the red dash dotted line represents the results of RAPF; the blue solid line represents the results of the proposed MPF. The black solid line denotes the mean of the data.

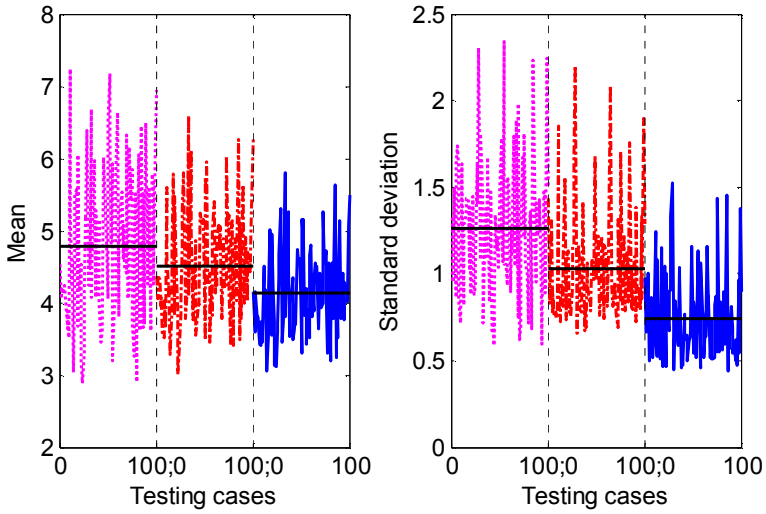

Figure D.5. Test results of three PFs when the variance of the process noise is 4: (a) mean of RMSE, (b) standard deviation of RMSE. The pink dashed line represents the results of SIR-PF; the red dash dotted line represents the results of RAPF; the blue solid line represents the results of the proposed MPF. The black solid line denotes the mean of the data.

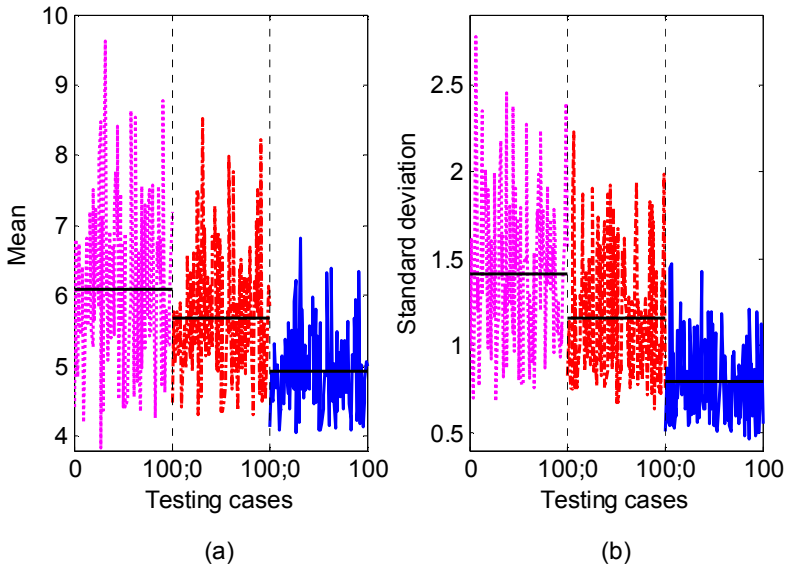

Figure D.6. Test results of three PFs when the variance of the process noise is 10: (a) mean of RMSE, (b) standard deviation of RMSE. The pink dashed line represents the results of SIR-PF; the red dash dotted line represents the results of RAPF; the blue solid line represents the results of the proposed MPF. The black solid line denotes the mean of the data.

Table D.3. The averaged mean and standard deviation of RMSE over 100 data sets.

Variance of process noise	Averaged mean of RMSE			Averaged standard deviation of RMSE		
	SIR-PF	RAPF	MPF	SIR-PF	RAPF	MPF
1	4.3906	3.9788	3.6303	1.1448	0.8020	0.6062
4	4.7744	4.5208	4.1420	1.2614	1.0315	0.7452
10	6.0968	5.6914	4.9356	1.4144	1.1583	0.7971

It should also be realized that, after several iterations of implementing PF, gaps may develop in the high likelihood area of the estimated posterior pdf, which is an indication of information loss. These gaps may degrade the estimation accuracy in the current iteration, which can, in turn, propagate to the subsequent iterations and continue to degrade estimation accuracy. By using the proposed mutation method, the pdf can be explored more thoroughly, especially in the pdf gaps, and consequently the estimation accuracy can be improved.

3.2. Battery Remaining Useful Life Prediction

Lithium-ion batteries are widely used in industrial and domestic applications. An effective prognostic tool is very useful to predict the future state of the battery, so as to diagnose the battery's health condition, and estimate its RUL information. Reliable RUL information is critically needed in many applications, such as electric vehicles and aircraft, to schedule battery recharging operations and prevent malfunction of the related equipment.

To test the proposed MPF technique for battery RUL prediction, the data sets of battery #6 and battery #18 from the NASA Prognostics Center of Excellence are employed [141]. Figure D.7, adopted from [142], schematically illustrates the battery prognostics tests. The load bank and environmental chamber were used to adjust load and environmental conditions of the battery cells, respectively. The electrochemical impedance spectroscopy (EIS) measurement was conducted in the battery health monitoring module (BHM), which was used to measure battery impedance. The three states of batteries (i.e., charge, discharge, and impedance) were controlled by a switching circuitry. The sensors' signals were collected by a data acquisition (DAQ) system. The aim of the experiments

was to identify different health states of battery cells with similar terminal voltages, and then to predict the RUL of the cells.

To generate these data sets, a lithium-ion battery was run through three different operational profiles: charge, discharge, and impedance, at room temperature. The charge process was conducted by feeding a 1.5A constant current to the battery until its voltage reached 4.2V, then the charge process continued in a constant voltage mode, until the charge current dropped to 20mA. The discharge process was carried out at a constant current mode (2A) until the battery voltage dropped to 2.5V. The aging process of batteries can be accelerated by repeatedly charging and discharging the batteries. The experiments were stopped when the battery reached end-of-life criterion, which was set at 30% fade in rated capacity.

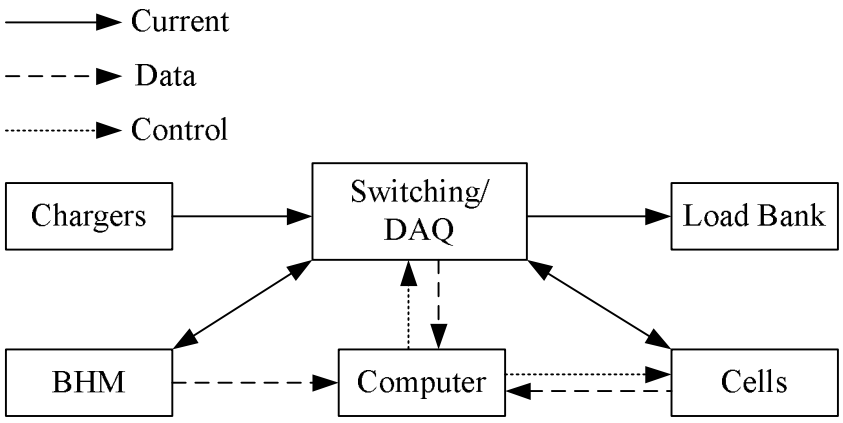

Figure D.7. Schematic illustration of the battery prognostics tests, adopted from [142]. BHM is the battery health monitoring modules and DAQ is the data acquisition system.

Figure D.8 shows the lumped-parameter model, where R_E and R_{CT} represent the electrolyte resistance and charge transfer resistance, respectively; R_W is the Warburg impedance, and C_{DL} is the dual layer capacitance. Although the battery capacity is usually inaccessible for measurement, R_E + R_{CT} is inversely proportional to the capacity C/1, and can be used to predict battery capacity drop [135]. R_E and R_{CT} can be estimated from EIS tests. The battery model can be formulated as

$$\Gamma_k = \Gamma_{k-1} + v_k \tag{D.19}$$

$$\Psi_k = \Psi_{k-1} \exp(\Gamma_k k) + w_k \tag{D.20}$$

$$Y_k = \Psi_{k-1} + \rho_k \tag{D.21}$$

where Ψ_k is the state vector (i.e., R_E or R_{CT}) at time step k, Γ_k is the exponential growth model parameter, and Y_k is the measurement vector containing battery parameters inferred from measured data. The state vector Ψ_1 at the first time step takes the initial value of R_E or R_{CT}. The exponent Γ_1 can be calculated from training data using least square estimate. v_k, w_k and ρ_k are Gaussian noise signals. The data sets of R_E, R_{CT} and C in battery #6 and battery #18 are smoothed to improve the RUL prediction using the battery model represented by Equations (D.19)-(D.21). The approximate linear relationship between $R_E + R_{CT}$ and $C/1$ in battery #6 is shown in Figure D.9.

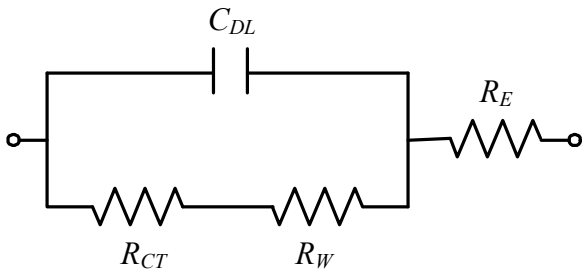

Figure D.8. Lumped-parameter model of a Lithium-ion battery. R_E and R_{CT} represent the respective electrolyte resistance and charge transfer resistance; R_W is the Warburg impedance and C_{DL} is the dual layer capacitance.

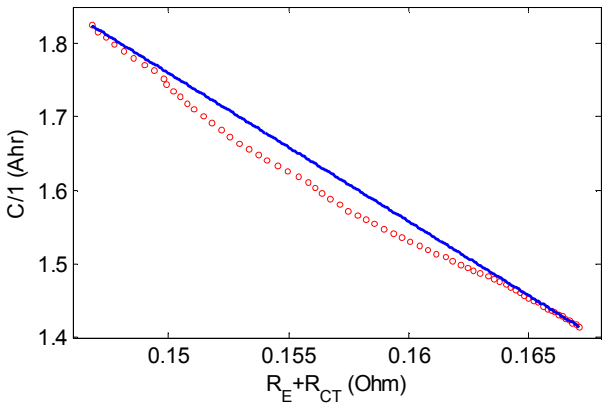

Figure D.9. The relationship between R_E+R_{CT} and $C/1$ in battery #6. The red circles represent measured R_E+R_{CT} versus $C/1$; the blue solid line is a linear fit.

The SIR-PF, RAPF and the proposed MPF techniques were implemented for battery state estimation and RUL prediction. The first part of the trajectory (i.e., R_E or R_{CT}) was employed to estimate the battery model parameters. Then the identified model for each technique was applied to predict the remaining part of the trajectory. In each iteration, 1000 particles were used to estimate the posterior pdf. The time to trigger the prediction depends on application requirements. Figure D.10 shows both the state tracking and the future state prediction of data R_E and $R_{CT,}$ with the prognosis starting at cycle 30. In Figure D.10, the system model is identified in the estimation period; multiple-steps-ahead forecast is conducted in the prediction period; that is, the n^{th} estimated data in the prediction period is the n-steps-ahead forecast using the identified model. It is seen that RAPF outperforms SIR-PF in both R_E and R_{CT} predictions, while the MPF provides even better prediction performance than the RAPF in these two prediction scenarios (R_E and R_{CT}). By a linear transformation, the tracking of the capacity C/1 is shown in Figure D.11; it is seen that RAPF generates less RUL prediction error (15.69 cycles) than SIR-PF (17.39 cycles) because of its diversified resampled particles. The MPF yields the minimum RUL prediction error (10.01 cycles) compared with SIR-PF and RAPF techniques, because it can explore the entire distribution with the help of the mutated particles.

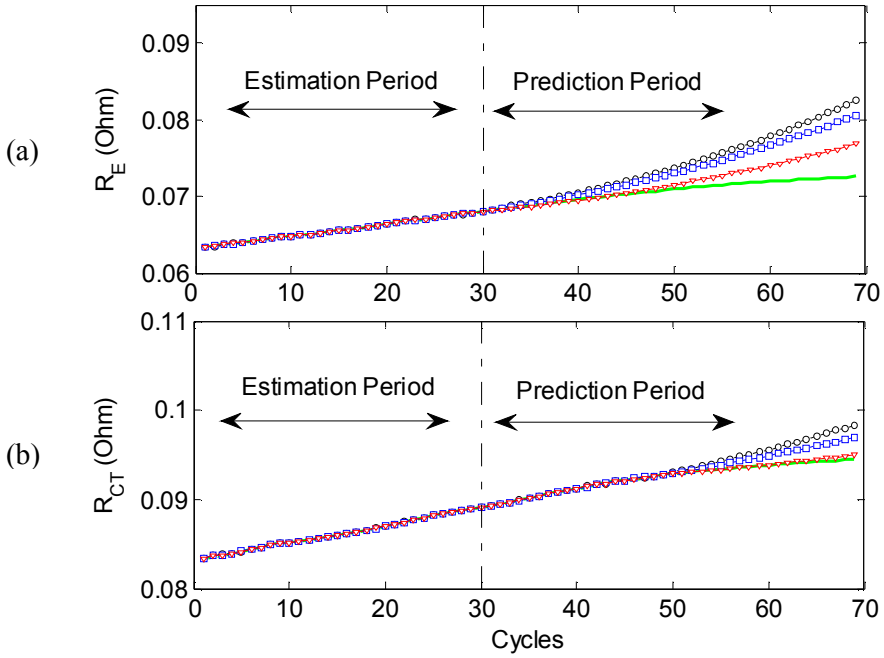

Figure D.10. State tracking and future state prediction at cycle 30 for the battery parameters (a) R_E and (b) R_{CT} using three PFs: SIR-PF (○ - black line), RAPF (□ - blue line) and MPF (∇ -red line). The green solid line is the true states of R_E and R_{CT}.

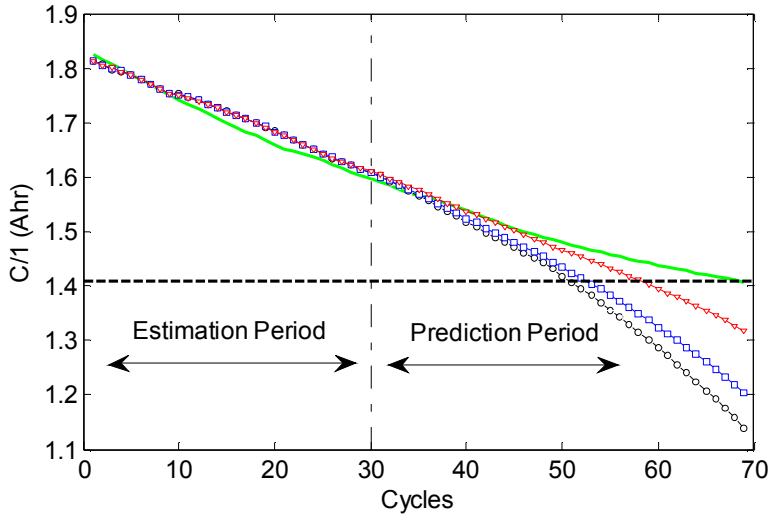

Figure D.11. Battery RUL prediction at cycle 30 using three different PFs: SIR-PF (○ - black line), RAPF (□ - blue line) and MPF (∇ -red line). The green solid line is the real measurement of *C*/1.

Figure D.12 shows the prognostics results of the battery parameters R_E and R_{CT} when the prediction is triggered at cycle 54. To clarify the forecasting distinction, only the estimation period from cycle 40 to cycle 54 and prediction period from cycle 55 to cycle 69 are shown in Figure D.12. The prediction performances of these three PFs (i.e., SIR-PF, RAPF and MPF) become more accurate in this case, because more data are used for model parameter estimation. It can be seen from Figure D.12 that the proposed MPF can predict the trend of R_E and R_{CT} more accurately than both RAPF and SIR-PF, thanks to its advanced mutation mechanism. Figure D.13 shows the prediction of the corresponding capacity $C/1$; it can be seen that the MPF yields the minimum RUL prediction error (0.58 cycles). On the other hand, at the end of the prediction, RAPF and SIR-PF cannot reach end-of-life threshold, and generate large errors of 0.008Ahr and 0.026Ahr, respectively, at the last cycle. The MPF outperforms both RAPF and SIR-PF because of its efficient mutation mechanism to diversify the particles and to enhance model parameter identification.

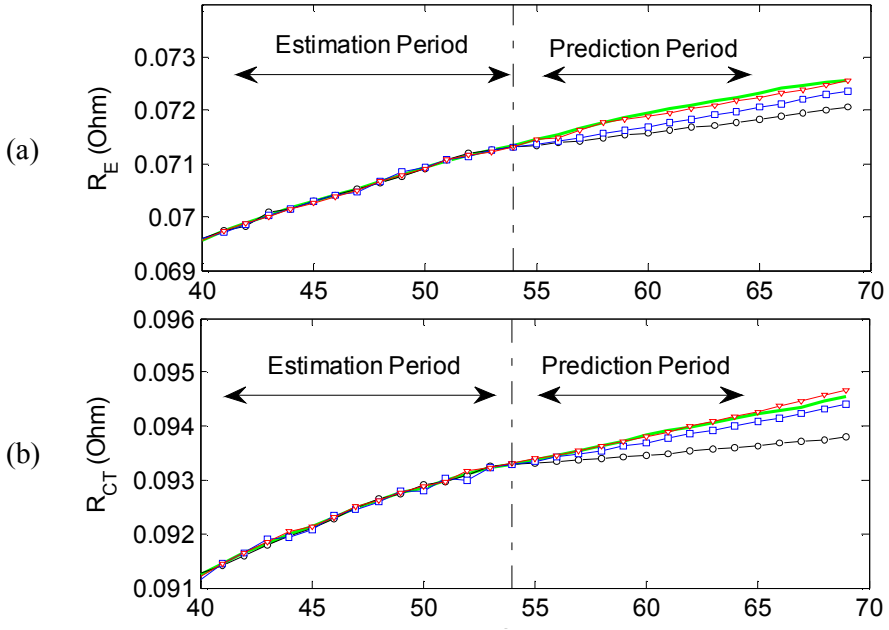

Figure D.12. State tracking and future state prediction at cycle 54 for the battery parameters (a) R_E and (b) R_{CT} using three PFs: SIR-PF (○ - black line), RAPF (□ - blue line) and MPF (∇ -red line). The green solid line is the true states of R_E and R_{CT}.

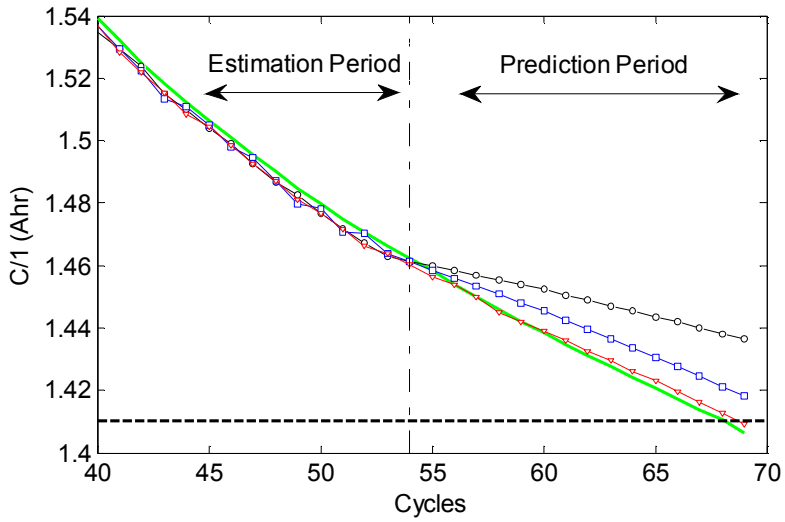

Figure D.13. Battery RUL prediction at cycle 54 using three different PFs: SIR-PF (○ - black line), RAPF (□ - blue line) and MPF (∇ -red line). The green solid line is the real measurement of *C*/1.

The uncertainty of the classical PF algorithms can be represented by the pdf of the estimated system states [139]. The pdfs of the estimated state (i.e., capacity) using SIR-PF, RAPF and MPF are demonstrated in Figure D.14, where their mean values are all set as zero to compare their spread (or uncertainty). Figure D.14a shows the pdfs of these three PFs at the cycle when their mean values just reach failure threshold as illustrated in Figure D.11. Figure D.14b illustrates the pdfs of three PFs at the cycle when their mean values just reach failure threshold (i.e., MPF) or the end of prediction cycle (i.e., SIR-PF and RAPF) as illustrated in Figure D.13. It is seen that the MPF outperforms both SIR-PF and RAPF with least uncertainty because of its capability to fully explore high likelihood area and prevent the possible deviations by low weight particles.

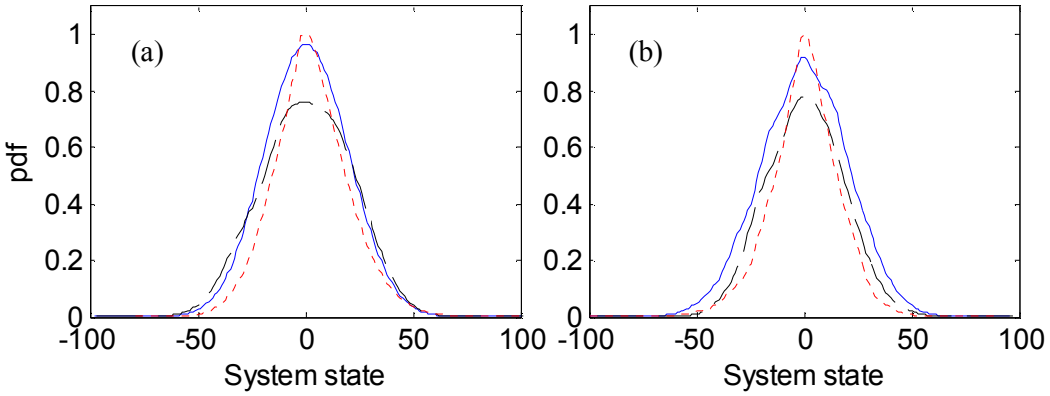

Figure D.14. Uncertainty representation of Battery RUL estimation corresponding to (a) prediction at cycle 30 and (b) prediction at cycle 54, corresponding to pdf of the SIR-PF (blue solid line), the RAPF (black dashed line), and the MPF (red dotted line).

The approximate linear relationship between $R_E + R_{CT}$ and $C/1$ in battery #18 is illustrated in Figure D.15. Figure D.16 shows the prognosis of the battery parameters R_E and R_{CT} when the prediction is triggered at cycle 5. It is seen from Figure D.16 that the proposed the MPF outperforms both the RAPF and the SIR-PF in predicting R_E and R_{CT} because of its effective posterior pdf estimation using the proposed mutation. Figure D.17 demonstrates the prediction of the corresponding capacity $C/1$; it is seen that the MPF yields less RUL prediction error (0.34 cycles) than that of RAPF (1.21 cycles) and SIR-PF (1.78 cycles), when they reach the end-of-life criterion. The MPF outperforms both the RAPF and the SIR-PF, because its effective mutation can improve posterior pdf estimation so as to enhance model identification.

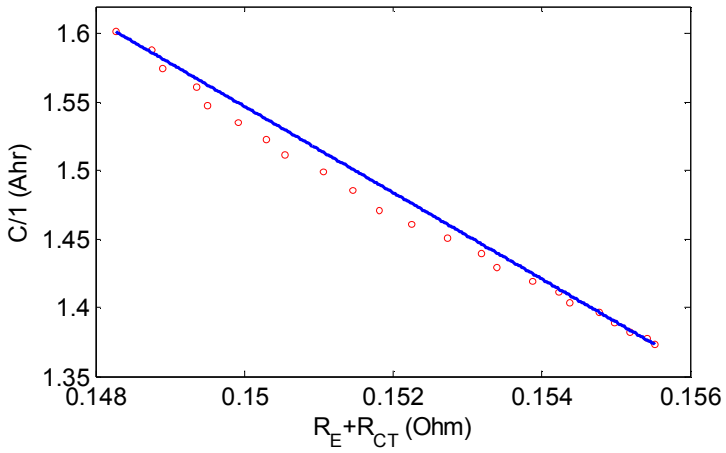

Figure D.15. The relationship between R_E+R_{CT} and C/1 in battery #18. The red circles represent measured R_E+R_{CT} versus C/1; the blue solid line is a linear fit.

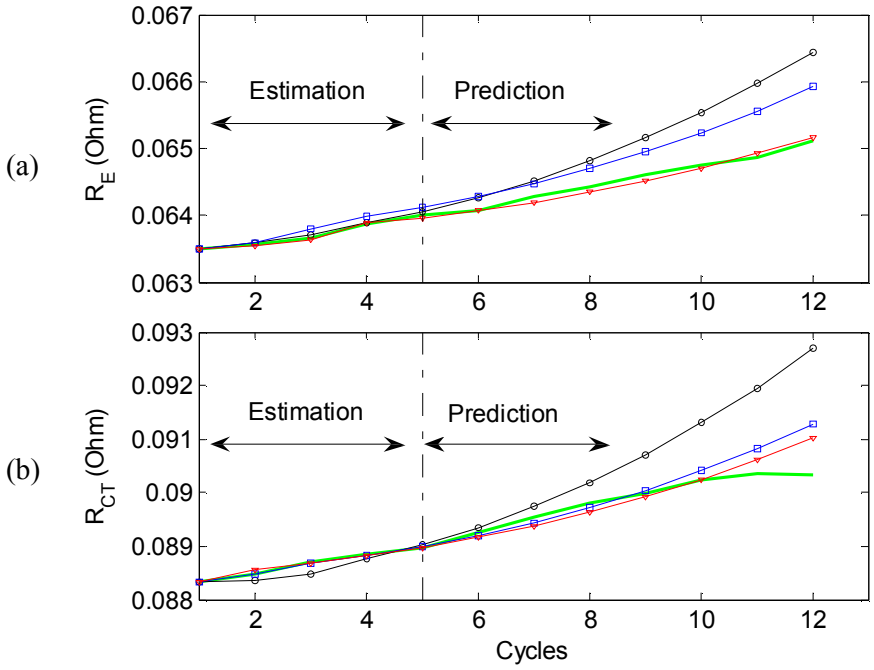

Figure D.16. State tracking and future state prediction at cycle 5 for the battery parameters (a) R_E and (b) R_{CT} using three PFs: SIR-PF (○ - black line), RAPF (□ - blue line) and MPF (∇ -red line). The green solid line is the true states of R_E and R_{CT}.

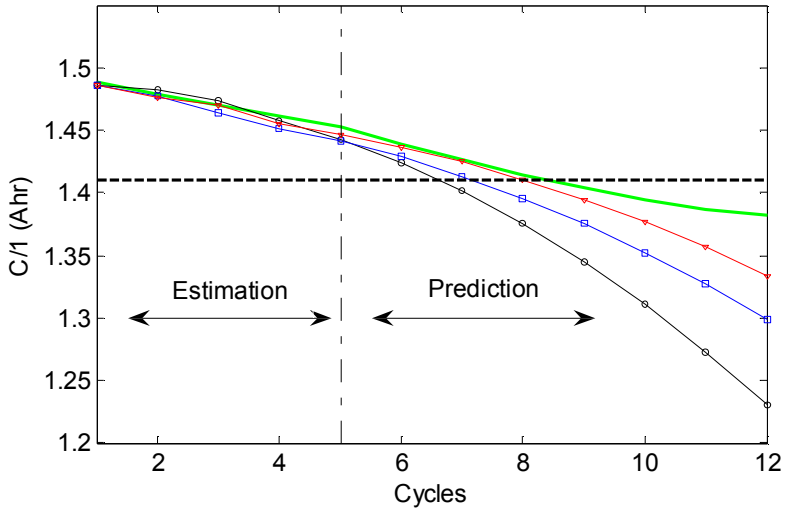

Figure D.17. Battery RUL prediction at cycle 5 using three different PFs: SIR-PF (○ - black line), RAPF (□ - blue line) and MPF (∇ -red line). The green solid line is the real measurement of *C*1.

References

[1] R. Fei, E. F. Fuchs, and H. Huag, "Comparison of two optimization techniques as applied to three-phase induction motor design," *IEEE Transaction on Energy Conversion*, vol. 4, no. 4, pp. 651-660, 1989.

[2] P. J. Tavner, Review of condition monitoring of rotating electrical machines, *IET Electrical Power Applications*, vol. 2, no. 4, pp. 215-247, 2008.

[3] Retrieved from the website: http://www.palawatr.co.th/2012/?q=content/brand-product-detail&nid=135.

[4] O. I. Okoro, "Steady and transient states thermal analysis of a 7.5-kW squirrel-cage induction machine at rated-load operation," *IEEE Transactions on Energy Conversion*, vol. 20, no. 4, pp. 730-736, 2005.

[5] A. J. Ellison and S. J. Yang, "Effects of rotor eccentricity on acoustic noise from induction machines," *Proceedings of IEE*, vol. 118, no. 1, pp. 174-184, 1971.

[6] R. D. Widdle Jr., C. M. Krousgrill Jr., and S. D. Sudhoff, "An induction motor model for high-frequency torsional vibration analysis," *Journal of Sound and Vibration*, vol. 290, no. 3-5, pp. 865-881, 2006.

[7] H. Ocak, and K. A. Loparo, "Estimation of the running speed and bearing defect frequencies of an induction motor from vibration data," *Mechanical Systems and Signal Processing*, vol. 18, no. 3, pp. 515-533, 2004.

[8] N. Tandon, G. S. Yadava, and K. M. Ramakrishna, "A comparison of some condition monitoring techniques for the detection of defect in induction motor ball bearings," *Mechanical Systems and Signal Processing*, vol. 21, no. 1, pp. 244-256, 2007.

[9] G. K. Singh, S. A. K. S. Ahmed, "Vibration signal analysis using wavelet transform for isolation and identification of electrical faults in induction machine," *Electric Power Systems Research*, vol. 68, no. 2, pp. 119-136, 2004.

[10] M. EI H. Benbouzid, "A review of induction motors signature analysis as a medium for faults detection," *IEEE Transactions on Industrial Electronics*, vol. 47, no. 5, pp. 984-993, 2000.

[11] S. Nandi, H. A. Toliyat and X. Li, "Condition monitoring and fault diagnosis of electrical motors – a review," *IEEE Transactions on Energy Conversion*, vol. 20, pp. 719-729, 2005.

[12] P. Vas, Parameter Estimation, Condition Monitoring, and Diagnosis of Electrical Machines, Clarendon Press Oxford, 1993.

[13] W. T. Thomson and M. Fenger, "Current signature analysis to detect induction motor faults," *IEEE Industry Application Magazine*, vol. 7, no. 4, pp. 26-34, 2001.

[14] N. M. Elkasabgy, A. R. Eastham, and G. E. Dawson, "Detection of broken bars in the cage rotor on an induction machines," *IEEE Transactions on Industrial Applications*, vol. 22, no. 6, pp. 165-171, 1992.

[15] H. Arabaci and O. Bilgin, "The detection of rotor faults by using short time fourier transform," *IEEE 15th Signal Processing and Communications Applications*, pp. 1-4, 2007.

[16] A. Ambardar, Analog and Digital Signal Processing, Brooks/Cole Publishing Company, 1999.

[17] J. A. Daviu, M. R. Guasp, J. P. Llinares, J. Park, S. B. Lee, J. Yoo and C. Kral, "Detection of broken outer-cage bars for double-Cage induction motors under the startup transient," *IEEE Transactions on Industry Applications*, vol. 48, no. 5, pp. 1539-1548, 2012.

[18] A. Sadeghian, Z. Ye and B. Wu, "Online detection of broken rotor bars in induction motors by wavelet packet decomposition and artificial neural networks," *IEEE Transactions on Instrumentation and Measurement*, vol. 58, no. 7, pp. 2253-2263, 2009.

[19] M. Pineda-Sanchez, M. Riera-Guasp, J. Roger-Folch, J. A. Antonino-Daviu, J. Perez-Cruz, and R. Puche-Panadero, "Diagnosis of induction motor faults in time-varying conditions using the polynomial-phase transform of the current," *IEEE Transactions on Industrial Electronics*, vol. 58, no. 4, pp. 1428-1439, 2011.

[20] M. Riera-Guasp, M. Pineda-Sanchez, J. Perez-Cruz, R. Puche-Panadero, J. Roger-Folch, and J. A. Antonino-Daviu, "Diagnosis of induction motor faults via gabor analysis of the current in transient regime," *IEEE Transactions on Instrumentation and Measurement*, vol. 61, no. 6, pp. 1583-1596, 2012.

[21] N. Mehala, Condition Monitoring and Fault Diagnosis of Induction Motor Using Motor Current Signature Analysis, Thesis, 2010.

[22] B. Akin, S. Choi, U. Orguner, and H. A. Toliyat, "A simple real-time fault signature monitoring tool for motor-drive-embedded fault diagnosis systems," *IEEE Transactions on Industrial Electronics*, vol. 58, no. 5, pp. 1990-2001, 2011.

[23] A. Soualhi, G. Clerc, and H. Razik, "Detection and diagnosis of faults in induction motor using an improved artificial ant clustering technique," *IEEE Transactions on Industrial Electronics*, vol. 60, no. 9, pp. 4053-4062, 2013.

[24] P. Eschmann, L. Hasbargen and K. Weigand, Ball and Roller Bearings: Their Theory, Design and Application, London: K G Heyden, 1958.

[25] M. Blodt, P. Granjon, B. Raison, and G. Rostaing, "Models for bearing damage detection in induction motors using stator current monitoring," *IEEE Transactions on Industrial Electronics*, vol. 55, no. 4, pp. 1813-1822, 2008.

[26] V. T. Tran, F. A. Thobiani, A. Ball, and B. K. Choi, "An application to transient current signal based induction motor fault diagnosis of Fourier-Bessel expansion and simplified fuzzy ARTMAP," *Expert Systems with Applications*, vol. 40, no. 13, pp. 5372-5384, 2013.

[27] M. E. H. Benbouzid, M. Vieira and C. Theys, "Induction motors' fault detection and localization using stator current advanced signal processing techniques," *IEEE Transactions on Power Electronics*, vol. 14, no. 1, pp. 14-22, 1999.

[28] M. J. Devaney and L. Eren, "Detecting motor bearing faults," *IEEE Instrumentation & Measurement Magazine*, vol. 7, no. 4, pp. 30-50, 2004.

[29] P. Konar and P. Vhattopadhyay, "Bearing fault detection of induction motor using wavelet and support vector machines," *Applied Soft Computing*, vol. 11, no. 6, pp. 4203-4211, 2011.

[30] L. Eren and M. J. Devaney, "Bearing damage detection via wavelet packet decomposition of the stator current," *IEEE Transactions on Instrumentation and Measurement*, vol. 53, no. 2, pp. 431-436, 2004.

[31] E. C. C. Lau and H. W. Ngan, "Detection of motor bearing outer raceway defect by wavelet packet transformed motor current signature analysis," *IEEE Transactions on Instrumentation and Measurement*, vol. 59, no. 10, pp. 2683-2690, 2010.

[32] Y. Pan, J. Chen and L. Guo, "Robust bearing performance degradation assessment method based on improved wavelet packet support vector data description," *Mechanical Systems and Signal Processing*, vol. 23, no. 3, pp. 669-681, 2009.

[33] L. Frosini, and E. Bassi, "Stator current and motor efficiency as indicators for different types of bearing faults in induction motors," *IEEE Transactions on Industrial Electronics*, vol. 57, no. 1, pp. 244-251, 2010.

[34] W. Zhou, B. Lu, T. G. Habetler, and R. G. Harley, "Incipient bearing fault detection via motor stator current noise cancellation using Wiener filter," *IEEE Transactions on Industry Applications*, vol. 45, no. 4, pp. 1309-1317, 2009.

[35] R. J. Romero-Troncoso, R. Saucedo-Gallaga, E. Cabal-Yepez, A. Garcia-Perez, R. A. Osornio-Rios, R. Alvarez-Salas, H. Miranda-Vidales, and N. Huber, "FPGA-based online detection of multiple combined faults in induction motors through information entropy and fuzzy inference," *IEEE Transactions on Industrial Electronics*, vol. 58, no. 11, pp. 5263-5270, 2011.

[36] M. Pineda-Sanchez, R. Puche-Panadero, M. Riera-Guasp, J. Perez-Cruz, J. Roger-Folch, J. Pons-Llinares, V. Climente-Alarcon, and J. A. Antonino-Daviu, "Application of the Teager-Kaiser energy operator to the fault diagnosis of induction motors," *IEEE Transactions on Energy Conversion*, vol. 28, no. 4, pp. 1036-1044, 2013.

[37] L. Devroye, L. Gyorfi and G. Lugosi, A Probabilistic Theory of Pattern Recognition, Berlin: Springer-Verlag, 1996.

[38] T. F. Li and T. C. Yen, "A Bayes empirical Bayes decision rule for classification," *Communications in Statistics –Theory and Methods*, vol. 35, no. 4, pp. 1137-1149, 2005.

[39] G. P. Zhang, "Neural networks for classification: a survey," *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, vol. 30, no. 4, pp. 451-462, 2000.

[40] B. Ayhan, C. Mo-Yuen and S. Myung-Hyun, "Multiple discriminant analysis and neural network based monolith and partition fault detection schemes for broken rotor bar in induction motors," *IEEE Transactions on Industrial Electronics*, vol. 53, no. 4, pp. 1298-1308, 2006.

[41] X. Jin, M. Zhao, T. W. S. Chow, and M. Pecht, "Motor bearing fault diagnosis using trace ratio linear discriminant analysis," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 5, pp. 2441-2451, 2014.

[42] S. P. Curram and J. Mingers, "Neural networks, decision tree induction and discriminant analysis: an empirical comparison," *The Journal of the Operational Research Society*, vol. 45, no. 4, pp. 440-450, 1994.

[43] X. Huang, T. G. Habetler and R. G. Harley, Detection of Rotor Eccentricity Faults in a Closed Loop Drive Connected Induction Motor Using an Artificial Neural Network, IEEE Transactions on Power Electronics, 22(4), 2007, 1552-1559.

[44] M. Seera, C. P. Lim, D. Ishak and H. Singh, Fault Detection and Diagnosis of Induction Motors Using Motor Current Signature Analysis and a Hybrid FMM-CART Model, IEEE Transactions on Neural Networks and Learning Systems, 23(1), 2012, 97-108.

[45] V. N. Ghate and S. V. Dudul, Cascade Neural Network Based Fault Classifier for Three Phase Induction Motor, IEEE Transactions on Industrial Electronics, 58(5), 2011, 1555-1563.

[46] J. S. R. Jang, C. T. Sun and E. Mizutani, Neuro-Fuzzy and Soft Computing, Prentice Hall, New Jersey, 1997.

[47] G. Yu, W. Xue, and Y. Zhou, "A nonmonotone adaptive projected gradient method for primal-dual total variation image restoration," *Signal Processing*, vol. 103, pp. 242-249, 2014.

[48] H. Guo, "Analysis of split weighted least-squares procedures for pseudo-hyperbolic equations," *Applied Mathematics and Computation*, vol. 217, no. 8, pp. 4109-4121, 2010.

[49] W. X. Zhao, and T. Zhou, "Weighted least squares based recursive parametric identification for the submodels of a PWARX system," *Automatica*, vol. 48, no. 6, pp. 1190-1196, 2012.

[50] J. K. Liu, and S. J. Li, "New hybrid conjugate gradient method for unconstrained optimization," *Applied Mathematics and Computation*, vol. 245, pp. 36-43, 2014.

[51] A. M. Sajo-Castelli, M. A. Fortes, and M. Raydan, "Preconditioned conjugate gradient method for finding minimal energy surfaces on Powell-Sabin trangulations," *Journal of Computational and Applied Mathematics*, vol. 268, pp. 34-55, 2014.

[52] J. C. Rodriguez-Quinonez, O. Sergiyenko, F. F. Gonzalez-Navarro, L. Basaca-Preciado, and V. Tyrsa, "Surface recognition improvement in 3D medical laser scanner using Levenberg-Marquardt method," *Signal Processing*, vol. 93, no. 2, pp. 378-386, 2013.

[53] M. Hsie, Y. F. Ho, C. T. Lin and I. C. Yeh, "Modeling asphalt pavement overlay transverse cracks using the genetic operation tree and Levenberg-Marquardt method," *Expert Systems with Applications*, vol. 39, no. 5, pp. 4874-4881, 2012.

[54] A. Almaksour, E. Anquetil, S. Quiniou, and M. Cheriet, "Evolving fuzzy classifiers: application to incremental learning of handwritten gesture recognition system," *2010 International Conference on Pattern Recognition*, pp. 4056-4059, 2010.

[55] P. Angelov, X. Zhou, and F. Klawonn, "Evolving fuzzy rule-based classifiers," *Proceedings of the 2007 IEEE Symposium on Computational Intelligence in Image and Signal Processing*, pp. 220-225, 2007.

[56] R. D. Baruah, P. Angelov, and J. Andreu, "Simpl_eClass: simplified potential-free evolving fuzzy rule-based classifiers," *IEEE International Conference on Systems, Man, and Cybernetics*, pp. 2249-2254, 2011.

[57] R. Schapire and Y. Singer, "Improved boosting algorithms using confidence-rated predictions," *Machine Learning*, vol. 37, pp. 297-336, 1999.

[58] A. Takemura, A. Shimizu and K. Hamamoto, "Discrimination of breast tumors in ultrasonic images using an ensemble classifier based on the AdaBoost algorithm with feature selection," *IEEE Transactions on Medical Imaging*, vol. 29, no. 3, pp. 598-609, 2010.

[59] J. H. Morra, Z. Tu, L. G. Apostolova, A. E. Green, A. W. Toga and P. M. Thompson, "Comparison of AdaBoost and support vector machines for detecting Alzheimer's disease through automated hippocampal segmentation," *IEEE Transactions on Medical Imaging*, vol. 29, no. 1, pp. 30-43, 2010.

[60] C. Kyrkou and T. Theocharides, "A flexible parallel hardware architecture for AdaBoost-based real-time object detection," *IEEE Transactions on Very Large Scale Integration Systems*, vol. 19, no. 6, pp. 1034-1047, 2011.

[61] X. Baro, S. Escalera, J. Vitria, O. Pujol, P. Radeva, "Traffic sign recognition using evolutionary AdaBoost detection and forest-ECOC classification," *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 1, pp. 113-126, 2009.

[62] W. Hu, W. Hu and S. Maybank, "AdaBoost-based algorithm for network intrusion detection," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 38, no. 2, pp. 577-583, 2008.

[63] T. Dietterich, "An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting, and randomization," *Machine Learning*, vol. 40, no. 2, pp. 139-157, 2000.

[64] G. C. Cawley and N. L. C. Talbot, "On over-fitting in model selection and subsequent selection bias in performance evaluation," *Journal of Machine Learning Research*, vol. 11, pp. 2079-2107, 2010.

[65] H. Masnadi-Shirazi and N. Vasconcelos, "Cost-sensitive boosting," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 33, no. 2, pp. 294-309, 2011.

[66] L. Landesa-Vazquez and J. L. Alba-Castro, "Double-base asymmetric AdaBoost," *Neurocomputing*, vol. 118, pp. 101-114, 2013.

[67] Y. Gao, G. Ji, Z. Yang and J. Pan, "A dynamic AdaBoost algorithm with adaptive changes of loss function," *IEEE Transactions on System, Man, and Cybernetics – Part C: Applications and Reviews*, vol. 42, no. 6, pp. 1828-1841, 2012.

[68] J. Friedman, T. Hastie and R. Tibshirani, "Additive logistic regression: a statistical view of boosting," *The Annals of Statistic*s, vol. 28, no. 2, pp. 337-407, 2000.

[69] G. Ratsch, T. Onoda and K. R. Muller, "Soft margins for AdaBoost," *Machine Learning*, vol. 42, pp. 287-320, 2001.

[70] J. Cao, S. Kwong and R. Wang, "A noise-detection based AdaBoost algorithm for mislabeled data," *Pattern Recognition*, vol. 45, pp. 4451-4465, 2012.

[71] Y. Gao and F. Gao, "Edited AdaBoost by weighted kNN," *Neurocomputing*, vol. 73, no. 16-18, pp. 3079-3088, 2010.

[72] K. Mohammadi, H. R. Eslami and R. Kahawita, "Parameter estimation of an ARMA model for river flow forecasting using goal programming," *Journal of Hydrology*, vol. 331, no. 1, pp. 293-299, 2006.

[73] E. Erdem and J. Shi, "ARMA based approaches for forecasting the Tuple of wind speed," *Applied Energy*, vol. 88, no. 4, pp. 1405-1414, 2011.

[74] J. Nowicka-Zagrajek and R. Weron, "Modeling electricity load in Californis: ARMA models with hyperbolic noise," *Signal Processing*, vol. 82, no. 12, pp. 1903-1915, 2002.

[75] R. B. C. Benitez, R. B. C. Paredes, G. Lodewijks, and J. L. Nabais, "Damp trend grey model forecasting method for airline industry," *Expert Systems with Applications*, vol. 40, pp. 4915-4921, 2013.

[76] C. Chen, P. Hsin, and C. Wu, "Forecasting Taiwan's major stock indices by the Nash nonlinear grey Bernoulli model," *Expert Systems with Applications*, vol. 37, pp. 7557-7562, 2010.

[77] J. Wu and J. Liu, "A forecasting system for car fuel consumption using a radial basis function neural network," *Expert Systems with Applications*, vol. 39, no. 2, pp. 1883-1888, 2012.

[78] S. Julkami, S. P. Simon, and K. Sundareswaran, "A spiking neural network (SNN) forecast engine for short-term electrical load forecasting," *Applied Soft Computing*, vol. 13, no. 8, pp. 3628-3635, 2013.

[79] A. Lucifredi, C. Mazzieri, and M. Rossi, "Application of multiregressive linear models, dynamic Kriging models and neural network models to predictive maintenance of hydroelectric power system," *Mechanical Systems and Signal Processin*g, vol. 14, no. 3, pp. 471-494, 2000.

[80] A. Parlos, O. Rais, and A. Atiya, "Multi-step-ahead prediction using dynamic recurrent neural networks," *Neural Networks*, vol. 13, pp. 765-786, 2000.

[81] R. Isermann, "On fuzzy logic applications for automatic control, supervision, and fault diagnosis," *IEEE Transactions on Systems, Man, and Cybernetics – Part A: Systems and Human*, vol. 28, no. 2, pp. 221-235, 1998.

[82] T. Liu, J. Singonahalli, and N. Iyer, "Detection of roller bearing defects using expert system and fuzzy logic," *Mechanical Systems and Signal Processing*, vol. 10, no. 5, pp. 595-614, 1996.

[83] S. Li and M. Elbestawi, "Fuzzy clustering for automated tool condition monitoring in machining," *Mechanical Systems and Signal Processing*, vol. 10, no. 5, pp. 533-550, 1996.

[84] C. Mechefske, "Objective machinery fault diagnosis using fuzzy logic," *Mechanical Systems and Signal Processing*, vol. 12, no. 6, pp. 855-862, 1998.

[85] L. Yu, and Y. Zhang, "Evolutionary fuzzy neural networks for hybrid financial prediction," *IEEE Transactions on System, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 35, no. 2, pp. 244-249, 2005.

[86] **D. Li,** W. Wang, and F. Ismail, "Enhanced fuzzy-filtered neural network for material fatigue prognosis," *Applied Soft Computing*, vol. 13, no. 1, pp. 283-291, 2013.

[87] D. G. Stavrakoudis, and J. B. Theocharis, "Pipelined recurrent fuzzy neural networks for nonlinear adaptive speech prediction," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 37, no. 5, pp. 1305-1320, 2007.

[88] **D. Li**, W. Wang, and F. Ismail, "A fuzzy-filtered grey network technique for system state forecasting," *Soft Computing*, in press.

[89] Y. Lin, J. Chang, and C. Lin, "Identification and prediction of dynamic systems using an interactively recurrent self-evolving fuzzy neural network," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 24, no. 2, pp. 310-321, 2013.

[90] S. Yilmaz and Y. Oysal, "Fuzzy wavelet neural network models for prediction and identification of dynamical systems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 21, no. 10, pp. 1599-1609, 2010.

[91] **D. Li**, W. Wang, and F. Ismail, "Fuzzy neural network technique for system state forecasting," *IEEE Transactions on Cybernetics*, vol. 43, no. 5, pp. 1481-1494, 2013.

[92] Y. M. Chen, and C. T. Lin, "Dynamic parameter optimization of evolutionary computation for on-line prediction of time series with changing dynamics," *Applied Soft Computing*, vol. 7, no. 4, pp. 1170-1176, 2007.

[93] C. A. Bliss, M. R. Frank, C. M. Danforth and P. S. Dodds, "An evolutionary algorithm approach to link prediction in dynamic social networks," *Journal of Computational Science*, vol. 5, no. 5, pp. 750-764, 2014.

[94] F. K. Wang, K. K. Chang, and Y. Y. Hsiao, "Implementing a diffusion model optimized by a hybrid evolutionary algorithm to forecast notebook shipments," *Applied Soft Computing*, vol. 13, no. 2, pp. 1147-1151, 2013.

[95] P. C. Ko, and P. C. Lin, "An evolution-based approach with modularized evaluations to forecast financial distress," *Knowledge-Based Systems*, vol. 19, no. 1, pp. 84-91, 2006.

[96] Z. Tang and C. Almeida, P. A. Figurehwick, "Time series forecasting using neural networks vs. Box-Jenkins methodology," *Simulation*, vol. 57, no. 5, pp. 303-310, 1991.

[97] M. C. Medeiros, A. Veiga, "A hybrid linear-neural model for time series forecasting," *IEEE Transactions on Neural Networks*, vol. 11, no. 6, pp. 1402-1412, 2000.

[98] M. Khashei, M. Bijari, "A novel hybridization of artificial neural networks and ARIMA models for time series forecasting," *Applied Soft Computing*, vol. 11, no. 2, pp. 2664-2675, 2011.

[99] T. Taskaya-Temizel, K. Ahmad, "Are ARIMA neural network hybrids better than single models?," *Proceedings of International Joint Conference on Neural Networks*, pp. 3192-3197, 2005.

[100] I. Rojas, O. Valenzuela, F. Rojas, A. Guillen, L. J. Herrera, H. Pomares, L. Marquez, and M. Pasadas, "Soft-computing techniques and ARMA model for time series prediction," *Neurocomputing*, vol. 71, no. 4-6, pp. 519-537, 2008.

[101] C. Voyant, M. Muselli, C. Paoli, and M. L. Nivet, "Numerical weather prediction (NWP) and hybrid ARMA/ANN model to predict global radiation," *Energy*, vol. 39, no. 1, pp. 341-355, 2012.

[102] W. Ji, and K. C. Chee, "Prediction of hourly solar radiation using a novel hybrid model of ARMA and TDNN," *Solar Energy*, vol. 85, no. 5, pp. 808-817, 2011.

[103] N. Kasabov, "Evolving fuzzy neural networks for supervised/unsupervised online knowledge-based learning," *IEEE Transactions on Systems Man, and Cybernetics, Part B Cybernetics*, vol. 31, no. 6, pp. 902-918, 2001.

[104] N. Kasabov, Evolving Connectionist Systems, Springer, London, 2007.

[105] N. Kasabov, J. Kim, M. Wstts, A. Gray, "FuNN/2 - a fuzzy neural network architecture for adaptive learning and knowledge acquisition in multi-modular distributed environments," *Information Sciences*, vol. 101, no. 3-4, pp. 155-175, 1997.

[106] N. K. Kasabov, "DENFIS: dynamic evolving neural-fuzzy inference system and its application for time-series prediction," *IEEE Transactions on Fuzzy Systems*, vol. 10, no. 2, pp. 144-154, 2002.

[107] **D. Li**, W. Wang, and F. Ismail, "A spectrum synch technique for induction motor health condition monitoring," *IEEE Transactions on Energy Conversion*, conditionally accepted.

[108] W. H. Press, S. Teukolsky, W. T. Vetterling and B. P. Flannery, Numerical Recipes 3rd Edition: The Art of Scientific Computing, Cambridge : Cambridge University Press, 2007.

[109] J. Ilonen, J. K. Kamarainen, T. Lindh, J. Ahola, H. Kalviainen and J. Partanen, "Diagnosis tool for motor condition monitoring," *IEEE Transactions on Industry Applications*, vol. 41, no. 4, pp. 963-971, 2005.

[110] **D. Li**, W. Wang, and F. Ismail, "A selective boosting technique for pattern classification," *Neurocomputing*, accepted.

[111] R. Schapire, Y. Freund, P. Bartlett and W. Lee, "Boosting the margin: a new explanation for the effectiveness of voting methods," *The Annals of Statistics*, vol. 26, no. 5, pp. 1651-1686, 1998.

[112] J. Alcala-Fdez, L. Sanchez, S. Garcia, M. N. D. Jesus, S. Ventura, J. M. Garrell, J. Otero, C. Romero, J. Bacardit, V. M. Rivas, J. C. Fernandez, F. Herrera, "KEEL: A software tool to assess evolutionary algorithms to data mining problems," *Soft Computing*, vol. 13, no. 3, pp. 307-318, 2009.

[113] P. Melville, N. Shah, L. Mihalkova, R. J. Mooney, "Experiments on ensembles with missing noisy data," *Proceedings of 5th International Workshop on Multiple Classifier Systems*, pp. 293-302, 2004.

[114] R. Lowry, Concepts and Applications of Inferential Statistics, online: http://vassarstats.net/textbook/, retrieved 19 February 2014.

[115] J. Demsar, "Statistical comparisons of classifiers over multiple data sets," *The Journal of Machine Learning Research*, vol. 7, pp. 1-30, 2006.

[116] **D. Li**, W. Wang, and F. Ismail, "A new ensemble learning technique for system state forecasting," *IEEE Transactions on System Man and Cybernetics: Systems*, under review.

[117] H. Drucker, "Improving regressor using boosting techniques," *Proceedings of the 14th International Conferences on Machnie Learning*, pp. 107-115, 1997.

[118] D. P. Solomatine and D. L. Shrestha, "AdaBoost.RT: a boosting algorithm for regression problems," *Proceedings of the International Joint Conference on Neural Networks*, pp. 1163-1168, 2004.

[119] J. D. Farmer, "Chaotic attractors of an infinite-dimensional dynamical system," *Physica D: Nonlinear Phenomena*, vol. 4, pp. 366-393, 1982.

[120] **D. Li**, W. Wang, and F. Ismail, "An evolving fuzzy neural predictor for multi-dimensional system state forecasting," *Neurocomputing*, vol. 145, pp. 381-391, 2014.

[121] W. Wang, D. Li, J. Vrbanek, "An evolving neuro-fuzzy technique for system state forecasting," *Neurocomputing*, vol. 87, pp. 111-119, 2012.

[122] W. Antweiler, Database retrieval system (v2.15), University of British Columbia, February 2, 1996. [Online]. http://fx.sauder.ubc.ca/data.html. [Accessed October 2, 2012].

[123] S. Gunal, D. G. Ece and O. N. Gerek, "Induction machine condition monitoring using notch-filtered motor current," *Mechanical Systems and Signal Processin*g, vol. 23, pp. 2658-2670, 2009.

[124] M. S. Arulampalam, S. Maskell, N. Gordon and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Transactions on Signal Processing,* vol. 50, no. 2, pp. 174-188, 2002.

[125] A. Doucet, N. D. Freitas and N. Gordon, Sequential Monte Carlo Methods in Practice, New York: Springer, 2001.

[126] J. Spall, Bayesian Analysis of Time Series and Dynamic Models, New York: CRC Press, 1988.

[127] S. Sivaraman and M. M. Trivedi, "A general active-learning framework for on-road vehicle recognition and tracking," *IEEE Transactions on. Intelligent. Transportation*, vol. 11, no. 2, pp. 267-276, 2010.

[128] S. Sutharsan, T. Kirubarajan, T. Lang and M. McDonald, "An optimization-based parallel particle filter for multitarget tracking," *IEEE Transaction on Aerospace and Electronic System.*, vol. 48, no. 2, pp. 1601-1618, 2012.

[129] W. He, N. Williard, M. Osterman, M. Pecht, "Remaining useful performance analysis of batteries," *IEEE Conference on Prognostics and System Health Management (PHM)*, pp. 1-6, 2011.

[130] W. Xian, B. Long, M. Li, H. Wang, "Prognostics of lithium-ion batteries based on the Verhulst model, particle swarm optimization and particle filter," *IEEE Transactions on Instrumentation and Measurement*, no. 99, pp. 1, 2013.

[131] C. Musso, N. Oudjane, and F. LeGland, "Improving regularized particle filters," *Sequential Monte Carlo Methods in Practice*, A. Doucet, J. F. G. de Freitas, and N. J. Gordon, Eds. New York: Springer-Verlag, 2001.

[132] J. Liu, W. Wang and F. Ma, "A regularized auxiliary particle filtering approach for system state estimation and battery life prediction," *Smart Material and Structures*, vol. 20, no. 7, pp. 1-9, 2011.

[133] R. V. D. Merwe, A. Doucet, N. D. Freitas, E. Wan, "The unscented particle filter," Technical Report, Cambridge University Engineering Department, 2000.

[134] G. Castella and C. P. Robert, "Rao-Blackwellization of sampling schemes," *Biometrika*, vol. 83, pp. 81-94, 1996.

[135] **D. Li**, W. Wang, and F. Ismail, "A mutated particle filter technique for system state estimation and battery life prediction," *IEEE Transactions on Instrumentation and Measurement*, vol. 63, no. 8, pp. 2034-2043, 2014.

[136] B. W. Silverman, "Density estimation for statistics and data analysis," *Monographs on Statistics and Applied Probability*, Chapman & Hall, London.

[137] L. Devroye, "A Course on Density Estimation," *Progress on Probability and Statistics*, Birkhauser, Boston.

[138] K. Deep, M. Thakur, "A new crossover operator for real coded genetic algorithm," *Applied Mathematics and Computation*, vol. 188, pp. 895-911, 2007.

[139] R. Douc, O. Cappe and E. Moulines, "Comparison of resampling schemes for particle filtering," *Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis*, pp. 64-69, 2005.

[140] D. Simon, *Optimal State Estimation: Kalman, H_∞, and Nonlinaer Approaches*, New York: Wiley, 2006.

[141] B. Saha and K. Goebel, "Battery data set", NASA Ames Prognostics Data Repository, [http://ti.arc.nasa.gov/project/prognostic-data-repository], NASA Ames, Moffett Field, CA, 2007.

[142] NASA Prognostics Center of Excellence, Battery Prognostics, Laboratory Setup: http://ti.arc.nasa.gov/tech/dash/pcoe/battery-prognostics/lab-setup/.

[143] M. Clerc, *Particle Swarm Optimization (ISTE)*, Wiley-ISTE, 2006.

[144] A. Arefi and M. R. Haghifam, "A modified particle swarm optimization for correlated phenomena," *Applied Soft Computing*, vol. 11, no. 8, pp. 4640-4654, 2011.

[145] M. M. Noel, "A new gradient based Particle swarm optimization algorithm for accurate computation of global minimum," *Applied Soft Computing*, vol. 12, no. 1, pp. 353-359, 2012.

[146] J. Jung, J. Lee, and B. Kwon, "Online diagnosis of induction motors using MCSA," *IEEE Transactions on Industrial Electronics*, vol. 53, no. 6, pp.1842-1852, 2006.