Aircraft Jet Engine Health Monitoring Through System Identification Using Ensemble Neural Networks

Mahdiyeh Amozegar

A thesis
in
The Department
of
Electrical & Computer Engineering

Presented in Partial Fulfillment of the Requirements
For the Degree of Master of Applied Science
Concordia University
Montréal, Québec, Canada

June 2015

CONCORDIA UNIVERSITY
School of Graduate Studies

This is to certify that the thesis prepared

By: **Mahdiyeh Amozegar**

Entitled: **Aircraft Jet Engine Health Monitoring Through System Identification Using Ensemble Neural Networks**

and submitted in partial fulfillment of the requirements for the degree of

Master of Applied Science (Electrical and Computer Engineering)

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the final examining commitee:

________________ Chair
Dr. R. Raut

________________ External Examiner
Dr. A. Bagchi

________________ Internal Examiner
Dr. K. Skonieczny

________________ Supervisor
Dr. K. Khorasani

Approved ________________
Dr. W. E. Lynch, Chair
Department of Electrical and Computer Engineering

________ 2015 ________________
Dr. Amir. Asif
Dean, Faculty of Engineering and Computer Science

Abstract

Aircraft Jet Engine Health Monitoring Through System Identification Using Ensemble Neural Networks

Mahdiyeh Amozegar

In this thesis a new approach for jet engine Fault Detection and Isolation (FDI) is proposed using ensemble neural networks. Ensemble methods combine various model predictions to reduce the modeling error and increase the prediction accuracy. By combining individual models, more robust and accurate representations are almost always achievable without the need of ad-hoc fine tunings that are required for single model-based solutions.

For the purpose of jet engine health monitoring, the model of the jet engine dynamics is represented using three different stand-alone or individual neural network learning algorithms. Specifically, a dynamic multi-layer perceptron (MLP), a dynamic radial-basis function (RBF) neural network, and a dynamic support vector machine (SVM) are trained to individually model the jet engine dynamics. The accuracy of each stand-alone model in identification of the jet engine dynamics is evaluated. Next, three ensemble-based techniques are employed to represent jet engine dynamics. Namely, two heterogenous ensemble models (an ensemble model is heterogeneous when different learning algorithms (neural networks) are used for training its members) and a homogeneous ensemble model (all the models are generated using the same learning algorithm (neural network)). It is concluded that the ensemble models improve the modeling accuracy when compared to stand-alone solutions. The best selected stand-alone model (i.e the dynamic radial-basis function neural network in this application) and the best selected ensemble model (i.e. a heterogenous ensemble) in term of the jet engine modeling accuracy are selected for performing the FDI study.

Engine residual signals are generated using both single model-based and ensemble-based solutions under various engine health conditions. The obtained residuals are evaluated in order to detect engine faults. Our simulation results demonstrate that the fault detection task using residuals that are obtained from the ensemble model results in more accurate performance. The fault isolation task is performed by evaluating variations in residual signals (before and after a fault detection flag) using a neural network classifier. As in the fault detection results, it is observed that the ensemble-based fault isolation task results in a more promising performance.

Acknowledgments

I would like to express my deepest gratitude to my supervisor, Professor Khashayar Khorasani, whose guidance, patience and support from the initial to the final stage enabled me to write this thesis. Completion of my masters would not have been possible without his continuous help and support. I appreciate all his contributions of time and ideas.

Contents

List of Figures

List of Tables

NOMENCLATURE

ANN	Artificial Neural Network
AANN	Auto Associative Neural Network
BBN	Bayesian Belief Network
CCR	Correct Classification Rate
CPM	Change Point Methods
DNN	Dynamic Neural Network
DT	Decision Tree
EKF	Extended Kalman Filter
ESN	Echo State Network
FDI	Fault Detection and Isolation
FL	Fuzzy Logic
GA	Genetic Algorithms
GC	Gaussian Classifier
GMM	Gaussian Mixture Model
GP	Gaussian Process
GRNN	Generalized Regression Neural Network
GRNN	Generalized Regression Neural Network
HI	Human-machine Interaction
HMM	Hidden Markov Model
IIS	Improved Iterative Scaling
KF	Kalman Filter
KKT	KarushKuhnTucker
K-NN	Nearest Neighbor Classifier
LDA	Linear Discriminant Analysis
LoR	Logistic Regression
LR	Linear Regression
LRM	Linear Ridge Model

LRT	Likelihood Ratio Test
LVQ	Learning Vector Quantisation
MCS	Multiple Classifier Systems
MLP	Multi Layer Perceptron
N	Engine Rotational Speed
NFIS	Nonparametric Fuzzy Inference System
NNN	nested neural network
OQDF	Orthogonal Quadratic Discriminant Function
PC	Parzen Classifier
PCA	Principle Component Analysis
PLS	Partial Least Squares
P_C	Compressor Pressure
PNN	Probabilistic Neural Network
P_T	Turbine Pressure
RBN	Radial Basis Network
RF	Random Forest
RNN	Recurrent Neural Network
RRT	Robust Ratio Thresholding
RS	Rough Sets
SOM	Self Organizing Maps
SVM	Support Vector Machines
T_C	Compressor Temperature
TDL	Tapped Delay Line
T_T	Turbine Temperature
SVM	Support Vector Machines

Chapter 1

Introduction

Fault Detection and Isolation (FDI) has captured a wide range of attention in various industries including aerospace. FDI plays an important role in increasing safety and reducing operational costs of an aircraft. This is applicable to different subsystems of an aircraft, which also includes the engine. Early diagnosis of jet engine faults reduces both the operational and maintenance costs of an aircraft.

Various algorithms have been proposed for fault detection and diagnosis in various applications. At the high level these algorithms can be categorized into two major classes: Model Driven and Data Driven methods. Model driven algorithms require a realistic mathematical model of the system, which might be expensive to derive. Data-driven models, on the other hand, do not require a mathematical model and they can be trained using available engine data. The main drawback of the data-driven methods is their lack of confidence, which comes from the fact that their knowledge is distributed over a set of nodes (unlike the model-based approaches where the knowledge is centralized in the mathematical model). To respond to this issue, we propose a fault detection and isolation algorithm based on ensemble of data-driven models. The agreement among the ensemble members reduces the chance of error while increasing the overall confidence.

Ensemble-based decision making is employed in our real life: the essence of democracy where a group of people vote to make a decision for choosing an elected official or deciding about a new law, the judicial systems whether based on a jury of peers or a panel of judges, etc. There are also more tangible examples in daily life: consulting with different doctors before agreeing to a major medical operation, reading users' reviews before buying an item and lots of other examples. As a matter of fact, no matter if ensemble-based systems are going to be used for daily applications or machine learning applications, the original goal of using them is the same; *improving our conˇ-dence of making the right decision* by considering various opinions and then combining them in an effective manner to finalize our decision. In this chapter we aim to present the overview of using ensemble systems for the purpose of fault detection and isolation in jet engines.

The remainder of this chapter is organized as follows. The statement of problem is presented in Section 1.1. Section 1.2 presents a literature review on ensemble learning and soft-computing approaches applied to FDI problem in various applications. Section 1.3 explains the contributions of the thesis, followed by the thesis outline in Section 1.4. Finally, Section 1.5 summarizes the present chapter.

1.1 Thesis Objectives

In recent years, there has been numerous papers in the computer science community discussing how to combine models or model predictions, in order to reduce model error and increase the prediction accuracy. By combining models, more robust and accurate models are almost always achievable without the need of ad-hoc fine tunings required for single-model solutions.

The main objective of this thesis is to develop an FDI scheme for a single-spool jet engine by combining single-model solutions, and building an ensemble system. The goal is to improve the performance of single-model solutions (such as modeling jet engine dynamics using a single neural network) by combining multiple learners into an ensemble. In other words we would like to benefit from different models by combining them in order to have more accurate predictions. Combining stand-alone models increases the accuracy by reducing the bias and variance of the predictions. This is critical when we focus on residual generation problem. Having less biased residual with less variance helps to detect slight variations in jet engine's performance due to degradation or a failure which leads to more accurate FDI mechanism (as compared to single-model solutions).

1.2 Literature Review

1.2.1 Fault Detection and Isolation

The term fault is defined as any unexpected or unpredicted deviation or change from the desired system's behavior that can happen for either an unbounded or bounded period of time. It is more cost effective to predict the possible failure in the system due to a fault before it contributes to the system unsafe performance that in turn the system efficiency may decrease and even a drastic failure is caused. For this purpose, health monitoring is considered to be useful. Technically, health monitoring refers to the techniques and processes that enables one to monitor the system condition so the failure in the system can be predictable in advance. This section aims to briefly summarize Fault Detection and Isolation (FDI) methodologies in the literature based on [32, 33, 34]. FDI methodologies are classified into model-based, and data-riven (process history-based) approaches, each can be implemented either in qualitative or quantitative manner [32] (see Figure 1.1).

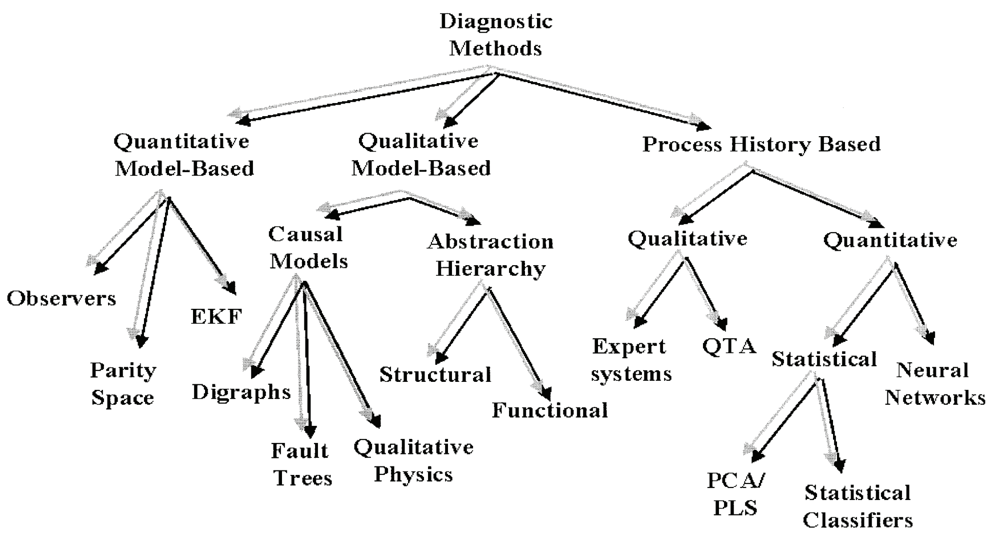


Figure 1.1: Classification of diagnostic methods [32].

Fig. 3. Classification of diagnostic algorithms.

Quantitative model-based approaches are based on explicit model of the plant under observation. Most of the works in this area focus on state-space estimation and input-output modelling of the system. Examples of quantitative model-based approaches are observer-based FDI and Extended Kalman Filters (EKF). Quantitative model-based approaches consist of two steps. The first step is to generate inconsistency between the actual output of the system and the expected behavior. This inconsistency is referred to as residual signal, which is the difference between actual output and expected behavior [32]. On the other hand, qualitative model-based approaches do not require mathematical modeling of the monitored system. Instead, they rely on qualitative modeling of the system's behavior. As an example, an FDI system with a large set of if-then-else rules (the rules are extracted from human experts) is categorized as qualitative model-based FDI system [33]. Examples of qualitative model-based methodologies are fuzzy inference systems, and fault trees [92], [93].

4

as first-principles models, frequency response models and so on. The first-principles models (also classified as macroscopic transport phenomena model (Himmelblau, 1978)) have not been very popular in fault diagnosis studies because of the computational complexity in utilizing these models in real-time fault diagnostic systems and the difficulty in developing these models. The most important class of models that have been heavily investigated in fault diagnosis studies are the input–output or state–space models and hence the focus is on these types of models.

5.1. Analytical redundancy

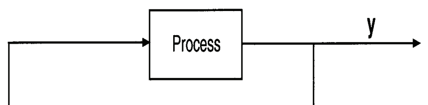
In the area of automatic control, change/fault detection problems are known as model-based FDI. Relying on an explicit model of the monitored plant, all model-based FDI methods (and many of the statistical diagnosis methods) require two steps. The first step generates inconsistencies between the actual and expected behavior. Such inconsistencies, also called *residuals*, are 'artificial signals' reflecting the potential faults of the system. The second step chooses a decision rule for diagnosis.

The check for inconsistency needs some form of redundancy. There are two types of redundancies, hardware redundancy and analytical redundancy. The former requires redundant sensors. It has been utilized in the control of such safety-critical systems as aircraft space vehicles and nuclear power plants. However, its applicability is limited due to the extra cost and analytical redundancy can be further classified into two categories (Basseville, 1988; Chow & Willsky, 1984; Frank, 1990), direct and temporal.

A direct redundancy is accomplished from algebraic relationships among different sensor measurements. Such relationships are useful in computing the value of a sensor measurement from measurements of other sensors. The computed value is then compared with the measured value from that sensor. A discrepancy indicates that a sensor fault may have occurred.

A temporal redundancy is obtained from differential or difference relationships among different sensor outputs and actuator inputs. With process input and output data, temporal redundancy is useful for sensor and actuator fault detection.

A general scheme of using analytical redundancy in diagnostic systems is given in Fig. 4. The essence of analytical redundancy in fault diagnosis is to check the actual system behavior against the system model for consistency. Any inconsistency expressed as residuals, can be used for detection and isolation purposes. The residuals should be close to zero when no fault occurs but show 'significant' values when the underlying system changes. The generation of the diagnostic residuals requires an explicit mathematical model of the system. Either a model derived analytically using first principles or a black-box model obtained empirically may be used.



Unlike the model-based approaches, the data-driven FDI does not require either a mathematical or a qualitative model of the monitored system. Data-driven FDI is based on available data acquired from the system. Similar to the model-based FDI, the data-driven approaches are classified into qualitative and quantitative. Qualitative data-driven FDI can be categorized into qualitative trend monitoring and expert systems. In these approaches, the obtained data is used to extract the set of rules governing the system [34]. Quantitative data-driven method is the category of FDI algorithms. In this approach data-driven approaches (e.g. artificial neural networks (ANN), support vector machines, etc.) are used to predict the expected behavior of the healthy system. The inconsistency between the actual behavior of the system and the prediction of FDI algorithm is used as an indication of fault occurrence. In another framework, data-driven FDI evaluates the measured inconsistency in order to isolate the detected fault [34]. Examples of quantitative FDI algorithms are artificial neural networks [143, 144, 145, 146] and support vector machines [146, 150].

In this thesis, the focus is on quantitative data-driven FDI algorithms. The contribution is to design an ensemble of quantitative data-driven methods for health monitoring of a jet engine. In the rest of this chapter, we review the ensemble learning methodologies with the application of health monitoring and time-series prediction. We also review the literature in order to find the most promising data-driven algorithms in order to be used in design of our ensemble system.

1.2.2 Jet Engine Fault Detection and Isolation

Jet engine condition monitoring is important both for reducing maintenance cost and increasing the flight safety. Therefore, it has been increasingly studied by the researches in recent years [94]. We discussed the model-based and data-driven fault diagnosis algorithms in the previous section, both approaches have been widely used in jet engine fault monitoring application. Kalman filter is a well-established quantitative model-based approach which has been extensively applied to jet engine fault diagnosis as in [95, 96, 97]. Fuzzy logic is a qualitative model-based approach extensively used for aircraft engine fault diagnosis [161, 171].

The mathematical complications required to derive the model of system especially in a case that the system nonlinear complexity is high is one of the most significant obstacles in using model-base approaches. On the other hand, since data-driven approaches rely on real-time or collected data from the sensors, there is no need of having the mathematical model of the jet engine. Data-driven approaches are widely used as an alternative for model-based approaches.

Several kinds of ANN are used in the application jet engine fault diagnosis. The use of dynamic neural networks for jet engine fault diagnosis is reported in [139, 140]. Feed-forward neural network is another widely used ANN for jet engine fault diagnosis [143, 144, 145, 146]. The use of RBF neural network for jet engine fault diagnosis is reported in [143, 179, 165, 174].

A major difficulty associated with data-driven approaches is the computational complexity for finding the appropriate learning method. Furthermore, there is always a chance that the selected learning algorithm does not satisfy the design requirements for some unseen samples of the input space. Ensemble learning has proven to improve individual learners generalization performance [184], [185], [187], and reduces the chance of selecting a learner with weak performance. An ensemble of classifiers is presented in [146] for fault isolation of jet engine using SVM, Decision Tree (DT), and MLP. The authors in [144], studied an ensemble classifier using MLP, Robust Ratio Thresholding (RRT) and Logistic Regression (LR) for fault diagnosis of jet engine. Random forest (RF) and Self Organizing Map (SOM) are used in [149] for fault detection of jet engine. Also, [179] presents an ensemble of neural networks using MLP and RBF for gas turbine fault isolation. According to our review, the use of ensemble learning for jet engine fault detection through system identification *has not been* reported. This research presents an ensemble of neural networks for fault detection and isolation of a single-spool jet engine through system identification.

1.2.3 Ensemble Learning

The very first use of ensemble learning goes back to 1979, where Dasarathy and Sheela partitioned the feature space using multiple classifiers [35]. Ensemble of similarly configured neural networks was first used in [187], the authors showed that an ensemble of neural networks can be used to improve the generalization performance. Later in 1990, [214] showed that a strong classifier with an arbitrary low error can be generated by combining a set of weak learners through an algorithm called boosting [118]. To extend the theory behind ensemble learning, [36] studied the bias-variance tradeoff, and shows that a single neural network is not able to learn complex problems standing alone.

Having established the theory behind ensemble learning, ensemble learning has captured lots of attention in computer science and engineering communities under various names [118] including: bagging [128], boosting [40, 135], mixture of experts [137], and neural networks ensemble [187], [186], [193]. The application of ensemble learning to time-series prediction and FDI problems is presented in the following section.

1.2.4 Ensemble Learning for Fault Detection and Isolation

The use of ensemble learning for FDI problem has been reported in several publications. This section reviews the literature This section focuses on ensemble techniques, where outputs of several predictors are aggregated together in order to form the final prediction. A variety of ensemble techniques have been applied to the FDI problem. Almost all authors demonstrate that the technique they propose outperforms some other methods chosen for the comparison; however, due to different data sets used by different authors and bearing in mind the fact that confidence intervals for the prediction accuracies are seldom provided, fair comparison of results obtained by different authors is hardly possible. Thus, in this section we focus on the architecture of studied ensemble system, rather than comparison of the obtained results. The details of this review is indexed in Table 1.1.

Loboda *et al.* showed that both MLP and RBF show acceptable performances for the application of gas turbine fault classification [143]. Xiao developed an ensemble classifier for fault diagnosis of aircraft engine using LR, MLP, LRT and RRT [144]. Zhang *et al.* developed an ensemble of feed-forward neural networks for fault diagnosis of chemical processes [145]. Yan *et al.* introduced an ensemble system for jet engine fault diagnosis by using SVM, MLP and DT [146]. Xiao *et al.* designed an ensemble system for fault diagnosis of gas turbine with GRNN, LoR and RF [148]. Varma *et al.* uses RS and SOM for anomaly detection problem of gas turbine [149] . Donat presented five fault classifiers based on K-NN, SVM, GMM, PNN and PCA [150]. Volponi presented an ensemble system based on MLP and RBF neural networks to improve diagnostic accuracy and reduce the rate of misdiagnosis for the aircraft engine gas path faults [179]. Kestner *et al.* introduced an offline fault diagnostics method for highly degraded industrial gas turbines based on Bayesian networks [180]. Huang *et al.* proposed a multiple classifiers fusion using within-class decision support for fault diagnosis where the base classifiers selected are K-NN, OQDF [152]. Amanda *et al.* as well as A. J. C. Sharkey *et al.* used ensemble of MLP networks for fault diagnosis of diesel engine [154], [155]. Lei *et al.* presented an MCS for fault detection problem of a gearbox by combining MLP, RBF and KNN [157]. Yan proposed a MCS with SVM, LDA, KNN , IIS, LVQ, GMM for fault diagnosis of an induction motor [158]. Oukhellou *et al.* designed an ensemble of MLP neural networks for fault diagnosis of railway track circuits [159]. Chen proposed an ensemble of RBF neural networks for fault diagnosis of power transformers [160]. Bonissone *et al.* presented an MCS for aircraft prognostic and health monitoring (PHM) with fuzzy classifiers, MLP, SOM, SVM and RF as the base classifiers [161]. Dong *et al.* introduced an expert system design method based on the neural network ensembles for missile fault diagnosis [164]. Nikunj *et al.* presented an ensemble of MLP and RBF for the aircraft health monitoring [165]. Filippi *et al.* designed an ensemble of MLPs fault-tolerance pattern recognition [167]. Ren *et al.* combined three classifiers MLP, FL and HI to solve fault diagnosis problem of an aero-engine [171]. Murphey *et al.* selected a two-step neural network ensemble approach by using MLP that is particularly suitable for solving vehicle diagnostics problems [172]. Chandroth *et al.* studied MCS with MLP, RBF, PCA and Wavelets for fault diagnosis of a diesel engine [174]. Aiming at more efficient fault diagnosis in mechanica systems, Georgoulas *et al.* presented an MCS with PCA, KNN and Gaussian Classifier as base classifiers [176]. An ensemble learning algorithm is proposed by Xu *et al.* based on individual MLP neural networks that are actively guided to learn diversity in power transformers [177]. Ren *et al.* studied a method of analog circuit fault diagnosis using AdaBoost with SVM-based base classifiers and Tent map is used to adjust parameters of SVM component classifiers for maintaining the diversity of weak classifiers [178]. More detailed review is shown in Table 1.1.

Table 1.1: A survey of hybrid and ensemble-based soft computing techniques applied to FDI.

Techniques	Description	Application
MLP+RBF	[143] runs comparative study between MLP and RBF for the application of gas turbine fault classification. The comparison results confirm that both MLP and RBF show acceptable performances while RBF is a little more accurate than MLP. However, RBF needs more available computer memory and computation time.	Gas turbine
MLP + LRT + RRT+LR	[144] developed an ensemble classifier for fault diagnosis of aircraft engine based. The ensemble classifier composed of 18 different classifiers including LR, MLP, LRT and RRT as inference engines. It shows that the fused result improves as compared with each classifier standing alone.	Gas turbine (jet engine)

Continued on next page

Table 1.1 – continued from previous page

Techniques	Description	Application
MLP	[145] develops an ensemble of feed-forward neural networks for fault diagnosis of chemical processes. To develop a diverse range of individual networks, each individual network is trained on a replication of the original training data generated through bootstrap re-sampling with replacement. The final decision is made based on majority voting.	Chemical Process
SVM+MLP+ DT	A multiple classifier system is developed for jet engine fault diagnosis with SVM, MLP and DT as its inference engine. The final decision is determined with three different approaches 1- averaging 2- dynamic selection and 3- dynamic fusion [146]	Gas turbine (jet engine)
GRNN + LoR + RF	[148] deigns a MCS for fault diagnosis of gas turbine with MLP, LoR and RF as its inference engines. It shows that by using a set of diverse classifiers, there is a potential to realize gains in classification accuracy over any individual classifier.	Gas turbine

Continued on next page

Table 1.1 – continued from previous page

Techniques	**Description**	**Application**
RF + SOM	[149] uses RS and SOM for anomaly detection problem of gas turbine. The paper discusses about diversity of classifiers but it doesn't present any metrics for measuring diversity of classifiers. It justifies that using different types of classifiers has a potential of having diverse classifiers.	Gas turbine
SVM + GMM + PCA	[150] initially presents five fault classifiers based on K-NN, SVM, GMM, PNN and PCA. It evaluates the performance of each classifier. Next it selects the three least accurate classifiers for fusion, reasoning that the least accurate classifiers should be diverse from each other.	Gas turbine

Continued on next page

Table 1.1 – continued from previous page

Techniques	Description	Application
MLP + RBF	To improve diagnostic accuracy and reduce the rate of misdiagnosis of the aircraft engine gas path faults, [179] presents an ensemble system based on MLP and RBF neural networks. The fusion algorithm employed in this research is Dempster-Shafer evidence theory and least square support vector machines (LSSVM).	Gas turbine
Bayesian	[180] presents an offline fault diagnostics method for highly degraded industrial gas turbines based on Bayesian networks where the health condition of each component is quantified in comparison to an expected value. The presented method uses multiple Bayesian network models each of which contains a subset of the unknowns. Their results are averaged according to how much each of the models is supported by the data.	Gas turbine

Continued on next page

Table 1.1 – continued from previous page

Techniques	Description	Application
KNN + OQDF + PC	[152] paper proposes a multiple classifiers fusion using within-class decision support for fault diagnosis where the base classifiers selected are K-NN, OQDF [153], and PC.	–
MLP	[154], [155] use ensemble of MLP networks for fault diagnosis of diesel engine. They discuss different architectures for a MCS. Specifically, they distinguish between modular (where a winner classifier which is selected dynamically takes the final action) and ensemble (where there is fusion algorithm for aggregating between the networks) MCS.	Diesel engine
MLP + RBF + KNN	[157] presents a MCS for fault detection problem of a gearbox. MLP, RBF and KNN are three classifiers which are combined using GA in order to make the final decision.	Gear box

Continued on next page

Table 1.1 – continued from previous page

Techniques	Description	Application
SVM + LDA + KNN + IIS + LVQ + GMM	[158] presents a MCS for fault diagnosis of an induction motor. The proposed algorithm trains six base classifiers based on a set of preprocessed data (the data is obtained after sensor data fusion). Three different fusion methods are applied 1- Bayesian belief 2- majority voting and 3- dynamic selection based on diversity. The last method outperforms the other two methods.	Induction motor
MLP	[159] designs an ensemble of MLP neural networks for fault diagnosis of railway track circuits. To generate the required diversity the networks are trained using different training sets. Fusion of classifiers is based on Dempster-Shafer classifier fusion method.	Railway track circuits

Continued on next page

Table 1.1 – continued from previous page

Techniques	Description	Application
RBF	[160] proposes an ensemble of RBF neural networks for fault diagnosis of power transformers. The authors train a number of RBF networks by selecting random training sets without replacement. Classifiers with the best performance are selected for fusion. The fusion algorithm is majority voting.	Power transformers
FL + MLP + SOM + SVM + RF	[161] presents a MCS for aircraft prognostic and health monitoring (PHM) with fuzzy classifiers, MLP, SOM, SVM and RF as the base classifiers. It measures dissimilarity between classifiers based on a few metrics before fusing their output.	Aircraft PHM
MLP	In [164] an expert system design method based on the neural network ensembles is proposed. The expert system design method is then applied to missile fault diagnosis.	Missile

Continued on next page

Table 1.1 – continued from previous page

Techniques	Description	Application
MLP + RBF	In [165] presents an ensemble where the mismatch between actual flight maneuver being performed and the maneuver predicted by the ensemble of MLP and RBF networks is a strong indicator that a fault is present. The authors do not study fault diagnosis problem in this paper.	Aircraft health monitoring
MLP	[166] gives a description of a two-stage classifier system for fault diagnosis of industrial processes. The first-stage classifier is used for fault detection and the second one is used for fault isolation and identification. The first stage generates the residual signals (acts as a reference model) while the second stage works as a classifier.	Aircraft health monitoring

Continued on next page

Table 1.1 – continued from previous page

Techniques	**Description**	**Application**
MLP	[167] designs an ensemble of MLPs to overcome a major concern in the use of neural networks which is the difficulty to define the proper network for a specific application, due to the sensitivity to the initial conditions and to overfitting and underfitting problems which limit their generalization capability.	fault-tolerance pattern recognition
CPM	In [170] ensemble of change point methods is used to present a fault prognosis algorithm. change-point methods include methods like Generalized Linear Models, logistic regression (methods based on maximum likelihood estimation). The fusion method used in the paper is weighted averaging and it is initiated from continuous behavior of the models.	–

Continued on next page

Table 1.1 – continued from previous page

Techniques	Description	Application
MLP + FL + HI	[171] combines three classifiers to solve fault diagnosis problem of an aero-engine: 1- multi-layer perceptron 2- fuzzy logic expert system 3- a rule based classifier based on human experts opinions. The authors deduce that the classifier should be diverse since they differ for their reasoning mechanism as artificial neural network, fuzzy set and human experts reasoning, respectively.	aero-engine
MLP	[172] presents a two-step neural network ensemble approach that is particularly suitable for solving vehicle diagnostics problems. First, the authors train a large pool of neural networks and select a diverse neural network ensemble based on large amounts of data acquired from a few available vehicles. Next, they train an ensemble decision function on a small amount of data that is acquired from the vehicle model to which the ensemble is applied.	Vehicles

Continued on next page

Table 1.1 – continued from previous page

Techniques	Description	Application
MLP + RBF + PCA +Wavelets	[174] studies MCS for fault diagnosis of a diesel engine. The author answers two main questions 1- Can the diagnostic performance be improved by combining the decisions of several individual classifiers 2- Is there a relationship between the robustness of the combined system and the methodological diversity used to create them? It also ranked the processes used to create diversity (classifier types, training set variation, training set composition and classifier structures).,	Diesel engine
GC + KNN + PCA	Aiming at more efficient fault diagnosis, [176] presents an MCS with PCA, KNN and Gaussian Classifier as base classifiers. Vibration signals from normal bearings and bearings with three different fault locations in a mechanical system is used as data set.	Mechanical systems

Continued on next page

Table 1.1 – continued from previous page

Techniques	Description	Application
MLP	In [174] The development of a neural net system for fault diagnosis in a marine diesel engine is described. Three different types of data were used: pressure, temperature and combined pressure and temperature. Subsequent to training, three nets were selected and combined by means of a majority voter to form a system which achieved 100% generalization to the test set. Following experimental evaluation of methods of creating diverse neural nets solutions, the authors conclude that the best results should be obtained when data is taken from two different sensors (e.g. pressure and sensor).	Marine diesel engine

Continued on next page

Table 1.1 – continued from previous page

Techniques	Description	Application
MLP	In [177] an ensemble learning algorithm is proposed based on individual MLP neural networks that are actively guided to learn diversity. By decomposing the ensemble error function, error correlation terms were included in the learning criterion function of individual networks. And all the individual networks in the ensemble were leaded to learn diversity through cooperative training. The method was applied in dissolved gas analysis based fault diagnosis of power transformer.	Power transformer
SVM	In [178] presents a method of analog circuit fault diagnosis using AdaBoost with SVM-based base classifiers. Each SVM classifier is equipped with a radial basis function kernel. Tent map is used to adjust parameters of SVM component classifiers for maintaining the diversity of weak classifiers.	Analog circuits

Ensemble Learning for FDI: A Concluding Remark

According to the literature, the use of ensemble learning for systems' health monitoring have shown an extensive interest. Numerous publications have addressed the benefits of ensemble learning towards FDI problem; however, no research has been reported on the use of *ensemble learning for health monitoring through system identiˇcation*. This thesis proposes a novel approach for the fault detection and isolation through system identification using ensemble methods.

1.3 Thesis Contributions

To the best of our knowledge ensemble learning has not been used for health monitoring of the jet engine *through system identiˇcation*. Several researches have employed ensemble learning to *evaluate* residual signals to detect or isolate a fault but no research has addressed the possible use of ensemble learning for *generating* the residuals. In other words, several researches have developed *ensemble of classiˇers* which receives residual signals; however, no research has addressed the use of *ensemble of regressors* to identify the jet engine dynamics and generating residual signals. The major contributions of this thesis can therefore be summarized as follows:

- A novel approach is proposed for identification of jet engine dynamics based on ensemble methods. According to the literature, this research reports the first use of ensemble learning for dynamic systems identification.
- A fault detection scheme is proposed based on system identification of jet engine using ensemble methods. Various ensemble architectures have been studied to determine the ensemble method with maximal improvement as compared with single model-based solutions (e.g. a solution based on only one type of neural network).

- A comparative study shows that the proposed ensemble-based system identification can reduce jet engine modeling error as compared to single-model-based solutions, and thus more accurate residual signals could be generated. The obtained residuals are then evaluated toward fault detection and isolation problem of the jet engine, and it is observed that the ensemble-based jet engine fault detection is more accurate (as compared to single-model-based solutions). An improvement in correct classification rate of engine faults is also observed in the fault isolation stage.

- From the standpoint of computational requirement, the use of ensemble methods may appear more costly (training multiple models instead of one); however, this could be compensated as ensemble methods remove the need of ad-hoc fine tunings required for single model-based solutions, as the key for having a more accurate ensemble is to increase the number of ensemble members. In theory, the accuracy of an ensemble model can be improved *arbitrarily* by increasing the number of ensemble members *without the need of having very accurate individual ensemble members*.

1.4 Thesis Outline

The organization of this thesis is as follows. Chapter 2 presents the necessary background information about jet engine dynamics, ensemble learning, and data driven algorithms used in this research. Moreover, the engine mathematical representation of a single spool jet engine is presented. Chapter 3, first presents a scheme for identifying the jet engine dynamics and generating residual signals using neural networks ensemble. The residual signals are later used in the chapter to accomplish fault detection task. Chapter 4 presents fault isolation scheme and the corresponding simulation results using ensemble of classifiers. The thesis conclusions and future work are given in Chapter 5.

1.5 Summary

This chapter provided an introduction to the jet engine fault detection and isolation problem. It also reviewed different FDI techniques used in the literature including data-driven approaches. A comprehensive review on the application of ensemble learning for FDI problem is presented. The review shows an extensive interest in ensemble systems for solving FDI problem; however, the use of ensemble system for health monitoring (i.e. FDI problem) through system identification has never been reported. Thus, the rest of this thesis focuses on the use of ensemble learning for fault detection and isolation of jet engine through system identification.

Chapter 2

Background Information

This chapter contains three parts. The first part presents a review on ensemble learning, bias-variance decomposition, importance of diversity in ensemble learning and the metrics for measuring diversity in ensemble systems. The second part presents the preliminaries about soft-computing methods used in this research as the ensemble system members. Finally, the third part is a review on jet engine dynamics and its possible failures.

2.1 Ensemble Learning

Ensemble systems were originally designed to reduce the variance and consequently increasing the accuracy. They captured an increasing attention among the machine learning community. This section provides an overview about ensemble systems, their properties, and their design procedures.

2.1.1 Justification for Ensemble Learning

Any classification error has two components with a trade-off relationship: *bias*, the accuracy of the classifier; and *variance*, the precision of the classifier when trained on different training sets. The fact is that usually the classifiers with low bias tend to have low variance and vice versa. On the other hand, averaging has a smoothing affect that can be employed for variance reduction. Therefore, in ensemble systems we first choose different classifiers with the fixed or similar bias and then combine them all together with averaging to reduce the variance.

The reduction of variability can be considered as reducing high frequency (high variance) noise employing a moving average filter. This filter works in a way that each sample of the signal is averaged by a neighbor of samples around it. If we suppose that the noise in each sample is independent, the noise component is averaged out while the information component stays unaffected. Indeed, the information component is common to all segments of the signal and after averaging operation, it is still unchanged. The same analysis is valid about increasing classifier accuracy by using an ensemble of classifiers. It can be assumed that classifiers make different errors on each sample while they are agreed with each other on their correct answers in terms of correct classifications. Hence, the error caused by misclassification decreases by averaging the classifier output and in turn, averaging out the error component.

Here, it is worthwhile to consider two issues. First, averaging is just one of the many ways of combining the classifier members in ensemble classifier. Secondly, making ensemble system from the ensemble members is not necessarily a guaranteed way to choose a performance that is better than that of the best ensemble member. Rather, it reduces the chance of selecting a classifier with a weak performance. Hence, if there is a member classifier with the better performance than one of ensemble classifiers, it is chosen and there is no need of using the ensemble classifier.

All the ensemble-based systems differ from each other in regard with the selection of the training data for the individual ensemble members, the criteria used for selecting the ensemble members of the ensemble system and/or the combination rules for joining the ensemble members together to make the finale ensemble system. Below, we will give more explanations about the presented discussions above.

2.1.2 Bias Variance Trade-off

A popular measure to evaluate the performance of a learner is *Mean Square Error* (MSE) [117]. The learning error in term of MSE might be used as a criterion for selecting a learning method. It can be shown that the learning error can be decomposed into two different components: 1- bias and 2- variance as follows [119], [117]:

$$\text{Learning error} = (\text{bias})^2 \quad \text{variance}$$

The optimal learner is the one which minimizes the learning error. Consequently, the optimal leaner is the one which has the 1- minimum bias and 2- minimum variance as compared with other learning algorithms.

In General *bias* is large if the learning method produces classifiers that are consistently wrong. Bias is small if:

- The classifiers are consistently right, or
- different training sets cause errors on different documents, or
- different training sets cause positive and negative errors on the same documents, but that average out to close to 0.

Variance is the variation of the prediction of learned classifier. Variance is large if choosing different training sets results in very different classifiers. It is small if:

- The training set has a minor effect on the classification decisions, whether they are correct or not.

In other words, variance measures how inconsistent the decisions are, not whether they are correct or incorrect.

In general, when comparing two different learners, in most cases the comparison shows that one method having higher bias and lower variance and the other lower bias and higher variance [117]. The decision for one learning method vs. another is then not simply a matter of selecting the one that has *small variance* or the one that has *small bias*. Instead, we have to weigh the respective merits of bias and variance in our application and choose accordingly. This tradeoff is called the *bias-variance tradeoff*.

Originally, ensemble learning methods were developed to improve accuracy by reducing the variance in learner outputs, while maintaining the bias of the learner low [118]. We further discuss how ensemble learning helps to reduce the variance. For this purpose we present mathematical explanation of bias-variance trade-off here. The more detailed explanations can be found in [113].

- **Bias-variance tradeoff for regression problem:** Suppose we want to learn a function f from R^N to R. We have n samples of the function f ,(x_i, y_i) where $i = 1, ..., n$ and $y_i = f(x_i)$. The ensemble consists of N members and the output of member α is denoted by $V^\alpha(x)$. The output of the ensemble which is a weighted average of each networks' output is denoted by:

$$\overline{V(x)} = \sum_\alpha \omega_\alpha V^\alpha(x)$$

We interpret the wight ω_α as our belief to the member α. Thus, we expect:

$$\sum_\alpha \omega_\alpha = 1,\ \omega_\alpha \quad 0$$

The *ambiguity* term on the ensemble x is defined as:

$$a^\alpha(x) = (V^\alpha(x) - \overline{V}(x))^2 \tag{2.1}$$

The *ensemble ambiguity* on input x is defined as follows and is simply the variance of weighted ensemble around the weighted mean. It measures the disagreement between different ensemble members on the input x, that is

$$\overline{a}(x) = \sum_\alpha \omega_\alpha a^\alpha(x) = \sum_\alpha \omega_\alpha \overline{a}(x) = (V^\alpha(x) - \overline{V}(x))^2 \tag{2.2}$$

The quadratic error of ensemble member α and the ensemble are defined as:

$$\begin{aligned} \epsilon^\alpha(x) &= (f(x) - V^\alpha(x))^2 \\ e(x) &= (f(x) - \overline{V}(x))^2 \end{aligned}$$

Now, if we add and subtract $f(x)$ to and from equation (2.2) we will end up with the next equation:

$$\overline{a}(x) = \sum_{\alpha} \omega_\alpha \epsilon^\alpha(x) - e(x) \tag{2.3}$$

Finally, we have:

$$e(x) = \overline{\epsilon}(x) - \overline{a}(x) \tag{2.4}$$

where $\overline{\epsilon}(x)$ is the weighted average of individual errors. We can average all these expressions over the distribution $p(x)$ of the input x. The generalization error and ambiguity term for ensemble member α is defined in the first two equations and the last equation is the generalization error of the ensemble, that is

$$E^\alpha = \int p(x)\epsilon^\alpha(x)dx \tag{2.5}$$

$$A^\alpha = \int p(x)a^\alpha(x)dx \tag{2.6}$$

$$E = \int p(x)e(x)dx \tag{2.7}$$

Considering the above equations and (2.4) we have:

$$E = \overline{E} - \overline{A} \tag{2.8}$$

where $\overline{E} = \sum_\alpha \omega_\alpha E^\alpha$ is the weighted average of individual ensemble members' error and $\overline{A} = \sum_\alpha \omega_\alpha A^\alpha$ is the *ensemble ambiguity*.

Equation (2.8) separates the generalization error of the ensemble into two terms. The first term is the weighted average of individual ensemble members' errors and the second term is the ambiguity term. Please note that the ambiguity term can be determined without any prior knowledge about the value of target function $f(x)$. We can evaluate the ambiguity term just based on the ensemble members' output.

Equation (2.8) states if the ensemble is strongly biased (E is a large number) then the ambiguity term is small. In this case the ensemble members implement very similar functions which agree together even outside of the training set. On the other hand, if the ambiguity term is high then the generalization error of the ensemble is less than the weighted average generalization error of each member. From equation (2.8) we can see that $E \quad \overline{E}$. In case of uniform weights for ensemble member contributions we always have:

$$E \quad \frac{1}{N}\sum_{\alpha} E^{\alpha}$$

This has been proved by several authors for instance in [113], [192]

- **Bias-variance tradeoff for classification problem:** The same concept can be developed for ensemble of classifiers. A formal mathematical proof for bias-variance tradeoff in classification problem can be found in [198].

2.1.3 Diversity in Ensemble Learning

1. **Regression Problem:** For regression ensembles, Krogh and Vedelsby [113] proved that the quadratic error of the ensemble estimator is guaranteed to be less than or equal to the average quadratic error of the components, that is

$$\mathrm{E}_{\text{ensemble}} = \mathrm{E}_{\text{individual models}} - A$$

where E is the prediction error with ... denoting averaging over all models and A represents the ensemble ambiguity which measures the difference in prediction of individual models from the overall ensemble [114]. Intuitively this means that larger the ambiguity term, the larger is the ensemble error reduction. However, if all the models have low prediction error it is likely that they are very similar to one another. If they are very different from one another, all of them may not have low prediction error.

This implies that the right balance is required between the *diversity* (ambiguity term) and the *individual accuracy* (the average error term) in order to achieve low ensemble error. Extensions of the model proposed by Krogh et. *al* have been studied by Brown et. *al* [115] who show that Negative Correlation (NC) plays an important role in the diversity of ensembles [195].

To *link* this section to *bias-variance trade-off*, the reader should note that the diversity of classifiers comes down to their variance which must be different from each other while each of them must maintain an acceptable level of individual accuracy (bias), since ensemble learning is a promising approach to reduce the variance of classifiers.

Since diversity is a *must* for designing an ensemble, we need to have some metrics to measure it. For this purpose, [116] introduced different metrics for measuring diversity of ensembles in *regression problem* and are presented later in this chapter.

2. **Classification problem**: An ensemble of classifiers is a set of classifiers whose individual decisions are combined in some way (typically by weighted or unweighted voting) to classify

new examples. One of the most active areas of research in supervised learning has been to study methods for constructing good ensembles of classifiers. The main discovery is that ensembles are often much more accurate than the individual classifiers that make them up.

A necessary and sufficient condition for an ensemble of classifiers to be more accurate than any of its individual members is if the classifiers are *accurate* and *diverse* [187]. An accurate classifier is one that has an error rate of better than random guessing on new x values. This concept has been illustrated in Figure 2.1 [118].

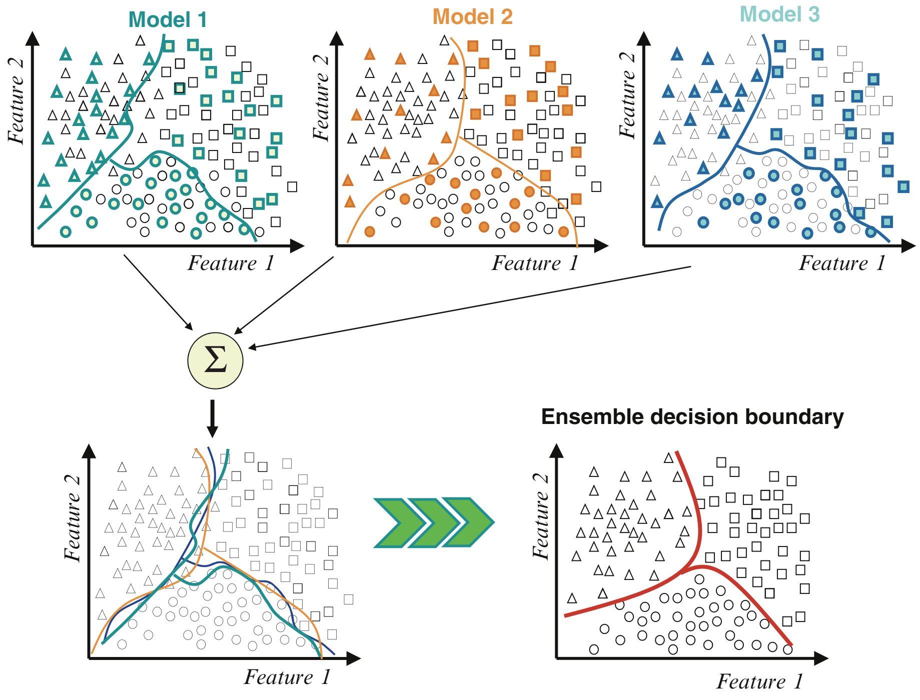


Figure 2.1: Variability reduction using MCS [118].

Two classifiers are diverse if they make different errors on new data points. To see why accuracy and diversity are good, imagine that we have an ensemble of three classifiers:

h_1, h_2, h_3 and consider a new case x. If the three classifiers are identical (i.e., not diverse), then when $h_1(x)$ is wrong, $h_2(x)$ and $h_3(x)$ will also be wrong. However, if the errors made by the classifiers are uncorrelated, then when $h_1(x)$ is wrong, $h_2(x)$ and $h_3(x)$ may be correct, so that a majority vote will correctly classify x. In other words by diversity we mean that the classifiers are independent in terms of error. More precisely, if the error rates of each classifier is equal to p 1 2 and if the errors are independent, then the probability that the majority vote will be wrong will be the area under the binomial distribution where more than half of the ensemble members are wrong.

Measuring Diversity in Regression Ensembles

As we previously discussed, diversity is a *must* for designing an ensemble system. This section presents metrics for measuring diversity among individual regressor (resp. classifiers) of an ensemble system based on [124], [125], [126], [127], [213].

In this section we present the metrics for assessing the diversity between regressors. Assume we have two regressors R^m and R^n. $Y^m = [y_1^m, ..., y_N^m]$ and $Y^n = [y_1^n, ..., y_N^n]$ are the continuous valued outputs of the regressors which are N-dimensional.

i) **Correlation coefficient:** The correlation between Y^m and Y^N is defined as follows. The correlation is inversely proportional with diversity, which means that two regressors with low correlation are preferred. μ_{Y^m} (resp. μ_{Y^n}) is the mean of Y^m (resp. Y^n).

$$\rho = \frac{\sum_{i=1}^{N}(y_i^m - \mu_{Y^m})(y_i^n - \mu_{Y^n})}{\sum_{i=1}^{N}(y_i^m - \mu_{Y^m})^2 \sum_{i=1}^{N}(y_i^n - \mu_{Y^n})^2} \tag{2.9}$$

ii) **Covariance:** covariance between Y^m and Y^n is defined as follows and it is very related to correlation coefficient. Similarly, covariance is inversely related to diversity.

$$Cov(Y^m, Y^n) = E[(Y^m - \mu_{Y^n})(Y^n - \mu_{Y^m})] \tag{2.10}$$

iii) **Chi-square:** Chi-square of Y^m with respect to Y^n is defined with the following equation. It is directly related with diversity between Y^m and Y^n, that is

$$\chi^2 = \frac{\sum_{i=1}^{N}(y_i^m - y_i^n)^2}{y_i^n} \tag{2.11}$$

iv) **Mutual information:** The mutual information between Y^m and Y^n are given by:

$$I(Y^m, Y^n) = H(Y^m) \quad H(H^n) - H(Y^m, Y^n)$$

where $H(Y^m)$ and $H(H^n)$ are the differential entropies of Y^m and Y^n and $H(Y^m, Y^n)$ is the join differential entropy between Y^m and Y^n [213]. It is inversely related with diversity of the regressors.

Measuring Diversity in Classification Ensembles

The first three criteria for member classifier selection are more traditional in the literature. The other two presented in [212] are based on the assumption of the significance of the classification errors being made. All of these approaches work in a pairwise fashion.

i) **Correlation Between Errors**: Intuitively, the independence of occurring errors should be important for member classifier selection. In turn, the correlation of errors is a natural choice for comparing the subsets of classifiers. The correlation $\rho_{a,b}$ is determined as follows:

$$\rho_{a,b} = \frac{Cov(v_e^a, v_e^b)}{\overline{Var(v_e^a)Var(v_e^b)}} \tag{2.12}$$

where v_e^a and v_e^b are the binary vectors of error occurrence in classifiers a and b, respectively, Cov refers to covariance and Var refers to variance. The best set is the one with minimal mean pairwise correlation.

ii) **Q Statistics:** Q static is useful to assess the similarity of two classifiers. For two classifiers a and b, it is defined as:

$$Q_{a,b} = \frac{N^{00}N^{11} - N^{01}N^{10}}{N^{00}N^{11} \quad N^{01}N^{10}} \tag{2.13}$$

where N^{11} is the number of times both classifiers are correct, N^{00} both classifiers are incorrect, and N^{10} and N^{01} are the number of times when just the first classifier or the second one is correct, respectively. When N^{11} andN^{00} are both equal to 1, the value of $Q_{a,b}$ is one. In other words, when two classifiers make the same correct and incorrect decisions, $Q_{a,b}$ becomes one. On the other hand, when the classifiers make errors on different inputs, negative $Q_{a,b}$ is obtained. In the case of dealing with a set contained more than two classifiers, the Q statistic of the whole set is the mean value of pairwise Q statistics. Therefore, it is expected that the best set of member classifiers is the one with minimum value of Q.

iii) **Mutual Information:** Calculating the mutual information of the classifiers is also beneficial for selecting a good set of member classifiers. By definition, mutual information measures the amount of information shared between classifiers. Therefore, it is logical that a set of classifiers is maximally diverse if the mutual information assigned to that set has the minimum value. The mutual information between two classifiers a and b is defined as follows:

$$I_{a,b} = \sum_{i=1}^{n}\sum_{j=1}^{n} p(c_i, c_j) log(\frac{p(c_i, c_j)}{p_a(c_i)p_b(c_j)}) \tag{2.14}$$

where n is the number of total classes and $c_i, i = 1, ..., n$ are the class labels. The mutual information of the error occurrence is also calculated in experiments and in that, just two classes, correct or incorrect, are used for each classifier. As in previous criterion, for calculating the mutual information of the larger set of classifiers, first the pairwise mutual information is calculated. The minimum mutual information offers the optimal subset of classifiers.

iv) **Ratio Between Different and Same Errors:** Calculating ratio between different and the same errors is another attempt to select the best set of member classifiers and it is defined as follows:

$$r_{a,b}^{DES} = \frac{N_{different}^{00}}{N_{same}^{00}} \tag{2.15}$$

where $N_{different}^{00}$ is the number of the times that the two classifiers made different errors at the same input and N_{same}^{00} is the number of the times that they made the same error. For more than two members, the mean of the pairwise ratios is calculated. The optimum subset is the one with the maximum discussed ratio.

v) **Weighted Count of Errors and Correct Results:** Information on correct decision should be taken into consideration with more emphasis on a situation which the classifiers agree on either the correct and incorrect results. For this purpose, the occurrences of the situation should be counted and then a suitable emphasis should be placed on a positive situation that is assigned to ”both correct” and a negative situation where ”both incorrect” is met:

$$r_{a,b}^{WCEC} = N^{11} \quad \frac{1}{2}(N^{10} \quad N01) - N_{different}^{00} - 5N_{same}^{00} \tag{2.16}$$

For multiple classifiers, the mean of the pairwise counts is used. The optimal subset of classifiers is selected in such a way to maximize the measure.

2.1.4 Creating Diverse Learners

As previously discussed, creating diverse set of learners is the key to successfully train an ensemble of regressor or classifiers. Intuitively, we know that if all ensemble learners provide the same output, there would be nothing to benefit from their combination. The importance of diversity for ensemble systems is well established in [193], [194]. Ideally, we would like individual learners to be independent or even negatively correlated [121], [195].

Thus, the method for creating diversity plays an important role in training an ensemble system. Generally speaking, two different scenarios can be considered for creation of diversity first, to manipulate the architecture of the system, and second to alter the training data that a learning method receives [219]. In this section we discuss these approaches and the ways they can be used to generate diversity.

Altering the Architectures

The number of works into using different architectures for ensemble systems is relatively small, and thus it requires more attention. If we want to diversify the error between the ensemble members, we can intuitively conclude that using different types of learning algorithms may produce the required diversity [219]. Alternatively, we may create different architectures by using the same learning method with different setting of parameters (e.g. neural networks with different number of hidden units). Comparative studies between these two approaches conclude that using different training algorithms is generally more effective than manipulating learners topology [221], [222]. Partridge *et al.* used MLP and RBF neural networks in an ensemble to determine the effect of using different network types in diversity, and showed that using different network types is more productive than variation of network's hidden units. Islam *et al.* [186] proposed an ensemble of neural networks which supports different types of neural networks.

Altering Training Data

Several methods attempt to produce diverse learners by supplying each learner with a slightly different training set. This approach is the most widely addressed method in ensemble learning. Different learners can be given different parts of the training set. So they will expectedly learn different aspects of a same task. The very popular methods of this category are: bagging and boosting. Bagging (which stands for Bootstrap Aggregation) algorithm is one of the popular ensemble algorithms, which better suits for relatively small amount of training data. Each learner is trained using a subset of training set which is obtained by random sampling of the original training set with replacement. A very well-known version of bagging is the Random Forest, which is an ensemble of decision trees trained with a bagging mechanism. Boosting is another approach for altering training data of ensemble members, which is very similar to bagging. The difference between bagging and boosting is the resampling procedure. In bagging all samples have equal chance of being selected in each training data set, as resampling takes place with replacement. However, in boosting, the training data set for each subsequent learner increasingly focuses on instances misclassified by previously generated learner. Because of this sequential training, boosting is more suitable for classification problem.

Combining Ensemble Members

The last step of an ensemble system design is the fusion mechanism used to combine the individual learners. The aggregation method depends on the type of the outputs in part. This means that the combination method should be different for learners with discrete output in comparison with learners with continuous outputs. The following summarizes the aggregation methods for both cases.

i) **Learners with discrete outputs:** This case happens only in classification problem, when only discrete outputs are available at the learners' outputs. Note that the continuous valued outputs can easily be converted to discrete outputs (by assigning for the class with the highest output), but not vice versa. Thus, the methods of this section can be also applied to the continues outputs for classification problem. Here is a list of popular fusion algorithms for discrete outputs:

   - **Majority voting** is a winner-take-all strategy where the weight of the vote of all the learners are equal.
   - **Weighted majority voting** is a winner-take-all strategy where each learner has its own weight of vote which can be different from the others.
   - **Borda Count** has a different approach than majority voting. In this approach each output receives an order of support from each learner. This means that the output of ensemble system, shows the level of support for each class.

ii) **Learners with continues outputs:** This case has applications both in classification and regression problems. In classification problem each learner gives a certain level of support to each class which should then be interpreted to determine the output of the ensemble system. The case is different for regression problem since its output is continuous by the nature. The popular fusion methods for continuous outputs are:

- **Algebraic Combiners** determine the output of ensemble using an algebraic function of individual learners' outputs.
- **Min/Max/Meadian rule combiners** simply takes Min/Max/Meadian of individual learners for the output of ensemble system.
- **Product rule** chooses the class whose product of supports from each classifier is the highest (in classification problem).
- **Generalized mean** defines a generalized algebraic function for averaging the individual learners output. Note that all previous combiners are special cases of generalized mean.

2.2 Neural Networks for Dynamic Systems Identification

Basic neural network architectures are capable to learn static nonlinear maps between inputs and outputs. In the static systems the system outputs at an instance n, $y(n)$, depends only on the inputs $x(n)$ at the same instant $y(n) = f\left(x(n)\right)$. Thus, static neural networks can be used for modeling of such systems. However, the main challenge in system identification is to model the dynamic systems. In dynamic systems, the current output depends not only on the current outputs, but also on the previous behavior of the system (i.e. states of the system). There are several ways to form a dynamic structure from a formerly static neural network. This section introduces different network architectures which are used in this thesis for the purpose of system identification.

2.2.1 Nonlinear Autoregressive Exogenous Model (NARX)

The system identification problem consists of parameterizing a suitable identification model and trying to minimize the error between the plant and identified model by adjusting its parameters. The nonlinear functions in the representation of the plants are assumed to belong to known classes of models [83]. One of these models is Nonlinear Autoregressive Exogenous (NARX) model. NARX model parameterizes any nonlinear dynamics as a (nonlinear) function of a regressor vector which contains current value of the system's input, as well as, the past values of inputs and outputs. In other words, NARX model describes the dynamics of a system using the following equation:

$$y(n) = f(y(n-1), ..., y(n-d_y), u(n), ..., u(n-d_u))$$

The nonlinear function f in the above equation can be approximated using different learning algorithms such as MLP neural networks, RBF neural networks, wavelets, and SVM [84].

NARX Model Structures

In NARX model inputs and outputs are fed into the model through tapped delay lines. Depending on the configuration of the feedback path, two types of NARX structures exist. In the following we present these two models.

1. *Parallel Identiˇcation Model*: In this model the feedback comes from the estimated output rather than the actual output of the plant itself. This is shown in Figure 2.2. In this structure

the estimated output at each instant n is described by the following equation:

$$\hat{y}(n) = f(\hat{y}(n-1), ..., \hat{y}(n-d_y), u(n), ..., u(n-d_u))$$

where $\hat{y}$ is the output of the estimated model, and u is the external input, d_y and d_u are the input and output delays, respectively (tapped delay lines in Figure 2.2). Identification then involves the estimation of f with for example a neural network. By assumption the plant is bounded-input bounded-output (BIBO) stable. This guarantees that all the signals in the plant are uniformly bounded. However, the stability of the identified model (e.g. neural network) can not be assured and has to proved. Thus, if a parallel model is used, the convergence of network parameters is not guaranteed [83]. To ensure the stability of the identification method the series-parallel, which is described below, is used.

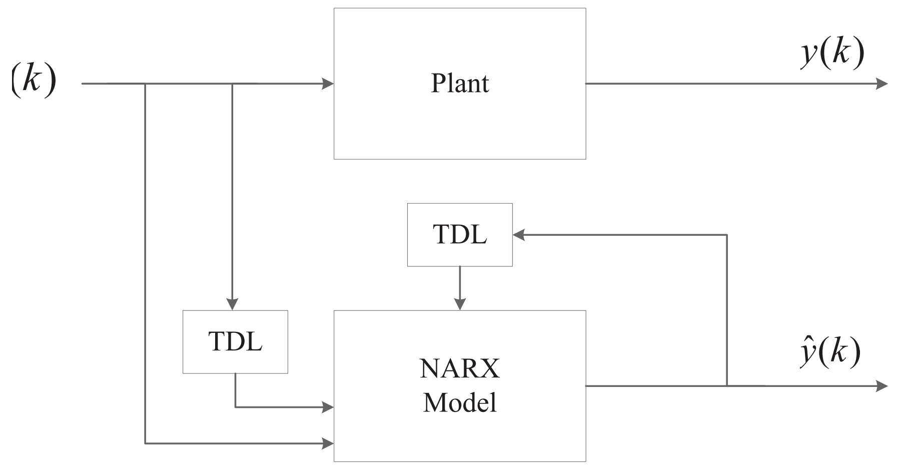


Figure 2.2: Parallel architecture of NARX model [83].

2. *Series-Parallel Model:* In contrast with the parallel model where estimated outputs are fed into the identification model, in series-parallel structure the actual outputs of the plant are

fed back to the identification model. This structure is shown in Figure 2.3. This forms the following equation for the identification model:

$$\hat{y}(n) = f(y(n-1), ..., y(n-d_y), u(n), ..., u(n-d_u))$$

In this model, the inputs and outputs of the plant form the input vector of the regressor function f whose output $\hat{y}(n)$ corresponds to estimated output of the plant at time n. Series-parallel structure has several advantages as compared with parallel model [83]. Since the plant itself is assumed to be BIBO stable, all signals which are used in identification process, which are inputs of the regressor f, are bounded. This guarantees that the identified model would be stable as there is no feedback from estimated output to the input of the identification model.

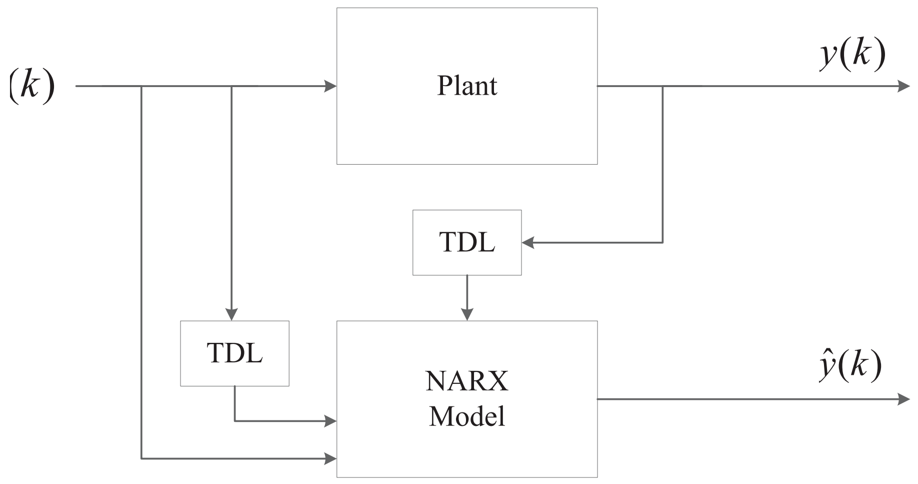


Figure 2.3: Series-parallel architecture of NARX model [83].

Also, assuming that $\hat{y}(n) \approx y(n)$, the series-parallel model can be replaced by a parallel model during *testing phase* without serious consequences. Thus, in this thesis we always use series-parallel structure during *identiˇcation (training) phase*. The series-parallel structure would be replaced by parallel structure during testing stage.

2.2.2 Multi-layer Perceptron

Multi-layer perceptron (MLP) is a type of multi-layer feedforward neural network. All nodes are fully connected to the nodes in adjacent layers, but there is no connection between neurons of the same layer or between neurons of non-adjacent layers. The structure of a multi-layer perceptron is shown in Figure 2.4.

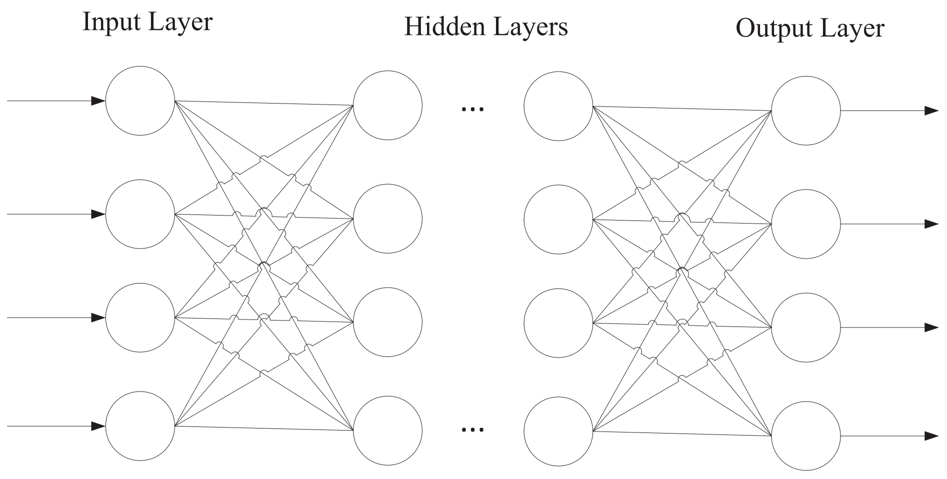


Figure 2.4: A multi-layer perceptron.

Inputs to the network are passed through each node in the input layer. The outputs of the input layer become the inputs of the next layer. Thus the last layer works as the output layer. The output of the i^{th} neuron in the k^{th} layer can be described as the following equation:

$$\begin{aligned} z_i^{(k)} &= \sum_{j=1}^{n_{k-1}} w_{ij}^k x_j^{(k-1)} \quad b_i^{(k)} \\ x_i^{(k)} &= a(z_i^{(k)}) \end{aligned}$$

where $x_i^{(k)}$ is the output of the i^{th} neuron in the k^{th} layer, w_{ij}^ks are the connection weights to the i^{th} neuron in the k^{th} layer, $b_i^(k)$ is the bias term, and $a()$ is the neuron's activation function. Two typical choices of activation are sigmoidal function (i.e. $a(z) = \frac{1}{1+e^{-z}}$), and $\tanh(z)$.

There are several algorithms for training MLP neural networks. The most popular training algorithm is the backpropagation. Backpropagation is a supervised learning algorithm. In this method the connection weights get updated in each iteration by comparing the network output with the expected output. For more information on backpropagation refer to [20], [23].

Remark 2.1. ***MLP-NARX.*** *The use of MLP neural networks in NARX model for dynamical systems identiˇcation has been reported in several publications including but not limited to [17], [18], [19], [20], [21], [22], [23]. In this research, MLP neural network is used in an NARX model to identify the jet engine dynamics. Figure 2.5 shows the structure of the MLP-NARX model during identiˇcation (i.e. training) stage.*

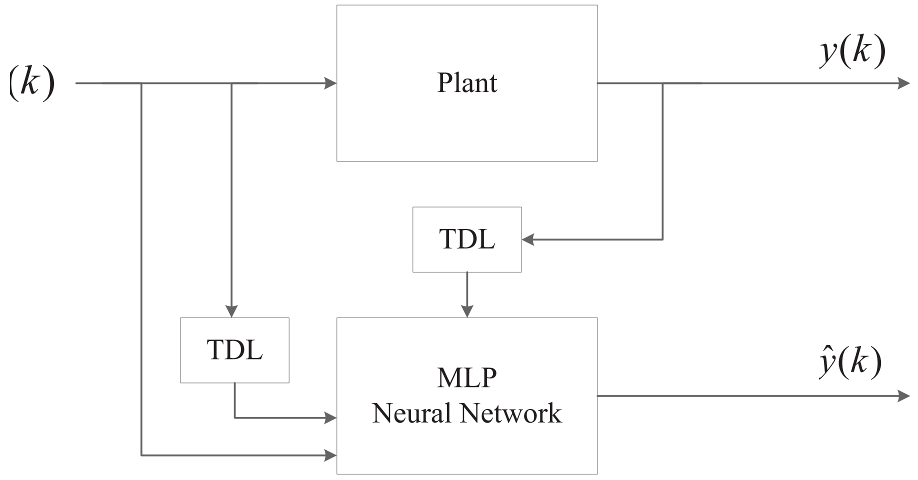


Figure 2.5: Nonlinear system identification using series-parallel MLP-NARX (training stage).

The series-parallel model would be replaced by parallel structure during testing phase as shown in Figure 2.6.

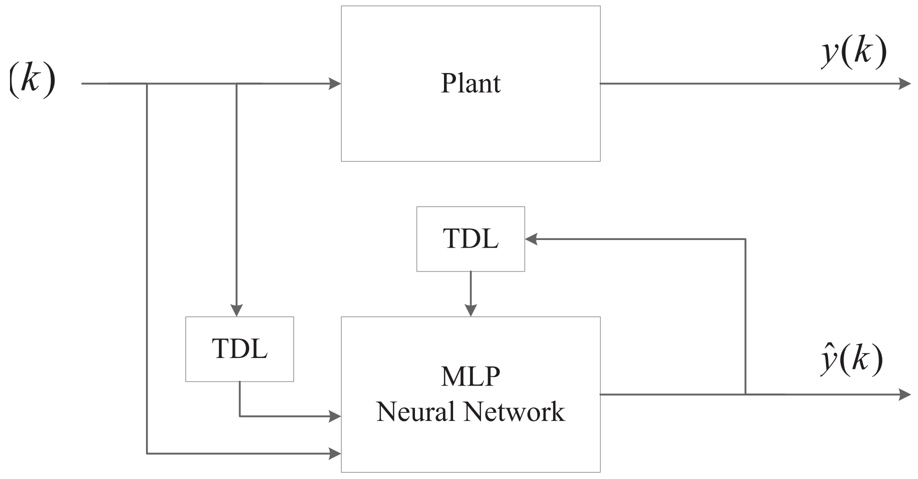


Figure 2.6: Nonlinear system identification using parallel MLP-NARX (testing stage).

2.2.3 RBF Neural Networks

RBF neural networks have attracted much attention because of their generalization ability and their simple structure which lessens the calculations as compared with multi-layer feed-forward neural networks [31]. RBF neural networks are feed-forward networks with only one hidden layer. The hidden layer of RBF network consists of RBF neurons. The wights between input layer and hidden layer are simply unity weights. Only the weights between the hidden layer (RBF neurons) and output layer are adjustable. In other words inputs are directly connected to the RBF neurons. The structure of RBF neural networks is shown in Figure 2.7.

ural network

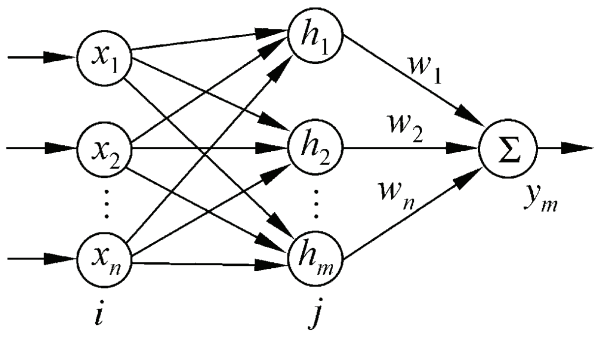
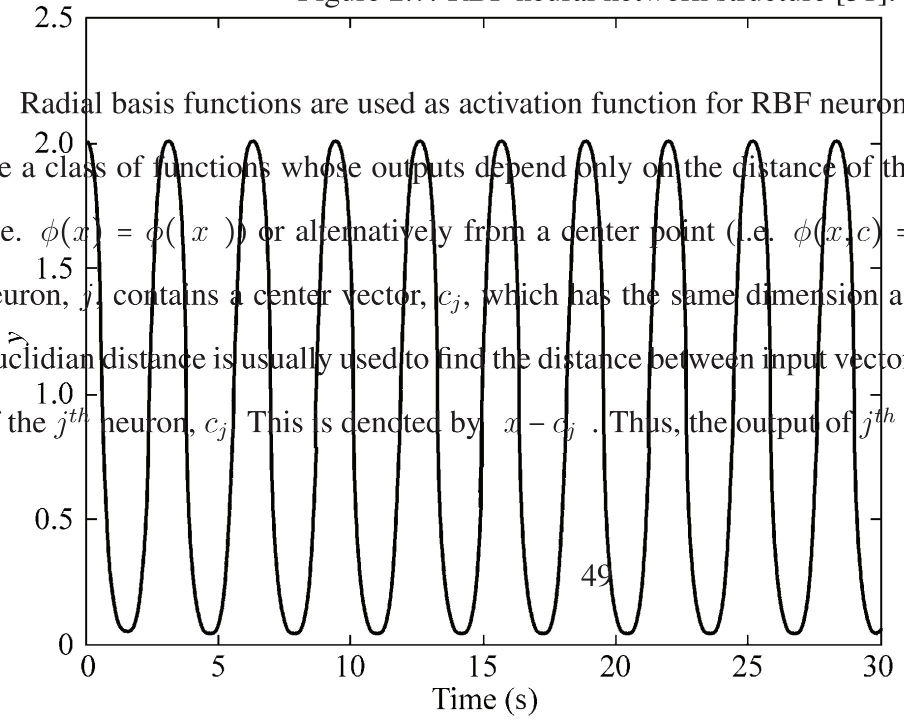


Figure 2.7: RBF neural network structure [31].

of RBF



Radial basis functions are used as activation function for RBF neurons. Radial basis functions are a class of functions whose outputs depend only on the distance of their input from the origin (i.e. $\phi(x) = \phi(\|x\|)$) or alternatively from a center point (i.e. $\phi(x, c) = \phi(\|x - c\|)$). Each RBF neuron, j, contains a center vector, c_j, which has the same dimension as its input vector x. The Euclidian distance is usually used to find the distance between input vector, x, and the center vector of the j^{th} neuron, c_j. This is denoted by $\|x - c_j\|$. Thus, the output of j^{th} RBF neuron is:

value of RBF is

$$h_j = \phi(\ \ x - c_j\ \ ), j = 1, ..., m$$

Most commonly, a Gaussian function is used as the activation function. In this case the above equation becomes:

$$h_j = \phi(\ \ x - c_j\ \ ) = \exp\quad -\frac{x - c_j}{2\sigma_j^2}\quad , j = 1, ..., m$$

Then the output of the network shown in Figure 2.7 would be:

$$y_m = \sum_{j=1}^{m} h_j w_j$$

The parameters which require training are the radial basis function centers, radial basis function widths, and output layer weights. Several training algorithms can be used for training *centers* of RBF units including but not limited to *subsets of data points*, *orthogonal least squares*, and *clustering algorithms*. In *subsets of data points* the centers of radial basis functions are selected *randomly* from the set of training data. On the other hand, *orthogonal least squares* method incrementally adds RBF units one-by-one each time choosing the data point which reduces the error the most as the center of the newly added unit. *Clustering algorithms* aim to find the set of points which most accurately represent the distribution of the data.

Among clustering algorithms *k-means clustering* algorithm is more common and it is used in this thesis. In this method, all the data point are *ˇrst* randomly assigned to the k clusters. *Second* the mean point of each cluster, S_j, is calculated. *Third* each point is reassigned to the cluster which has the closest mean. These steps are repeated until no point changes its cluster. It can be shown that this algorithm is equivalent to minimizing the following sum of squares clustering function:

$$J = \sum_{j=1}^{k} \sum_{x_i \in S_j} \quad x_i - c_j$$

The radial basis function widths are usually set to the covariance of data points in each cluster in this algorithm. Least squares is mostly used to determine *output layer weights*. As previously explained the RBF model output is governed by the following equation.

$$y_m = \sum_{j=1}^{M} h_j w_j \tag{2.17}$$

where h_j is the j^{th} RBF neuron output and w_js are the output layer weights. The least square method minimizes the following error function with respect to the weights of the output layer.

$$S = \sum_{i=1}^{N} (y_i - y_m(x_i))^2$$

where N is the number of training samples, y_i is the target of the i^{th} sample, and $y_m(x_i)$ is the predicted target for the i^{th} sample. By adding a weight penalty term to the above error function we have:

$$C = \sum_{i=1}^{N} (y_i - y_m(x_i))^2 \quad \sum_{j=1}^{M} \lambda_j w_j^2$$

The above function is called ridge regression. Minimizing the above function with respect to the w_js gives:

$$\frac{\partial C}{\partial w_j} = 2 \sum_{i=1}^{N} (y_i - y_m(x_i)) \frac{\partial y_m(x_i)}{\partial w_j} \quad 2\lambda_j w_j$$

according to equation 2.17 we have, $\frac{\partial y_m(x_i)}{\partial w_j} = h_j(x_i)$ then:

$$\sum_{i=1}^{N} h_j(x_i) y_m(x_i) \quad \lambda_j \hat{w}_j = \sum_{i=1}^{N} y_i h_j(x_i)$$

where $\hat{w_j}$ is the optimized value of w_j. Showing it in the matrix form we have:

$$h_j^T y_m \quad \lambda_j \hat{w}_j = h_j^T y_i$$

or equivalently:

$$h^T y_m \quad \Lambda \hat{w} = h^t y$$

where $h = [h_1, ..., h_M]$, $y_m = [y_m(x_1), ..., y_m(x_n)]^T$, and $\Lambda = diag \quad \lambda_i$. Simplifying the above equation results in the optimal weights of the output layer as follows:

$$\hat{w} = (h^T h \quad \Lambda)^{-1} h^T y$$

Remark 2.2. ***RBF-NARX.*** *The use of RBF neural networks in the NARX model for dynamical system identiˇcation has been reported in several publications including but not limited to [8], [10], [11], [12], [13], [14], [15], [16]. In this research, RBF neural networks is used in an NARX model to identify the jet engine dynamics. Figure 2.8 shows the structure of the RBF-NARX model used during identiˇcation (i.e. training) stage.*

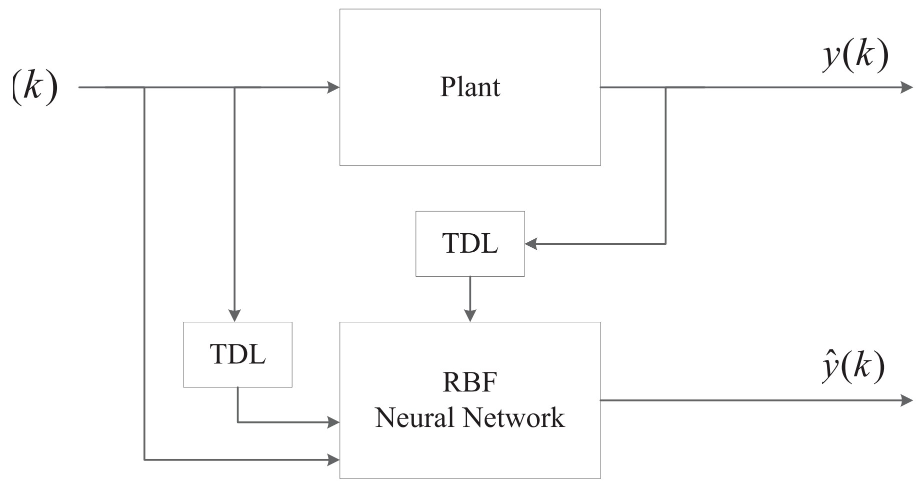


Figure 2.8: Nonlinear system identification using series-parallel RBF-NARX (training stage).

The series-parallel model would be replaced by parallel structure during testing phase as shown in Figure 2.9.

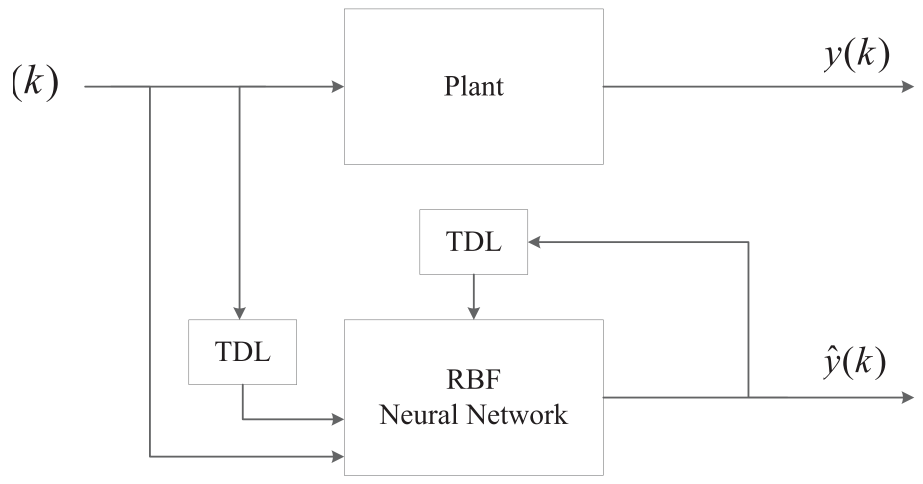


Figure 2.9: Nonlinear system identification using parallel RBF-NARX (testing stage).

2.2.4 Support Vector Regression

In support vector regression, the idea is to map the input data into a high dimensional feature space (H) using a generally nonlinear mapping (ϕ), and then use a linear regression in this high-dimensional space. Thus, the original problem is to estimate a generally nonlinear function based on the available data $D = \quad (x_1, y_1), ..., (x_i, y_i) \quad$ from n-dimensional input space ($x_i > R^n$) to assuming one-dimensional output space ($y > R$). Note that one-dimensional output space dimension does not impose any limitations, as a problem with multi-dimensional output space can be decomposed into several problems whose output space is one-dimensional. SVR converts this nonlinear regression problem into a linear regression problem in a higher dimensional input-space using a nonlinear function $\phi(x)$. This means that the sample space would be mapped to $D' = \quad (\phi(x_1), y_1), ..., (\phi(x_i), y_i) \quad$. Now the regression problem (in this thesis system identification) would be to find a function f such that:

$$f(x) = \sum_{i=1}^{l} w_i \phi(x_i) \quad b \tag{2.18}$$

where $\phi(x_i)$ is the input feature, w_i and b are regression coefficients and bias respectively. Assuming the function $f(x)$ approximates (x_i, y_i) with precision ϵ, then the coefficients w_i and b are determined by minimizing the following risk function:

$$R(C) = C \sum_{i=1}^{l} L_\epsilon \quad \frac{1}{2} \quad w \quad ^2$$

where $L_\epsilon = max \quad 0, \; y - f(x) \; - \epsilon$, and it is called $\epsilon-$insensitivity loss function. Note that the L_ϵ does not penalize errors less than $\epsilon \quad 0$. The second term, $\frac{1}{2} \quad w \quad ^2$ is a measure of function flatness.

C is a constant to regulate the trade-off between training error and model flatness, and is there to penalize the large deviations from final predicted hyperplane. Introduction of the slack variables $\xi^{(*)}$ converts the problem to minimizing the following equation:

$$\begin{array}{rl} min & \frac{1}{2} \;\; w \;\; ^2 \quad C\sum_{i=1}^{l}(\xi_i^{(*)} \quad \xi_i) \\ \text{subject to:} & ((wx_i) \quad b) - y_i \quad \epsilon \quad \xi_i \\ & y_i - ((wx_i) \quad b) \quad \epsilon \quad \xi_i \\ & \xi_i^{(*)}, \xi_i \quad 0 \end{array}$$

Introducing the Lagrange multipliers $\alpha_i^{(*)}$, and $\eta_i^{(*)}$, the Lagrangian primal function of the dual optimization problem becomes:

$$\mathcal{L}_{\mathcal{P}} = \frac{1}{2} \;\; w \;\; ^2 - \sum_{i=1}^{l} \alpha_i(\xi_i - y_i \quad (w.x_i) - b - \epsilon) - \sum_{i=1}^{l} \alpha_i^*(\xi_i^* - (w.x_i) \quad b \quad y_i - \epsilon) - \sum_{i=1}^{l}(\eta_i\xi_i \quad \eta_i^*\xi_i^*)$$

Taking the derivatives with respect to w, b, ξ, and ξ^* to find KKT conditions leads to $w = \sum_{i=1}^{l}(\alpha_i^* - \alpha_i)x_i$. Substituting this new relation equation (2.2.4) has the form $f(x) = \sum_{i=1}^{l}(\alpha_i^* - \alpha_i)\phi(x_i)^T\phi(x_i) \quad b$. For computational convenience, the form $\phi(x_i)^T\phi(x_i)$ is defined as the kernel function which has the form:

$$k(x_i, x_j) = \phi(x_i)^T\phi(x_i)$$

Thus the approximated function has the form:

$$f(x) = \sum_{i=1}^{l} (\alpha_i^* - \alpha_i) k(x_i, x_j) \quad b$$

The Lagrange multipliers can be obtained by maximizing the following dual problem:

$$W(\alpha^*) = \frac{1}{2} \sum_{i,j=1}^{l} (\alpha_i - \alpha_i^*) k(x_i, x_j)(\alpha_i - \alpha_i^*) \quad \epsilon \sum_{i=1}^{l} (\alpha_i \quad \alpha_i^*) - \sum_{i=1}^{l} (\alpha_i - \alpha_i^*) y_i$$

subject to:

$$\begin{aligned} \sum_{i=1}^{l} (\alpha_i - \alpha_i^*) &= 0 \\ 0 \quad \alpha_i^* \quad C & \end{aligned}$$

The nonzero coefficients obtained from the above problem form a hyperplane which is the so-called support vectors.

Remark 2.3. ***SVR-NARX.*** *The use of SVR-NARX model in system identiˇcation has been reported in various publications. A general framework for nonlinear system identiˇcation with SVR based on NARX model is presented in [1]. In another framework, [2] combines Least-Square Support Vector Machines (LS-SVM) with NARX model for identiˇcation of Weiner-Hammerstein systems. Other SVR-based system identiˇcation methods in companionship with ARX models are reported in [4, 3, 5, 6]. In this research, support vector regression is used in an NARX model to identify the jet engine dynamics. Figure 2.10 shows the structure of the SVM-NARX model used during identiˇcation (i.e. training) stage.*

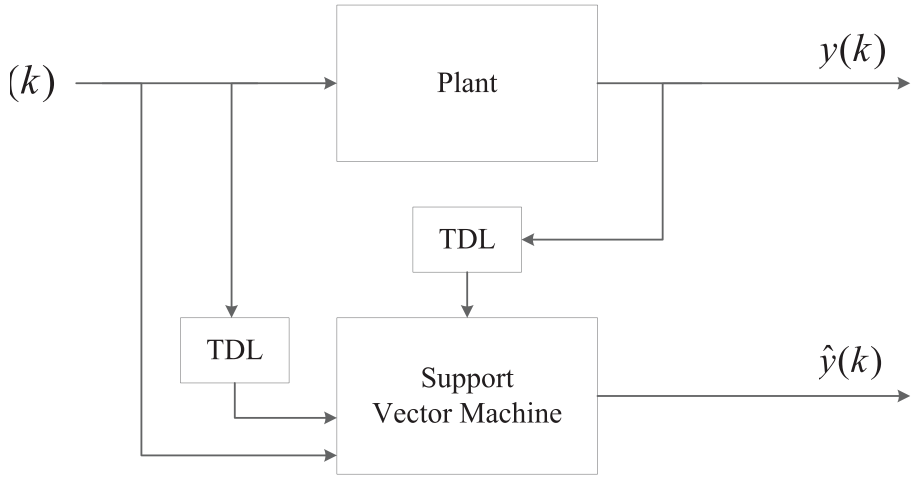


Figure 2.10: Nonlinear system identification using series-parallel SVM-NARX (training stage).

The series-parallel model would be replaced by parallel structure during testing phase as shown in Figure 2.11.

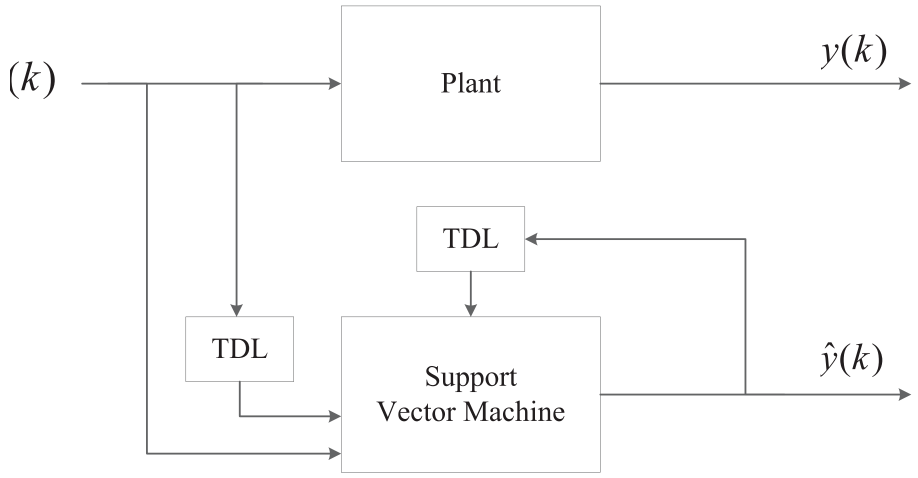


Figure 2.11: Nonlinear system identification using parallel SVM-NARX (testing stage).

2.3 Jet Engine Overview

A gas turbine jet engine can be introduced as an internal combustion engine which basically consists of an upstream rotating compressor coupled to a downstream turbine through a combustion chamber. The mixture of air and fuel is ignited in the combustor, then the produced gas flows to the turbine. Since the gas in the turbine is of high volume as well as high velocity, the turbine blades start spinning. This generated energy can be used for different purposes. One of these is to use gas turbine as an aircraft jet engine where generated energy propels the aircraft.

In summary, a single spool jet engine, works as follows:

1. *First*, the temperature and pressure of the intake air increases in the compressor (compression phase).

2. *Second*, the mixture of fuel and high pressure air coming from compressor is ignited in the combustion chamber. In this phase it is desirable to keep the pressure unchanged while increasing the temperature and volume (heating phase).

3. An expansion phase in the turbine where the flow energy converts to the mechanical energy to power the compressor. In this phase the air temperature and pressure drops.

4. A further expansion phase in the nozzle where the flow speed increases and it returns to the inlet pressure. The thrust needed for the aircraft to move forward is provided by this high speed gas.

In this thesis, the fault detection and isolation of a single spool jet engine is studied. A single spool engine consists of a compressor, a combustion chamber, and a shaft which is driven by a single turbine. The schematic of a typical single spool jet engine is shown in the Figure 2.12.

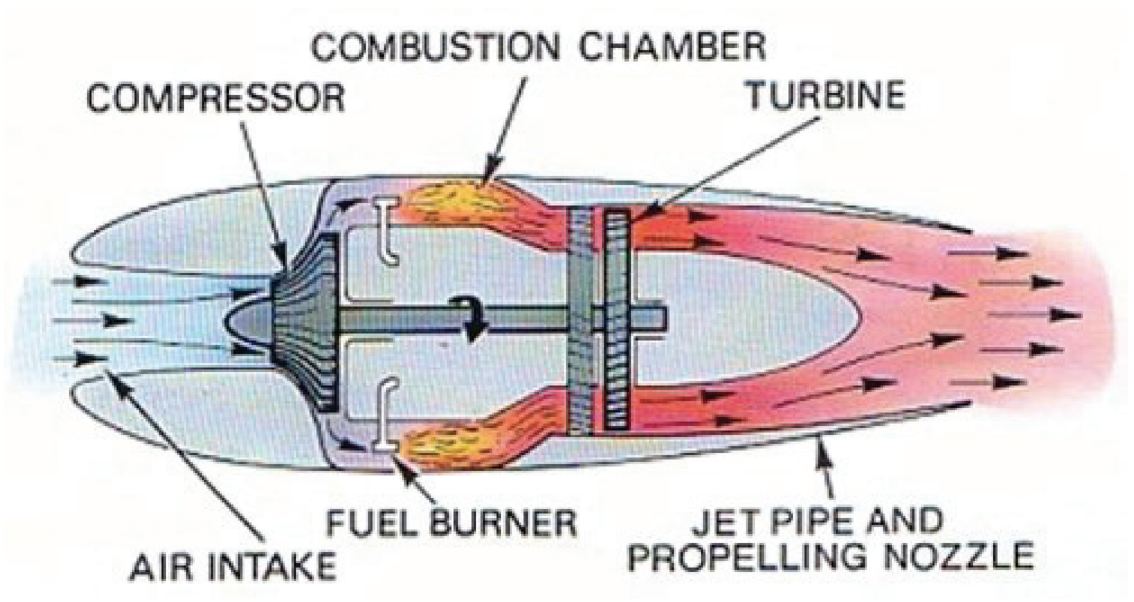


Figure 2.12: schematic of a single spool turbofan jet engine [237].

The mathematical model of a single-spool jet engine is described in this section.

2.3.1 Jet Engine Mathematical Model

This section presents the mathematical model of a single spool jet engine. Although this research is based on the data-driven approaches, but the data used for the purpose of simulations is generated by a mathematical model of a single-spool jet engine developed in Simulink environment. Thus, it is necessary to review the mathematical model of a jet engine dynamics. We should also note that using a simulation model is of special interest for generating a faulty engine data, since it may not be generally available for the real jet engine.

Rotor Dynamics

By applying the concept of energy balance between the shaft and the compressor, the following equation would be obtained:

$$\frac{dE}{dt} = \eta_{mech} W_T - W_C$$

where $E = \frac{J(\frac{N.2\pi}{60})^2}{2}$, η_{mech} denotes the mechanical efficiency, W_T denotes the power generated by the turbine, W_C denotes the power consumed by the compressor and J is the rotor moment of inertia. N stands for the number of turns which is a function of time (rotation per minute).

Volume Dynamics

To model the volume dynamics, it is assumed that the engine components themselves have zero volume. This assumption simplifies the modeling process by eliminating of algebraic loops, and it allows to develop a generic model based on jet engine dynamics. Also, we assume that the gas has zero speed and has homogenous properties over volume. Taking the above considerations into account, the volume dynamics would be described by the following equation:

$$\dot{P} = \frac{RT}{V}(\sum \dot{m}_{in} - \sum \dot{m}_{out}) \tag{2.19}$$

where P denotes the pressure, R denotes the gas constant, T denotes the temperature, V stands for the volume, $\dot{m}_{in}$ and $\dot{m}_{out}$ denote the input mass flow and the output mass flow, respectively.

2.3.2 Modeling of Engine Components

Intake Duct

The intake duct is positioned before the compressor and supplies the engine with the required air flow at highest possible pressure. In the intake duct air's velocity decreases while its pressure and temperature increase. Assuming adiabatic process, the inlet pressure ratio to ambient pressure would be described by the following equation:

$$\frac{P_d}{P_{amb}} = [1 \quad \eta_d \frac{\gamma - 1}{2} M^2]^{\frac{\gamma}{\gamma-1}} \tag{2.20}$$

where M denotes the Mach number and P_{amb} denotes the ambient pressure, η_d denotes the isentropic efficiency and γ stands for specific heat capacity ratio. The inlet temperature ratio ,$\frac{T_d}{T_{amb}}$, can also be expressed in terms of M as

$$\frac{T_d}{T_{amb}} = 1 \quad \frac{\gamma - 1}{2} M^2 \tag{2.21}$$

where T_{amb} is the ambient temperature.

Compressor

The role of compressor in a jet engine is to provide high pressure air to the combustion chamber. For the Simulink model used in this thesis, the behavior of compressor which is considered as a quasi-steady component is determined by the compressor performance map, obtained from a commercial software package called GSP [226].

For a given pressure ratio, π_C, and the corrected rotational speed, $N \quad \overline{\pi}$, the corrected mass flow rate ($\dot{m}_C\frac{\sqrt{\theta}}{\delta}$), and η_C can be obtained from the performance map by using a proper interpolation technique, where $\theta = \frac{T_i}{T_o}$ and $\delta = \frac{P_i}{P_o}$, that is $\dot{m}_C\frac{\sqrt{\theta}}{\delta} = f_{\dot{m}_C}\ N \quad \overline{\theta}, \pi_C$ and $\eta_C = f_{\eta_C}\ N \quad \overline{\theta}, \pi_C$.

Once these parameters are obtained, the compressor temperature rise can be found by the following formula [97]:

$$T_o = T_i\ 1 \quad \frac{1}{\eta_C}(\pi_C{}^{\frac{\gamma-1}{\gamma}} - 1) \tag{2.22}$$

where T_o denotes the compressor output temperature, T_i denotes the compressor input temperature and η_C is the efficiency of the compressor. Also, the mechanical power is obtained as follows:

$$W_C = \dot{m}_C C_p (T_o - T_i) \tag{2.23}$$

where $\dot{m}_C$ denotes the compressor mass flow rate and C_p denotes the specific heat at a constant pressure. In a single spool engine, the connection between the compressor and the turbine is made by the only shaft in the system. The speed of the engine is determined by the shaft (rotor) speed and in turn the speed is a function of the power which the turbine generates and also the total moment of inertia of the rotary system. The relation between the power that the compressor consumes and the speed of the shaft is given by the following formula:

$$W_C = \frac{J(2\pi N)^2}{2} \tag{2.24}$$

where J denotes the moment of inertia of the shaft and N is the rotor speed indicated in RPM.

Combustion Chamber

As previously explained, the combustion chamber is where the mixture of fuel and high pressure air flowing from the compressor is ignited. As a result, the temperature increases while the gas pressure is desired to be kept unchanged. The combustion chamber represents both the energy accumulation and the volume dynamics between the compressor and the turbine at the same time. The dynamics of combustion chamber is described by the following equations [97]:

$$\dot{P}_{CC} = \frac{P_{CC}}{T_{CC}}\dot{T}_{CC} \quad \frac{\gamma R T_{CC}}{V_{CC}}(\dot{m}_C \quad \dot{m}_f \quad \dot{m}_T) \tag{2.25}$$

$$\dot{T}_{CC} = \frac{1}{c_v m_{CC}}\left[(c_p T_C \dot{m}_C \quad \eta_{CC} H_u \dot{m}_f - c_p T_{CC}\dot{m}_T) - c_v T_{CC}(\dot{m}_C \quad \dot{m}_f \quad \dot{m}_T)\right] \tag{2.26}$$

where T_{CC} and P_{CC} denote the combustion chamber temperature and pressure respectively, $\dot{m}_C$ and $\dot{T}_{CC}$ denote the compressor mass flow rate and the turbine mass flow rate respectively, $\dot{m}_f$ denotes the fuel flow rate, γ denotes the heat capacity ratio, R stands for the gas constant, c_p and c_v stand for specific heat at constant pressure and volume respectively, and H_u is the fuel specific heat.

Turbine

In a jet engine, the turbine function is to extract a portion of the pressure and kinetic energy from the high-temperature gases coming from the combustion in order to drive the compressor and accessories. In a typical jet engine, about 75 percent of the internally produced power is consumed to derive the compressor and the rest is used for generating the required thrust [227]. Turbine is a rotatory component through which the gas of high temperature flows and its behavior, like the compressor, is represented by characteristic maps (from the software package GSP [226]). Given the pressure ratio, π_T and the corrected rotational speed, $N \quad \overline{\theta}$, the corrected mass flow rate, $\dot{m}_T\frac{\sqrt{\theta}}{\delta}$, and the efficiency, η_T are found from the performance map, that is $\dot{m}_T\frac{\sqrt{\theta}}{\delta} = f_{\dot{m}_T}(\frac{\sqrt{\theta}}{\delta}, \pi_T)$ and $\eta_T = f_{\eta_T}(\frac{N}{\sqrt{\theta}}, \pi_T)$. The temperature drop and the turbine mechanical power which is proportional to the temperature decrease in the turbine are given as follows:

$$T_o = T_i[1 - \eta_T(1 - {\pi_T}^{\frac{\gamma-1}{\gamma}})] \tag{2.27}$$

$$W_T = \dot{m}_T c_p(T_i - T_o) \tag{2.28}$$

where $\dot{m}_T$ denotes the compressor mass flow rate, and c_p denotes the specific heat at a constant pressure. In a jet engine, the power generated by the turbine and the power consumed by the compressor are proportional.

Nozzle

Nozzle is the last part of a jet engine in which the working fluid is expanded to produce a high-velocity jet. The high pressure exhaust gas is accelerated in a jet pipe placed between the turbine outlet and the nozzle throat to come close to the ambient pressure and consequently, to produce the necessary thrust. The nozzle exit temperature T_{n_o} is given by the equation as follows [97]:

$$T_{n_i} - T_{n_o} = \eta_n T_{n_o}[1 - (\frac{1}{P_{n_i} \quad P_{amb}})^{\frac{\gamma-1}{\gamma}}] \tag{2.29}$$

where η_n is the nozzle efficiency, T_{n_i} denotes the nozzle inlet temperature, T_{n_o} is the nozzle outlet temperature, P_{n_i} is the nozzle inlet pressure, and P_{amb} is the ambient pressure. If the condition (2.30) holds, the mass flow is computed by equation (2.31), that is:

$$\frac{P_{amb}}{P_{n_i}} \quad [1 \quad \frac{1-\gamma}{\eta_n(1 \quad \gamma)}]^{\frac{\gamma}{\gamma-1}} \tag{2.30}$$

$$\frac{\dot{m}_C \quad \overline{T_{n_i}}}{P_{n_i}} = \frac{u}{\overline{T_{n_i}}} \frac{A_n}{R} \frac{P_{amb}}{P_{n_i}} \frac{T_{n_i}}{T_{n_o}} \tag{2.31}$$

where $\frac{u}{\sqrt{T_{n_i}}} = \quad \overline{2c_p\eta_n(1-(\frac{P_{amb}}{P_{n_i}}))^{\frac{\gamma-1}{\gamma}}}$ and $\frac{T_{n_i}}{T_{n_o}} = 1 - \eta_n(1-(\frac{P_{amb}}{P_{n_i}}))^{\frac{\gamma-1}{\gamma}}$. On the other hand, if the condition (2.30) does not hold, the nozzle mass flow rate is give as follows:

$$\frac{\dot{m}_C \quad \overline{T_{n_i}}}{P_{n_i}} = \frac{u}{\overline{T_{n_i}}} \frac{A_n}{R} \frac{P_{crit}}{P_{n_i}} \frac{T_{n_i}}{T_{crit}} \tag{2.32}$$

where $\frac{u}{\sqrt{T_{n_i}}} = \frac{2\gamma R}{\gamma+1}$, $\frac{P_{crit}}{P_{n_i}} = (1 - \frac{1}{\eta_n}(\frac{\gamma-1}{\gamma+1}))^{\frac{\gamma}{\gamma-1}}$ and $\frac{T_{crit}}{T_{n_i}} = \frac{2}{\gamma+1}$. It is assumed that $P_{n_i} = P_{LT}$ and $T_{n_i} = T_M$. T_M is found from the energy balance relation in the mixer as follows:

$$T_M = \frac{\dot{m}_T T_T \quad \beta\dot{m}_C T_C}{\dot{m}_T \quad \beta\dot{m}_C} \tag{2.33}$$

where β is the bypass ratio. A schematic view of the information explained above is shown in Figure 2.13.

Fig. 1. General architecture of our proposed MM-based FDI scheme.

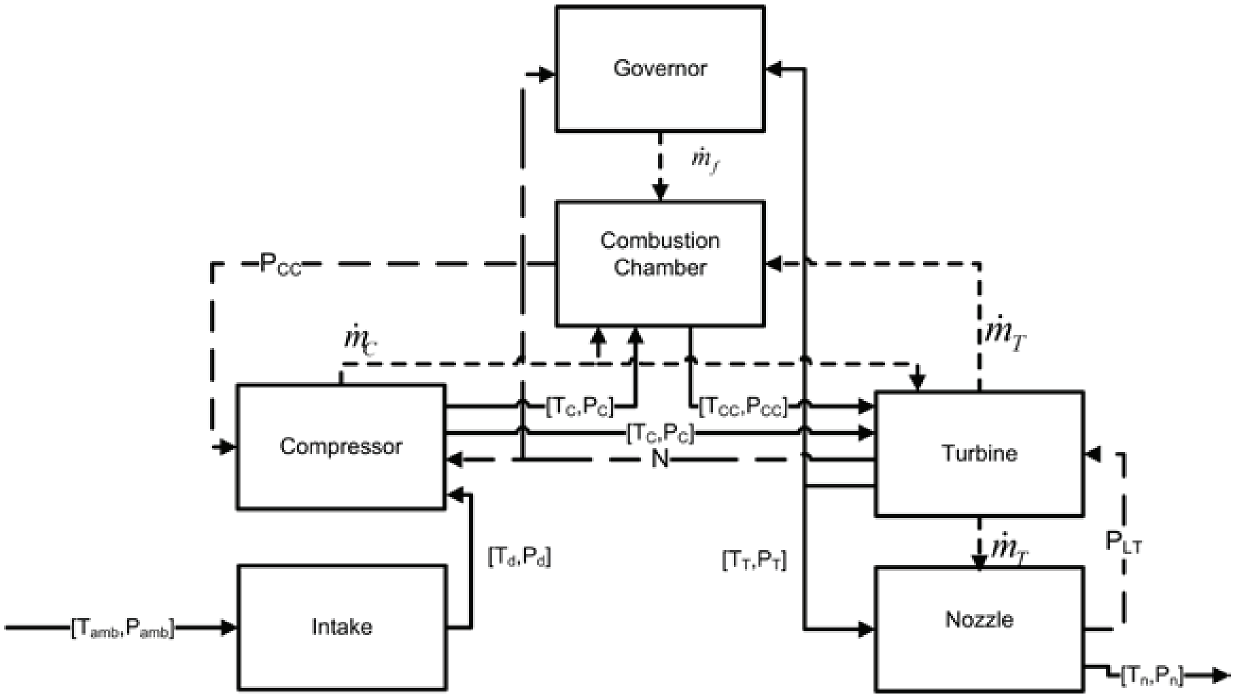


Fig. 2. Information flow diagram in a modular modeling of the jet engine dynamics.

Figure 2.13: Aircraft jet engine modules and information flow chart.

The Set of Nonlinear Equations

ech denotes the mechanical efficiency and J denotes the inertia of the shaft connecting the compressor to th

The set of nonlinear equations which describe the behavior of a single-spool jet engine is obtained

ore, using [21] the following dynamics for the fuel mass flow rate are considered

in [97], which shows that the system is a nonlinear 4^{th} order system. At the inlet of the jet engine the ratio of duct temperature to ambient temperature, and duct temperature to ambient temperature are described by the following equation:

$$\tau\frac{d\dot{m}_f}{dt}+\dot{m}_f = Gu_{fd}$$

$$\frac{T}{T_{amb}} = 1 \quad \frac{\gamma-1}{2}M^2$$

the time constant, G is the gain associated with the fuel valve, and u_{fd} denotes the fuel demand which is feedback from the rotational speed as described in [21]. A modular Simulink model is developed to sin engine nonlinear dynamics as described by equations (5) and (6). Figure 2 shows the information flow p ink model of the engine.

e 3 shows the series of steady states that are obtained from our nonlinear model and the GSP [24] at PLA o 1. At each point, the initial condition of the PLA is set equal to 0.3 followed by a transient to reach to t esponding to the desired PLA. Since the steady state corresponding to each PLA is independent of the p transients (unless the compressor surges), it provides a suitable basis for comparison. As can be obser he responses corresponding to our model and the GSP match each other within an acceptable error toleran difference between the two representations is manifested in terms of the complexity of the mathemati ave used, by taking into account that our structure is simpler as compared to the more complicated repre

$$\frac{P}{P_{amb}} = 1 \quad \eta_d \frac{\gamma - 1}{2} M^2 \quad ^{\frac{\gamma}{\gamma - 1}}$$

The set of nonlinear equations describing a single spool jet engine behavior is given by:

$$\begin{aligned}
\dot{T}_{CC} &= \frac{1}{c_\nu m_{CC}} \left(c_P T_C \dot{m}_C \quad \eta_{CC} H_u \dot{m}_f - c_P T_{CC} \dot{m}_T\right) - c_\nu T_{CC}(\dot{m}_C \quad \dot{m}_f - \dot{m}_T) \\
\dot{N} &= \frac{\eta_{mech} \dot{m}_T c_P (T_{CC} - T_T) - m_C c_P (\dot{T}_C - T_d)}{JN(\frac{\pi^2}{30})} \\
\dot{P}_T &= \frac{RT_{M_i}}{V_{M_i}} (\dot{m}_T \quad \frac{\beta}{1 \quad \beta} \dot{m}_C - \dot{m}_n) \\
\dot{P}_{CC} &= \frac{P_{CC}}{T_{CC}} \dot{T}_{CC} \quad \frac{\gamma R T_{CC}}{V_{CC}} (\dot{m}_C \quad \dot{m}_f - \dot{m}_T)
\end{aligned}$$

where T_{CC} is temperature of combustion chamber, N stands for the rotational speed, P_C is the compressor pressure, P_T is the turbine pressure, m_{CC} is the mass flow in combustion chamber, c_ν is the specific heat at constant volume, c_p is the specific heat at constant pressure, T_C is the compressor temperature, T_T is the turbine temperature, $\dot{m}_C$ is the compressor mass flow rate, η_{CC} is the combustion chamber efficiency, H_u is the fuel specific heat, $\dot{m}_f$ is the fuel mass flow rate, η_{mech} is the mechanical efficiency, T_d is the diffuser temperature, $\dot{m}_n$ is the nozzle mass flow rate, β is the bypass ratio, T_{M_i} is the mixer temperature, V_{M_i} is the volume mixer, and P_{CC} is the combustion chamber pressure.

The state variables in the single-spool jet engine are selected as:

$$x = [T_{CC}, N, P_T, P_{CC}]$$

The T_C and T_T are given by $T_C = T_d[1 \quad \frac{1}{\eta_C}(\pi_C^{frac\gamma-1\gamma} - 1)$ and $T_T = T_{CC}[1 - \eta_T(1 - \pi_T^{frac\gamma-1\gamma})$. Note that we assume $P_T = P_{noz}$. The output equation is given by [97]:

$$z = [P_C, T_C, N, P_T, T_T]$$

Remark 2.4. *Jet engines are instrumented with several thermocouples working in a wide range of temperatures. In particular, the highest temperature is the turbine temperature. This temperature can be instrumented in small engines. However, in larger engines turbine temperature can be over 1500* $\deg C$ *[232], which makes the use of conventional Nickle-based thermocouples impossible. The thermocouples currently used in jet engines have an operating temperature limitation of about* $1000 \deg C$ *and, as a result the turbine temperature is not usually measurable. The requirement for a higher temperature capability has been reported by gas turbine manufacturers [233]. An ongoing research (*HEATTOP *project) at university of Cambridge supported by leading gas turbine manufacturers such as Rolls-Royce and Siemens is working on development of very high temperature thermocouples for the use in gas turbines [234], [235]. As of September 2013, a ˇrst prototype is manufactured and tested at* $1300 \deg C$ *[236].*

In our research, we deˇne two different scenarios to perform FDI task. First, *we assume that the turbine temperature is* not measurable *and consequently cannot be used for FDI application.* Second, *assuming the success of HEATTOP project, we consider the turbine temperature to be* measurable *and usable for FDI application. We then study how this achievement may improve the performance of FDI task.*

Control Input

The mass flow rate of the main fuel, $\dot{m}_f$, is considered as a mechanism to control the engine parameters. The fuel mass flow rate is a function of the power level angle (PLA) adjusted by the pilot. The mass flow rate dynamics is determined as follows:

$$\tau\frac{\dot{m}_f}{dt} \quad \dot{m}_f = Gu_{fd} \tag{2.34}$$

where τ is the time constant, G is the gain associated with the fuel valve and u_{fd} denotes the fuel demand, which is determined by a rotational speed feedback [97].

2.3.3 Faults in the Jet Engine

By an abrupt fault one means any rapid reduction in any of the engine performance parameters such as compressor efficiency. Engine degeneration is a gradual reduction in the engine performance during its operation. An example of this kind of fault is the engine degradation due to fouling or erosion. Like any other system a jet engine is prone to three different types of failures 1- actuator faults, 2- component faults and 3- sensor faults as shown in Figure 2.14. Each of these fault types are explained below.

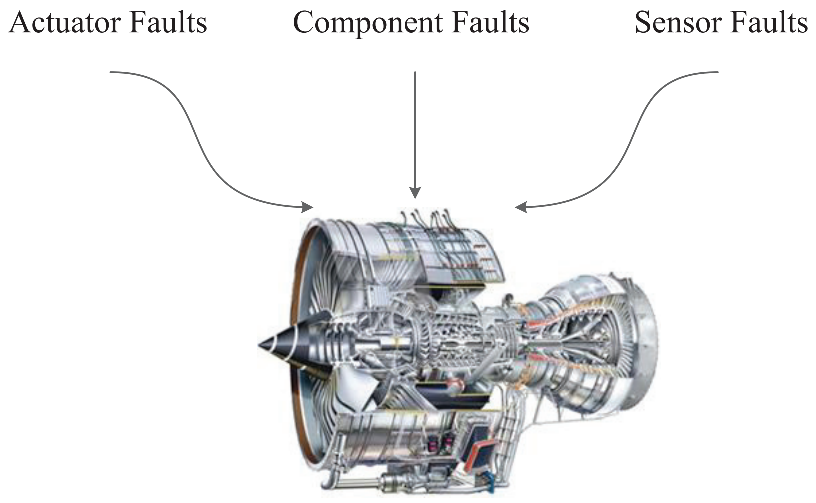


Figure 2.14: Jet engine fault types [237].

- Actuator fault: actuator fault can be defined as any reduction in the actual capability of engine actuators. In other words, an actuator fault is interpreted as any failure which affects the effectiveness of the control input. An example of actuator fault is the fuel valves failure to open or close correctly which leads to loss of effectiveness in delivery of the fuel.

- Component faults: component faults affect the healthy performance of the engine component. Fouling and erosion are two examples of component faults. Fouling is caused by accumulation of small particles on the turbine or the compressor blades contributing to reduction of the blades cross sections and finally overall reduction in the flow capacity. Fouling is caused by adherence of microparticles between $2 - 10\mu m$ which affects the smoothness of surfaces. Fouling results in the changes of aerofoils shape and it may happen in both compressor and turbine. An indication of fouling in compressor (respectively turbine) is reduction in compressor (respectively turbine) mass flow rate [229]. Thus, compressor and turbine mass flow rates can be considered as engine health parameters as an indication of fouling. Typically, fouling results in 5% decrease in mass flow rate, and up to 13% in the engine's output power. Figure 2.16 shows a fouled compressor.

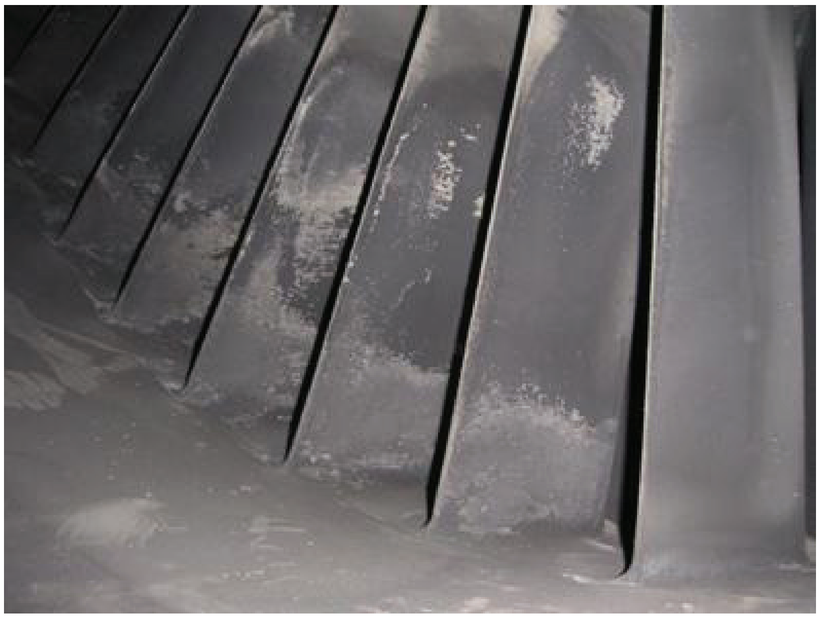


Figure 2.15: Fouled compressor [230].

Erosion is caused by collision of particles larger than $10\mu m$ with the compressor and the turbine blades. The particles remove the material in the gas flow path which leads to changes in aerofoil profiles. An indication of turbine (respectively compressor) erosion is decrease in efficiency of turbine (respectively compressor).

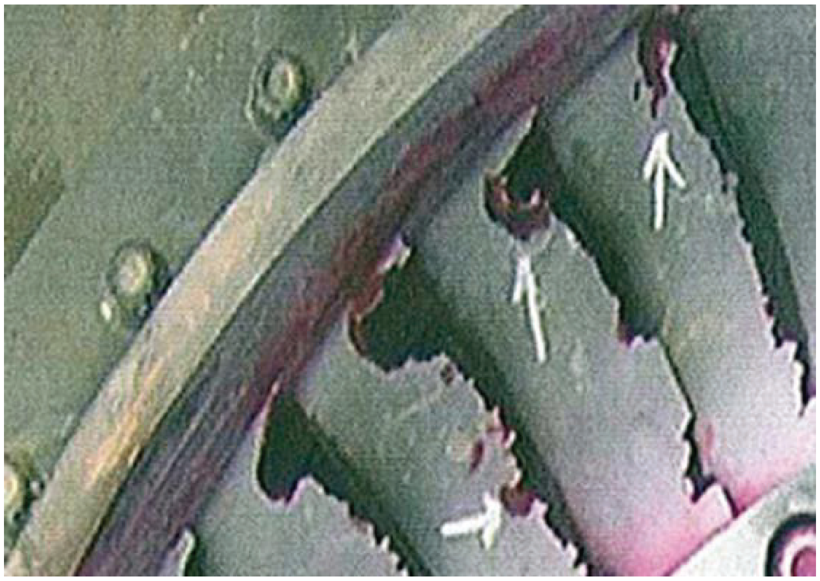


Figure 2.16: Erosion of turbine blades [230].

Foreign Object Damage (FOD) is another component fault caused by strike of relatively large objects. FOD fault usually results in decrease of efficiency and mass flow rate of turbine and compressor. FOD fault can result up to 5% decrease in compressor and turbine efficiency. Figure 2.17 shows damage caused by a foreign object entered to a jet engine.

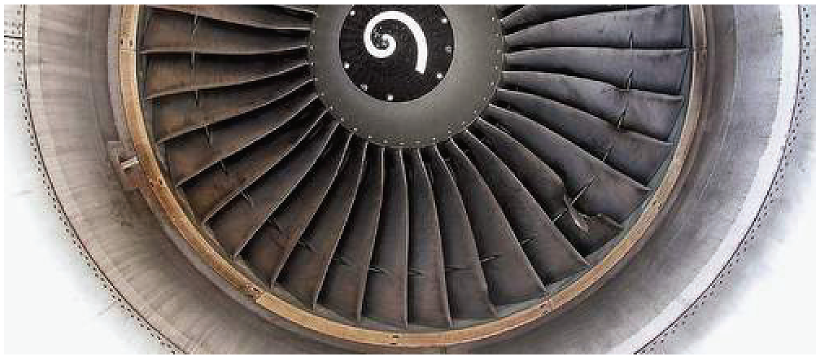


Figure 2.17: Turbine blade damage caused by foreign object [230].

- Sensor faults: sensor fault occurs when the output of a sensor is different from the actual values of the measured parameter (e.g. wrong temperature reading of a thermocouple is an example of sensor fault). A sensor fault may lead to poor regulation or tracking performance, or even affect the stability of the control system. Moreover, a faulty sensor output may also cause inaccurate diagnostics/prognostics, resulting in unnecessary replacement of system components or mission abortion. Therefore, it is important to correctly assess the health of on-board sensors [228]. Examples of jet engine sensor faults are compressor pressure sensor failure, compressor temperature sensor failure, turbine pressor sensor failure.

In this thesis, the main focus is on component faults. The engine health parameters considered in this research are compressor efficiency, compressor mass flow rate, turbine efficiency, and turbine mass flow rate. Thus, any change in any of the above mentioned engine parameters is an indication of a fault due to a degradation such as fouling or erosion.

Table 2.1: Jet engine component fault indications.

Component Fault	Description
F_{mc}	Decrease in the compressor flow capacity ($\dot{m}_C$)
F_{ec}	Decrease in the compressor efficiency (η_C)
F_{mt}	Decrease in the turbine flow capacity ($\dot{m}_C$)
F_{et}	Decrease in the turbine efficiency (η_C)

In order to model the engine's health status, we define the F_{mc}, F_{ec}, F_{mt}, and F_{et} as fault gains which are indicating compressor mass flow rate, compressor efficiency, turbine mass flow rate, and turbine efficiency respectively. For a healthy jet engine we consider the fault gains to be at 100% of their nominal values (i.e. $F_{mc} = F_{ec} = F_{mt} = F_{et} = 1$). Any degradation in an engine component results in decrease of one or more engine parameters, which causes the fault gains to have a value less than one. For typical component faults we consider a reduction up to 8% in engine health parameters (note that above 8% decrease in engine health parameters is considered to be so severe and consequently it can be detected and isolated relatively as easy compared to the less severe faults).

2.4 Summary

In this chapter a review of the required preliminaries for this research is presented. First the proven concept for ensemble learning was presented including bias-variance decomposition, the importance of diversity in ensemble learning and the metrics for measuring diversity in ensemble systems. Then in the second part the required preliminaries for the data-driven models used in this research were presented. Also, the required background for identification of dynamical systems were presented. Finally, the chapter reviewed required background information on jet engine including jet engine structure and its mathematical modeling as well as possible faults

Chapter 3

Ensemble Learning for Jet Engine Fault Detection

In previous chapter, we discussed ensemble learning as a powerful tool for solving both regression (e.g. system identification) and classification (e.g fault isolation) problems. In this chapter, we ˇ*rst* select three individual learning algorithm based on their characteristics and their popularity in literature for identification of jet engine dynamics. *Second*, we model the jet engine dynamics using the selected individual learning methods in order to *validate* them. *Third*, we build two different ensemble systems to model the jet engine dynamics. *Fourth*, designed ensemble systems are used to generate residual signals for fault detection (FD) of the jet engine. The performance of the ensemble-based fault detection system is compared with each individual methods selected from the literature.

As explained in Chapter 2 the diversity among individual ensemble members is essential. Clearly, there is no more information to be gained from a large set of identical individual learners.

Thus, the diversification approach plays an important role in the design process. The two general approaches for ensuring diversity which are used in this thesis are:

- Using different types of learning algorithms (e.g. different types of neural networks). In this first approach, the diversity among the learners is created by choosing different types of learners in general.
- Training learners using different training data. In this approach each learner is trained on a different subset of the training set. The diversity in this approach is created as each learner learns a part of the available data.

We then compare the proposed ensemble-based FD system with conventional (non-ensemble) methods in the literature from two perspectives. *First*, the accuracy of identified jet engine model, and *second*, accuracy of fault detection process. The individual learners which are trained in the first stage are used as benchmarks. These methods are used for modeling and FD of jet engine or gas turbine in the following references.

Model	Application	Reference
MLP-NARX	[24], [25], [26], [27], [28]	Jet engine modeling & FD
RBF-NARX	[24], [25]	Jet engine modeling & FD

Table 3.1: Jet engine fault detection (FD) methods selected from the literature as benchmarks.

3.1 Generating Engine Data

The required engine data can either be collected directly from a jet engine (if it is available) or with the help of a simulation model that is as realistic as possible. The later possibility is of special interest for collecting data on the different faulty situations, since generally those data may not be available for the real jet engine.

In this thesis, the data is generated using a Simulink model of a single-spool jet engine. For fault detection purpose, we collect the data from a healthy jet engine model. This data is used to identify dynamics of a healthy engine, which will be later used for generating residual signals. The data is collected while engine is operating in cruise mode. This corresponds to the PLA signal (set by the pilot) between $50°$ and $60°$ ($PLA > [50°, 60°]$) [200]. The relation between PLA set by the pilot and the engine fuel flow rate ($\dot{m}_f$) is described by the following equation [200].

$$\dot{m}_f = \begin{cases} \frac{PLA \times \dot{m}_f^{\max}}{70} & \text{if } PLA \quad 70° \\ \dot{m}_f^{\max} & \text{if } PLA \quad 70° \end{cases}$$

so according to the above the fuel flow rate which corresponds to the cruise mode would be approximately between 70% to 85% of the maximum fuel flow rate or in other words:

$$\frac{\dot{m}_f^{cruise}}{\dot{m}_f^{max}} > (0.7, 0.85)$$

To have a realistic scenario for collecting the training data we assume that the jet engine is operating in the cruise mode ($\frac{\dot{m}_f^{cruise}}{\dot{m}_f^{max}} > (0.7, 0.85)$) for *an hour (3600 sec)*. The PLA and thus the fuel flow rate makes slight random changes during the flight every *5 minutes (300 sec)*. This results in an input profile as shown in Figure 3.1.

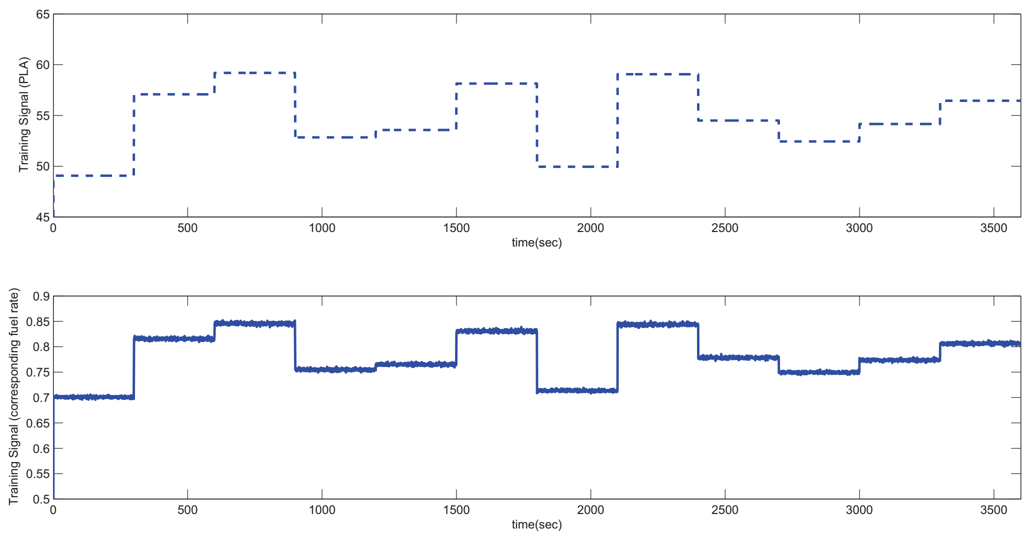


Figure 3.1: Top: PLA used for generating engine data in the cruise mode. Bottom: engine fuel flow rate $\dot{m}_f$ in the cruise mode used for generating engine data.

The generated training data contains the measured variables along the engine's gas path, as well as engine's fuel flow rate. A summary of the generated data is presented in Table 3.2.

Table 3.2: Training data generation summary.

Flight duration	3600 sec
Flight mode	cruise
Fuel rate	$\frac{\dot{m}_f^{cruise}}{\dot{m}_f^{max}} > (0.7, 0.85)$, $PLA > [50°, 60°]$ [200]
Instrumented parameters	P_C, compressor pressure P_T, turbine pressure T_C, compressor pressure T_T, turbine temperature (see Remark 2.4) N, rotational speed
Actuator parameter	$\dot{m}_f$ fuel flow rate (see Remark 3.1)
Total # of samples	3601

Remark 3.1. *Note that the actuator signal is set by the ight controller, and is not usually instrumented by a sensor. For a jet engine the actuator signal is the fuel ow rate and it is calculated based on the PLA set by the pilot.*

In order to have a realistic engine data, we add measurement noise to the collected engine data parameters. The percentage of noise applied to each of the engine parameters is shown in Table 3.3.

Table 3.3: Measurement noise standard deviations as percentage of engine parameter values at cruise condition[199].

P_C	T_C	N	P_T	T_T	$\dot{m}_f$
0.164	0.23	0.051	0.164	0.097	0.51

We observed that normalization of the data improves the performance of the learners. Thus, the following min-max normalization function is applied for preprocessing of the data, that is

$$X_n = 2 \quad \frac{X_{max} - X}{X_{max} - X_{min}}$$

The generated data is divided into training, testing, and cross-validation data sets in the next section, and are used for training and validating the jet engine model.

3.1.1 Training, Testing, and Cross-Validation Data Sets

The generated engine data is divided into three data sets: training data set, testing data set, and validation data set. The training data set, and cross-validation date set are used during training phase. The training data set is presented to the models during training phase. Since the model only sees the training data during training phase there is always a chance of *overˇtting*. This means that the model memorizes the training data, while it looses generalization capability to unseen data. In order to prevent the model from overfitting, we need to have a cross-validation set. The cross-validation data set is a subset of the training data which is not used for the training but instead is used for validation of the trained model during the training. The training stops, if the model's performance on cross-validation data set does not improve for i iterations in a row. Partitioning of the available data into training, test, and cross-validation data sets can affect the generalization performance of the model. We conduct the following experiment to determine an optimal partition of the available data into training, testing, and cross-validation data sets. We first consider partitioning the available data into training and testing data sets with different sizes which are 1- training data set = 40% of available data, testing data set = 60% of available data 2- training data set = 50% of available data, testing data set = 50% of available data 3- training data set = 60% of available data, testing data set = 40% of available data. Then for each case we use a part of the training data as cross-validation data set (in different experiments we use 20%, 30%, 40%, 50%, and 60% of the training data as cross-validation data set). In each case, the generalization error of the trained model is calculated on the testing data set. The results are reflected in Figures 3.2 to 3.6. Please note that the figures show the results for an MLP-NARX model where the network parameters (e.g. number of neurons are fixed to the optimal values obtained in the following). Ideally, we would like to repeat the above mentioned procedure for all the models and with various network parameters; however, in the remainder of this chapter, and due to the computational complexity we first partition the available data into training, and testing, and then based on the outcome of this experiment we use 40% of the training data set for cross-validation.

In order to evaluate the successfulness of cross-validation we compare the generalization error for different sizes of validation data with the case where no cross-validation is performed. We set the maximum number of iterations to a relatively big number (i.e. 100 iterations) noting the cross-validations stops the training in less than 10 iterations. It should be also noted that if we increase the maximum number of iterations to bigger numbers, then the generalization error would become even worse due to overfitting of the training data. Tables 3.5 to 3.8 compare the generalization error with and without cross-validation. One can see that early stopping with cross-validation reduces the generalization error significantly by preventing overfitting of training data. According to the experimental results (see Tables 3.5 to 3.8) one can see that using 40% of the available training data for cross-validation purpose results in a better generalization performance. Thus, in the remainder of this chapter we select 40% *of the training data set* for performing cross-validation.

Table 3.4: The effectiveness of cross-validation for identification of compressor pressure.

size of validation set (% of training data)	Generalization error	# of iteration	Stopping criteria
–	17.3162	100	max iteration
20%	6.3513	12	validation stop
40%	0.0330	9	validation stop
60%	0.1283	5	validation stop
80%	1.1678	5	validation stop

Table 3.5: The effectiveness of cross-validation for identification of compressor temperature.

size of validation set (% of training data)	Generalization error	# of iteration	Stopping criteria
–	19.2212	100	max iteration
20%	1.7559	5	validation stop
40%	2.2323	6	validation stop
60%	5.7995	6	validation stop
80%	2.0269	7	validation stop

Table 3.6: The effectiveness of cross-validation for identification of rotational speed.

size of validation set (% of training data)	Generalization error	# of iteration	Stopping criteria
–	97.3162	100	max iteration
20%	35.8760	20	validation stop
40%	31.8124	19	validation stop
60%	36.0391	23	validation stop
80%	39.5571	21	validation stop

Table 3.7: The effectiveness of cross-validation for identification of turbine temperature.

size of validation set (% of training data)	Generalization error	# of iteration	Stopping criteria
–	106.8616	100	max iteration
20%	39.1656	7	validation stop
40%	34.4623	16	validation stop
60%	37.4369	7	validation stop
80%	43.1970	6	validation stop

Table 3.8: The effectiveness of cross-validation for identification of turbine pressure.

size of validation set (% of training data)	Generalization error	# of iteration	Stopping criteria
–	9.1278	100	max iteration
20%	0.1162	7	validation stop
40%	0.0514	6	validation stop
60%	0.7248	6	validation stop
80%	0.2229	8	validation stop

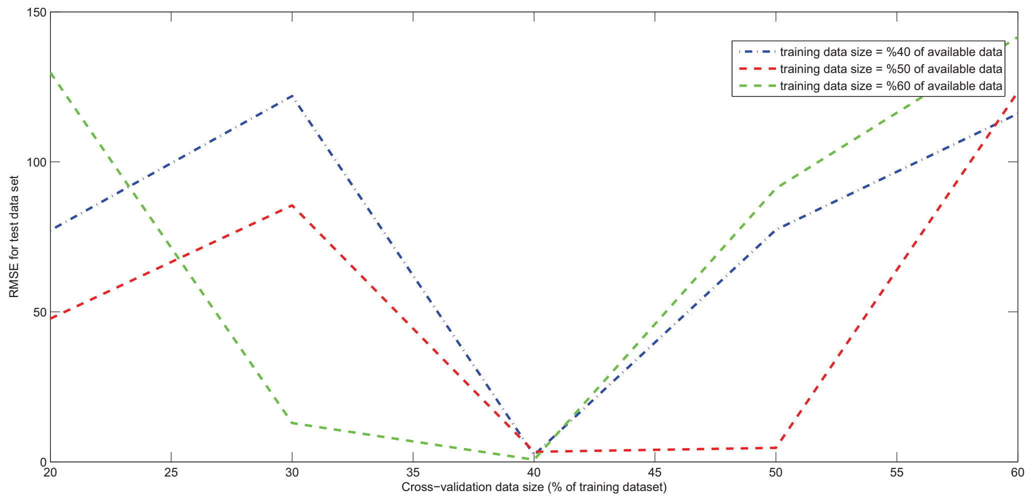


Figure 3.2: Compressor temperature estimation error vs. size of cross-validation data for different sizes of training data.

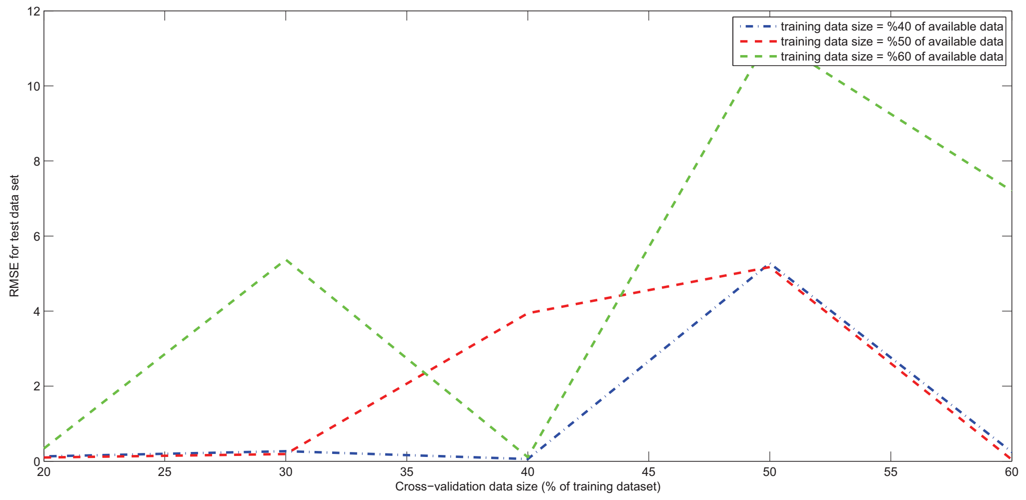


Figure 3.3: Compressor pressure estimation error vs. size of cross-validation data for different sizes of training data.

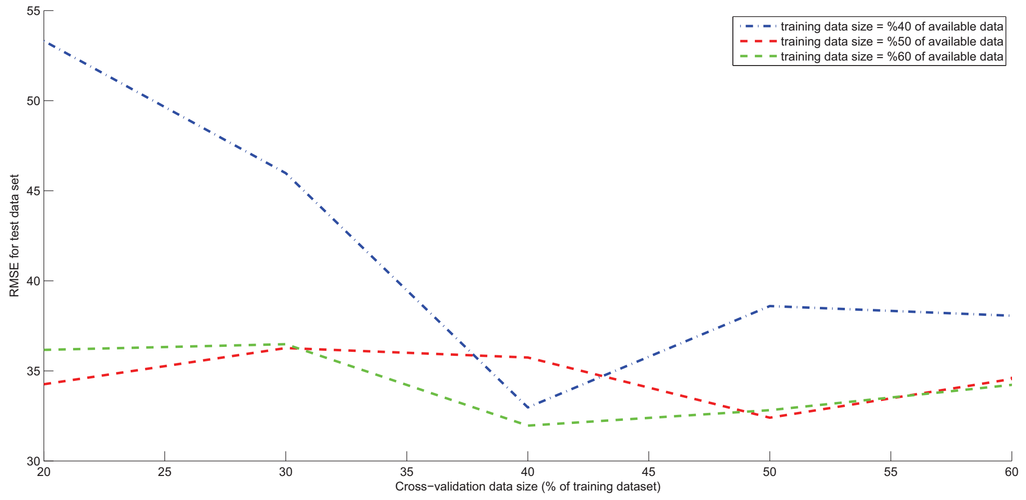


Figure 3.4: Rotational speed estimation error vs. size of cross-validation data for different sizes of training data.

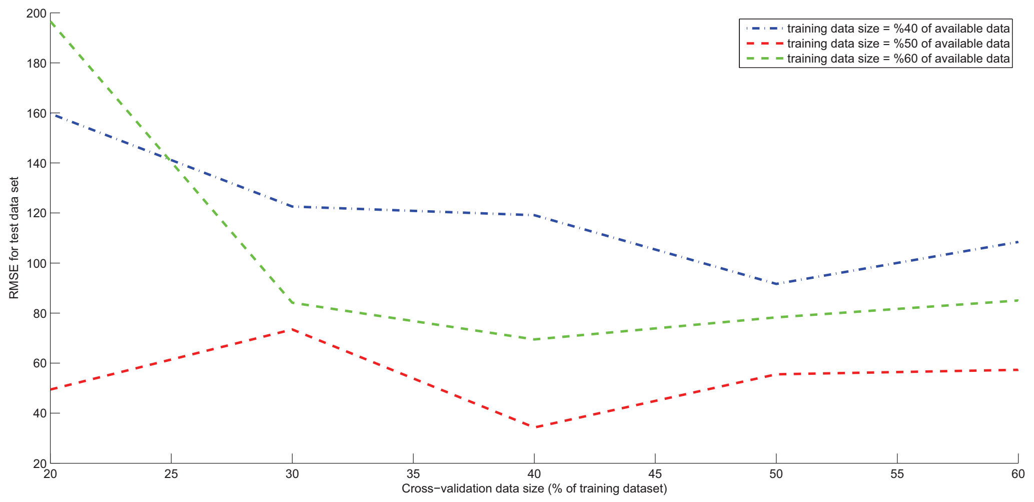


Figure 3.5: Turbine temperature estimation error vs. size of cross-validation data for different sizes of training data.

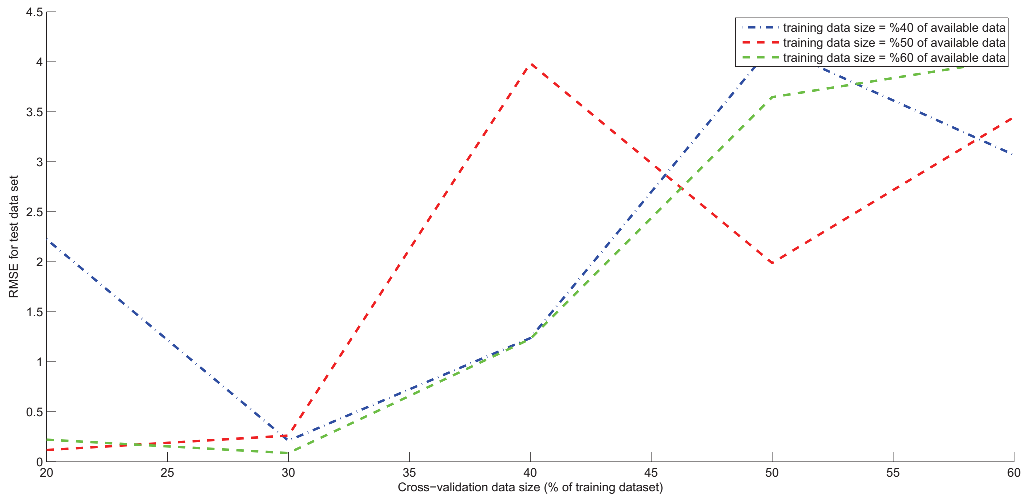


Figure 3.6: Turbine pressure estimation error vs. size of cross-validation data for different sizes of training data.

3.2 Neural Network Construction

In summary, the neural networks are constructed in two steps. First, the available engine data is partitioned into training, validation and testing data and the optimal size of the data sets is determined by experimenting different scenarios (i.e. different sizes for training, validation and testing data sets). Second, the partitioned data is used to construct the neural network while adjusting different network parameters (i.e. # of neurons, delays). Twelve different scenarios are considered for partitioning of the available data as described in the following:

- **Scenarios 1 - 4:** In these scenarios, first the available data is partitioned into 40% training data and 60% testing data. Then validation data size is selected as a percentage of the training data as follows.
  - **Scenario 1:** 40% of the available engine data is selected as training data and 60% of the engine data is selected as testing data with 20% of the training data is selected as validation data set.
  - **Scenario 2:** 40% of the available engine data is selected as training data and 60% of the engine data is selected as testing data with 40% of the training data is selected as validation data set.
  - **Scenario 3:** 40% of the available engine data is selected as training data and 60% of the engine data is selected as testing data with 60% of the training data is selected as validation data set.
  - **Scenario 4:** 40% of the available engine data is selected as training data and 60% of the engine data is selected as testing data with 80% of the training data is selected as validation data set.

Figure 3.7 shows how the engine data is divided for the scenarios 1-4.

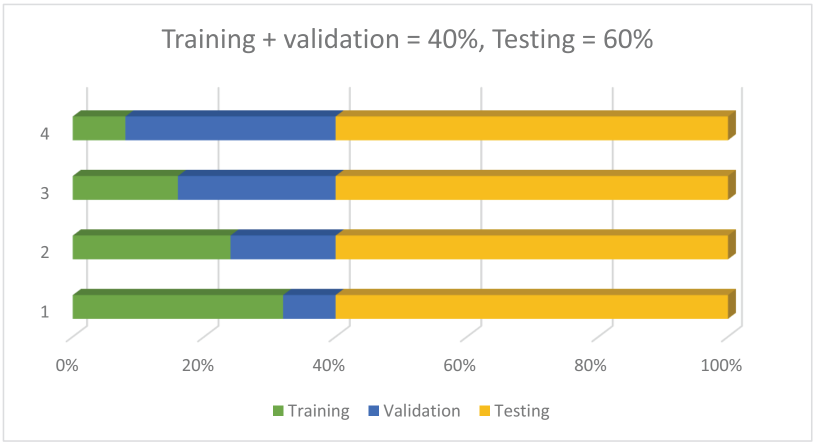


Figure 3.7: Partitioning of the available data into training, testing and validation data sets. Training data size = 60% of available data. Testing data size = 40% of the training data. Validation data size selected as a percentage of the training data.

- **Scenarios 5 - 8:** In these scenarios, first the available data is partitioned into 40% of training data and 60% testing data. Then validation data size is selected as a percentage of the training data as follows.
  - **Scenario 5:** 50% of the available engine data is selected as training data and 50% of the engine data is selected as testing data with 20% of the training data is selected as validation data set.
  - **Scenario 6:** 50% of the available engine data is selected as training data and 50% of the engine data is selected as testing data with 40% of the training data is selected as validation data set.
  - **Scenario 7:** 50% of the available engine data is selected as training data and 50% of the engine data is selected as testing data with 60% of the training data is selected as

validation data set.

- **Scenario 8:** 50% of the available engine data is selected as training data and 50% of the engine data is selected as testing data with 80% of the training data is selected as validation data set.

Figure 3.8 shows how the engine data is divided for the scenarios 5-8.

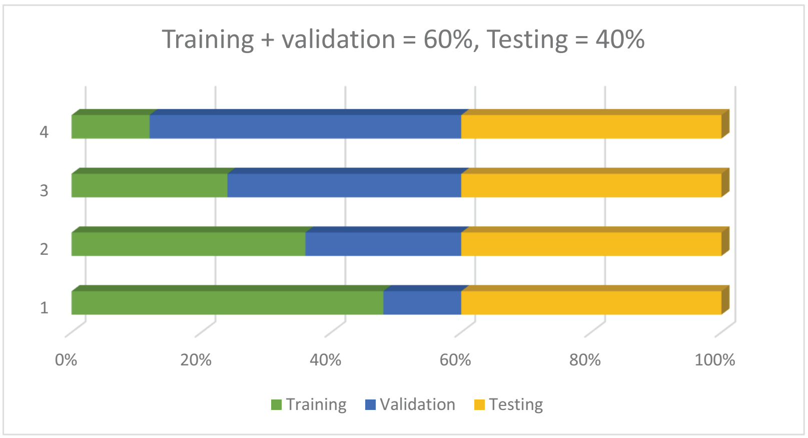


Figure 3.8: Partitioning of the available data into training, testing and validation data sets. Training data size = 50% of available data. Testing data size = 50% of the training data. Validation data size selected as a percentage of the training data.

- **Scenarios 9 - 12:** In these scenarios, first the available data is partitioned into 60% of training data and 40% testing data size. Then validation data size is selected as a percentage of the training data as follows.

  - **Scenario 9:** 60% of the available engine data is selected as training data and 40% of the engine data is selected as testing data with 20% of the training data is selected as validation data set.

– **Scenario 10:** 60% of the available engine data is selected as training data and 40% of the engine data is selected as testing data with and 40% of the training data is selected as validation data set.

– **Scenario 11:** 60% of the available engine data is selected as training data and 40% of the engine data is selected as testing data with and 60% of the training data is selected as validation data set.

– **Scenario 12:** 60% of the available engine data is selected as training data and 40% of the engine data is selected as testing data with and 80% of the training data is selected as validation data set.

Figure 3.9 shows how the engine data is divided for the scenarios 9-12.

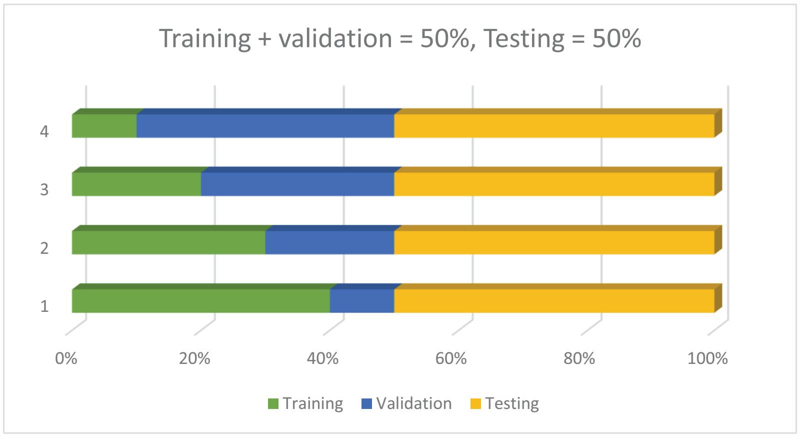


Figure 3.9: Partitioning of the available data into training, testing and validation data sets. Training data size = 60% of available data. Testing data size = 40% of the training data. Validation data size selected as a percentage of the training data.

Once the data is partitioned into training, validation and testing data sets (different scenarios are considered) then the networks are constructed and network parameters are adjusted to achieve the least generalization (i.e. testing error). Different model parameters (i.e. using different values for neurons/delays) are considered at this stage. Each model is trained using the training data. The validation data is used for stopping of the training in order to avoid over fitting of training data. The testing data is not exposed to the network during the training stage and it is only used to calculate the generalization error (i.e. $RMSE_{testing}$)

Remark 3.2. *It should be noted that once the data is partitioned (e.g. 30% for training, 20% for validation and 50% for testing) then the samples in each set are ˇxed and do not change during the model parameters selection stage. Thus, in each experiment the testing data is ˇxed and not exposed to the model at the training and validation stages. It should also be noted that the experiments for partitioning of the data are independent from each other. For example, if data is partitioned into 30% of training data, 20% of validation and 50% of the testing data then it means that 30%, 20% and 50% of the available data are* randomly *selected as training, validation and testing data sets respectively.*

3.3 Jet Engine Dynamics Identification

System identification plays an important role in fault detection algorithms. It is always required to have a reference model which generates the expected outputs of a healthy jet engine. The residual signals are then generated by comparing the outputs of the actual engine with the predictions of the reference model. In this thesis, different machine learning algorithms are trained to identify the dynamics of a single-spool jet engine. These models are later combined to design an ensemble system which estimates the normal behavior of a jet engine.

The jet engine dynamic is identified based on Nonlinear Autoregressive Exogenous model

(NARX) which is commonly used in system identification [217]. NARX relates the current value of an identified system to its previous values of inputs and outputs. Generally speaking, a system can be described as a function of its inputs and outputs. This can be summarized using the following equation:

$$y(k) = f(y(k-1), ..., y(k-d_y), u(k), ..., u(k-d_u)) \tag{3.1}$$

where u and y are the input and output vectors of the system respectively, and f is a nonlinear relation between the current value of the output $y(k)$ and the previous values of input and output vectors. When identifying a system using NARX model the goal is to find a (generally) nonlinear function $\hat{f}$ as follows:

$$\hat{y}(k) = \hat{f}(y(k-1), ..., y(k-\hat{d}_y), u(k), ..., u(k-\hat{d}_u)) \tag{3.2}$$

If we determine the time delays ($\hat{d}_y$) and ($\hat{d}_u$) then the generally nonlinear function $\hat{f}$ can be determined such that it identifies the system dynamics. According to [215] a proper approximation requires the order and time-delay of the approximated function to be equal or greater than the actual system's delays. In other words, we need to select $\hat{d}_y \quad d_y$ and $\hat{d}_u \quad d_u$. Two structures can be assumed for NARX model which are described in Chapter 2.

During training phase *series-parallel NARX model* is used for identification of jet engine dynamics. In this model actual outputs of the jet engine are fed back to the identification model rather than estimated outputs. This structure is shown in Figure 3.10.

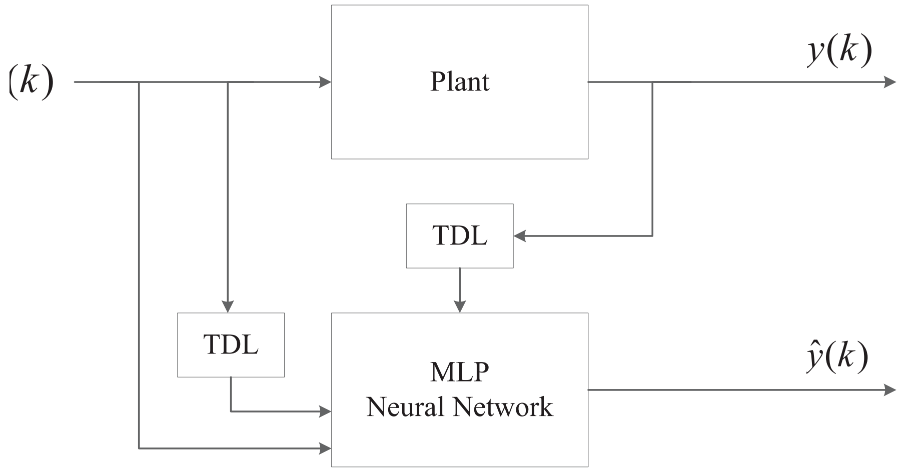


Figure 3.10: Series-parallel NARX model used for training stage [83].

Since the jet engine itself is BIBO stable, all signals which are used in identification process are bounded. This guarantees that the identified model of the engine is stable. Assuming that $\hat{y}(n) \approx y(n)$, the series-parallel model can be replaced by a parallel model during *recall phase* without serious consequences. The structure of the recall phase is shown in Figure 3.11.

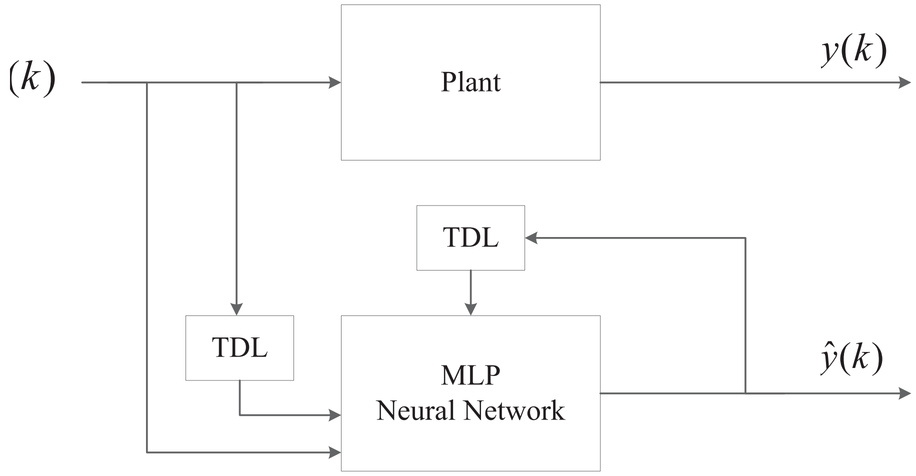


Figure 3.11: Parallel NARX model during recall phase [83].

Note that y represents engine parameters which can be P_T, T_T, P_C, TC, N.

3.4 Jet Engine Dynamics Identification using MLP-NARX

The use of MLP neural networks in NARX model for dynamical systems identification has been reported in several publications including but not limited to [17], [18], [19], [20], [21], [22], [23]. NARX model has proven ability in identification of a wide range of nonlinear systems [20], [23]. Feed-forward MLPs have been widely used in various applications. A common feature in all these applications is that MLPs are employed to realize some complex nonlinear functions. Theoretically, it is shown that even a single-layer perceptron can approximate any nonlinear function [216]. Thus, the theoretical foundation of nonlinear systems modeling using MLPs is already established.

Nonlinear Autoregressive Exogenous (NARX) model parameterizes any nonlinear dynamics as a (nonlinear) function of a regressor vector which contains current value of the system's input, as well as, the past values of inputs and outputs. In other words, NARX model describes the dynamics of a system using the following equation:

$$y(n) = f(y(n-1), ..., y(n-d_y), u(n), ..., u(n-d_u))$$

The nonlinear function f in the above equation is some nonlinear function. In this section MLP is used for approximation of the function f with application to jet engine system identification. Two different architectures are previously presented for NARX model in Chapter 2 and Section 3.2. The *series-parallel NARX model* uses the actual outputs of the plant are fed back to the identification model during training phase. There are two main advantages associated with this structure [83]. First, all signal which are used in the identification process (which are inputs of the regressor f) are bounded. This guarantees that the identified model would be also stable. Also, assuming that the identified model is close enough to the actual system, the series-parallel model can be replaced by a parallel model during *testing phase* without serious consequences. Thus, series-parallel structure is used during training phase of neural networks. The series-parallel NARX structure used for identification of jet engine dynamics is shown in Figure 3.12.

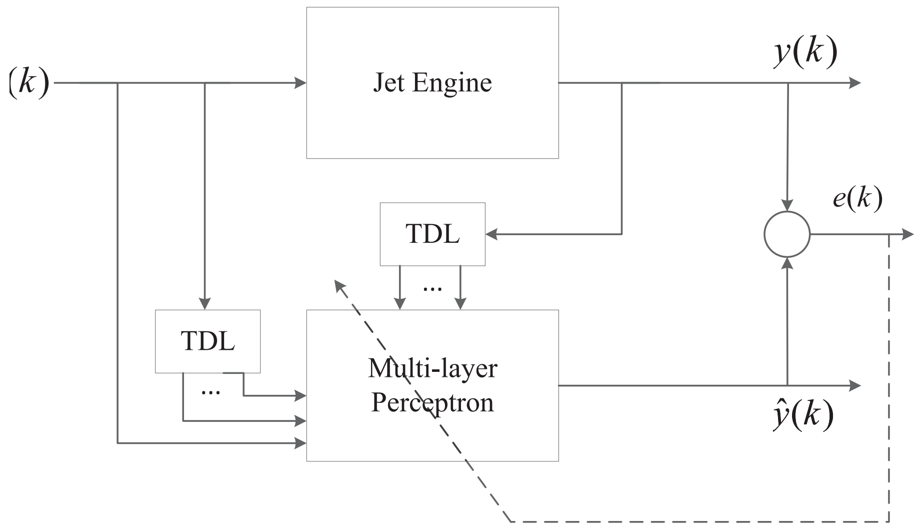


Figure 3.12: Schematic of MLP-NARX during training phase.

Note that in Figure 3.12, y can be any of the engine outputs (P_T, T_T, P_C, T_C, N) and u is the fuel flow rate $(\dot{m}_f)$. We train a separate neural network for each of the engine outputs. Figure 3.13 shows the system architecture during training. In this architecture each engine output is identified using a separate model. A series-parallel MLP-NARX model is used to model each engine output. The inputs to each network are the vectors $[\dot{m}_f(n), ..., \dot{m}_f(n - d_u)]$ and $[y(n-1), ..., y(n-d_y)]$ where y can be any of the engine outputs which are P_T, T_T, P_C, T_C and , N.

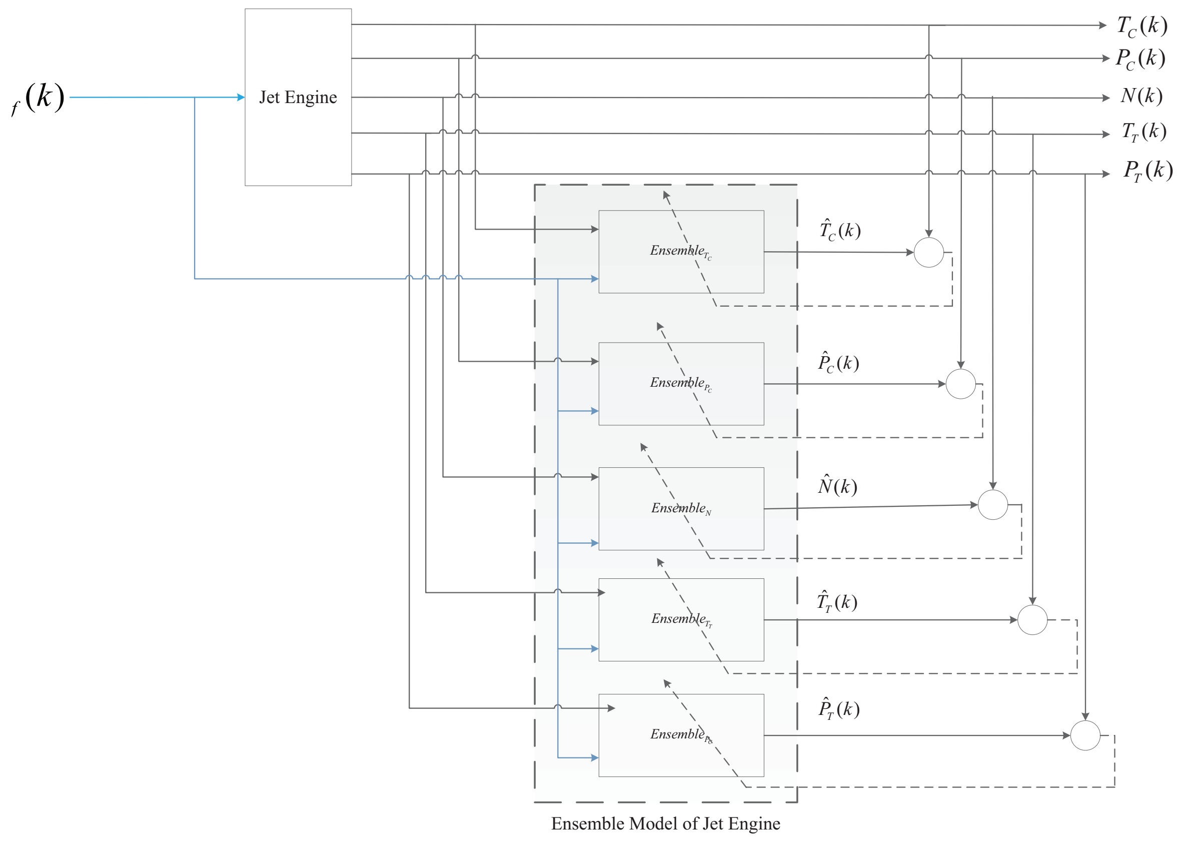


Figure 3.13: The architecture of MLP-NARX model of the jet engine during training phase.

Before training the neural networks we have several parameters to adjust for each networks structure. These parameters include the number of hidden layers, number of neurons, number of delays (d_u,d_y), and size of training set. Generally speaking, there is no rule to determine the optimal values of the above mentioned parameters for a specific application. Thus, we follow a constructive algorithm in order to achieve the desired performance. According to the approximation theorem [216] *any* nonlinear function can be approximated using a single-layer perceptron. Hence for simplicity we limit the number of hidden layers to one. We should note that according to [215] using only a single hidden layer may result in larger number of hidden neurons.

We start with a relatively small structure for the neural networks (one hidden layer, two hidden neurons, and the number of delays equal to two). *For avoiding too complex networks* we limit the number of hidden neurons to 20, and the number of delays to 10 ($d_u \quad 10, d_y \quad 10$). We use *Root Mean Squared Error (RMSE)* in order to evaluate the training, and generalization performance of the trained networks as follows:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} (P_i - T_i)^2}{n}}$$

where P_i and T_i are the network prediction and the target for the i^{th} sample, respectively and n is the total number of samples. We also consider the *Mean of Absolute Error (μ_{ae})* and *Standard Deviation of Absolute Error (σ_{ae})*, as we expect the error to be randomly distributed around zero for an appropriately constructed neural network. In the following, a summary of the construction of networks for each of the engine outputs is presented as follows:

$$\begin{aligned}
\mu_{ae} &= std(\; P_i - T_i \;) \\
\sigma_{ae} &= \frac{\sum_{i=1}^{n} \; P_i - T_i}{n}
\end{aligned}$$

Remark 3.3. *For system identiˇcation purposes estimating the model order (d_u and d_y in the NARX model) is specially critical. Generally speaking, the correct order of the model is not known a priori, and it is usually determined by constructing networks with several different model orders. In this construction, it is common to start with relatively small model orders, and then increasing* *order until the computed RMSE meets our design criteria. Thus, choosing* large enough *model order is a key. Assuming that the system dynamics is governed by the following difference equation, the appropriate delays must be selected such that* $d_u \quad N$*, and* $d_y \quad M$ *[7].*

$$y(n) = f\ \ (y(n-1), ..., y(n-M)) \quad\ g\ \ u(n), ..., u(n-N)$$

To reduce the number of trained networks, and with a mild assumption we consider $d_u = d_y = d$*. Then an appropriate delay,* d*, should be chosen such that* $d \quad \max\ \ M, N$

Remark 3.4. *As previously stated series-parallel NARX model is used during the training stage as shown in Figure 3.12. However, during the testing stage the series-parallel architecture is replaced by parallel architecture as shown in Figure 3.14.*

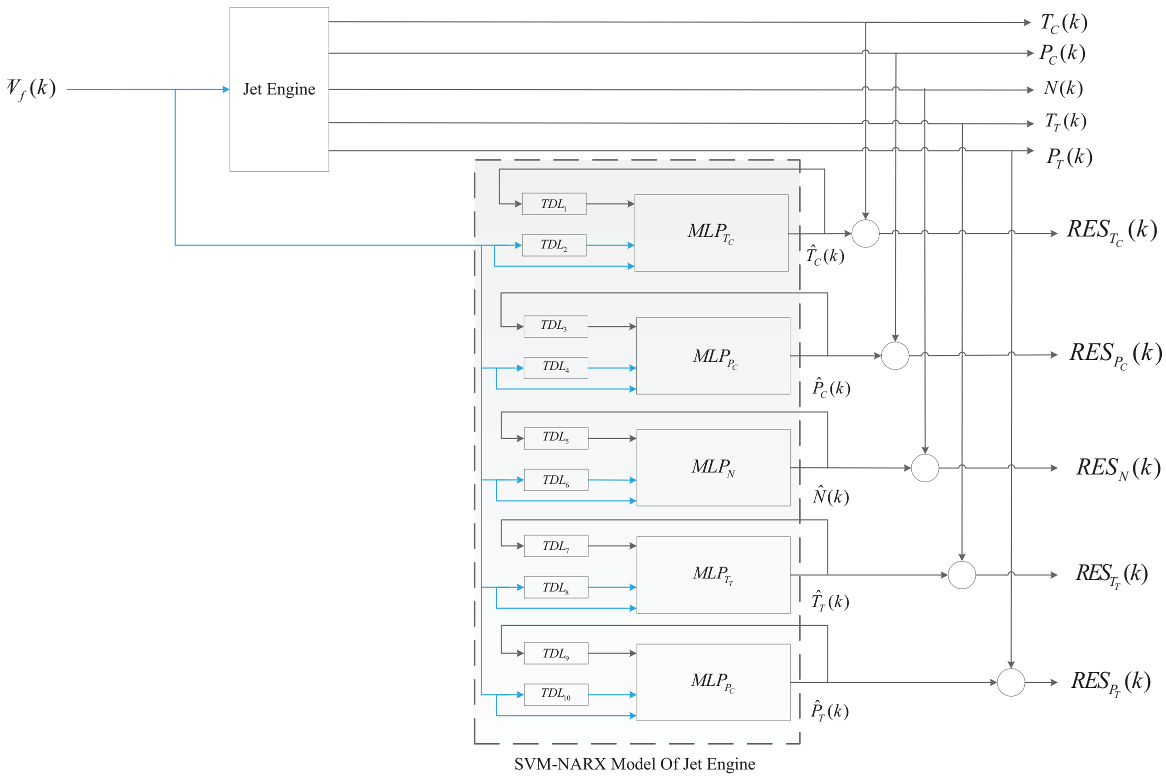


Figure 3.14: The architecture of the MLP-NARX model of jet engine during testing phase.

3.4.1 MLP-NARX Model of the Compressor Temperature

This section summarizes the construction of MLP-NARX model for identifying the dynamics of the compressor temperature. Three different sizes are selected for training data set. First, several neural networks are constructed using 40% of available data (1200 out of 3001 samples). Second, we construct several networks using 50% of available data (1501 out of 3001 samples), and finally, we construct the MLP-NARX neural networks using 60% (1801 out of 3001 samples).

In each case, we start with a relatively small structure for the neural networks (one hidden layer, five hidden neurons, and the number of delays equal to two). *For avoiding too complex networks* we limit the number of hidden neurons to 20 and the number delays to 10 ($d_u \quad 10, d_y \quad 10$). Also, as previously discussed in Remark 3.3, we assume $d_y = d_u$ in order to limit the number of construction trials. The architectures for training of the neural networks are shown in Figure 3.12. The network is trained using backpropagation as previously described in Chapter 2.

The summary of the network constructions is shown in Table 3.9.

Table 3.9: Summary of construction of MLP-NARX for modelling the compressor temperature (see the appendix for extensive summary).

# neurons	**# delays**	**# training samples**	**RMSE**$^{T_C}_{total}$	**RMSE**$^{T_C}_{training}$	**RMSE**$^{T_C}_{test}$	**%RMSE**$^{T_C}_{total}$	**mean** (μ_{ae})	**Std** (σ_{ae})
14	7	1501	108.7845	109.3749	108.1905	16.4491	106.325	23.0051
14	8	1501	111.0706	109.9238	112.2064	16.8075	105.2197	35.5797
14	4	1801	1.3272	1.4965	1.0216	0.202	1.0032	0.86898
14	5	1801	5.1862	5.8258	4.0405	0.7897	3.9202	3.396
14	6	1801	63.8587	63.2542	64.7554	9.653	63.6005	5.7383
14	7	1801	4.0198	4.6657	2.7821	0.61273	3.0276	2.6448
14	8	1801	78.5176	78.3627	78.7496	11.88	78.4966	1.8173
15	4	1200	3.0191	4.0894	2.0113	0.46038	1.839	2.3948
15	5	1200	11.5477	12.252	11.0536	1.7562	9.3723	6.7472
15	6	1200	2.6182	3.3374	2.0003	0.39905	1.8458	1.8572
15	**7**	**1200**	**3.5106**	**5.5082**	**0.56647**	**0.53711**	**0.74164**	**3.432**
15	8	1200	2.0647	2.3486	1.8516	0.31423	1.6327	1.2641
15	4	1501	70.2922	70.4178	70.1664	10.6334	58.1478	39.5013
15	5	1501	44.7489	44.8696	44.6278	6.768	34.1267	28.9501
15	6	1501	2.0963	2.493	1.6039	0.31954	1.3973	1.5629
15	7	1501	4.1934	4.6624	3.6644	0.63987	2.8925	3.0366
15	8	1501	4.8707	5.5448	4.0862	0.74261	3.4636	3.425

Table 3.9: Summary of construction of MLP-NARX for modelling the compressor temperature (see the appendix for extensive summary).

# neurons	# delays	# training samples	RMSE$^{T_C}_{total}$	RMSE$^{T_C}_{training}$	RMSE$^{T_C}_{test}$	%RMSE$^{T_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
15	4	1801	2.128	2.4083	1.6187	0.32415	1.5784	1.4274
15	5	1801	3.4745	4.2668	1.6931	0.53056	1.7334	3.0117
15	6	1801	2.2914	2.8764	0.84487	0.34997	0.87841	2.1167
15	7	1801	5.401	5.8805	4.5882	0.82196	4.3454	3.2081

In summary the best performance is achieved by the network with the following parameters:

Table 3.10: Best MLP-NARX for modeling the compressor temperature in terms of $RMSE_{test}$.

# neurons	# delays	# training samples	RMSE$^{T_C}_{total}$	RMSE$^{T_C}_{training}$	RMSE$^{T_C}_{test}$	%RMSE$^{T_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
15	**7**	**1200**	**3.5106**	**5.5082**	**0.56647**	**0.53711**	**0.74164**	**3.432**

The engine output and the trained network output for both training and testing data are shown in Figure 3.15.

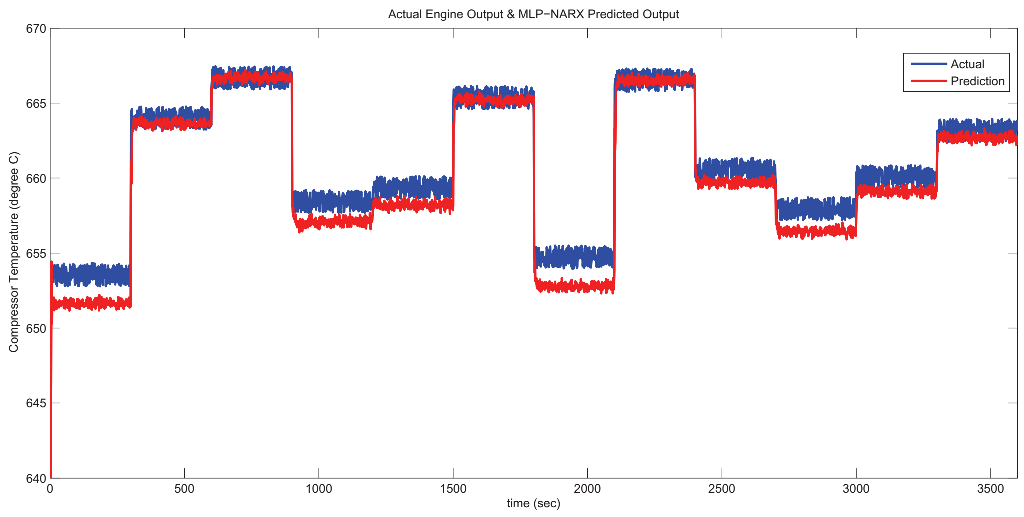


Figure 3.15: MLP-NARX model prediction and actual engine output (compressor temperature).

3.4.2 MLP-NARX Model of the Compressor Pressure

This section summarizes the extensive simulations which performed for training MLP-NARX model of the compressor pressure. Three different sizes are selected for the training data (40%, 50%, 60% of the available data). In each case, we start with a relatively small neural network structure, and then we construct more complex networks by adding neurons and delays. To avoid too complex networks we limit the number of hidden neurons to 20 and the number delays to 10 $(d_u \quad 10, d_y \quad 10)$. Also, as previously discussed in Remark 3.3, we assume $d_y = d_u$ in order to limit the number of trials. The architecture for training the neural networks is shown in Figure 3.12. The network is trained by using backpropagation as previously described in Chapter 2.

The summary of network construction is shown in Table 3.11.

Table 3.11: Summary of construction of MLP-NARX for modelling the compressor pressure (see the appendix for extensive summary).

# neurons	# delays	# training samples	RMSE$^{P_C}_{total}$	RMSE$^{P_C}_{training}$	RMSE$^{P_C}_{test}$	%RMSE$^{P_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
12	6	1200	4.8941	4.9334	4.8677	40.6131	4.7393	1.221
12	7	1200	0.06571	0.087446	0.045822	0.55726	0.038364	0.053357
12	8	1200	0.13281	0.18103	0.086906	1.1381	0.078402	0.10721
12	4	1501	7.2386	7.2594	7.2178	59.9556	7.2338	0.26471
12	5	1501	0.075964	0.095119	0.049915	0.64918	0.042239	0.063149
12	6	1501	0.17796	0.24437	0.060041	1.53	0.061056	0.16719
12	7	1501	7.7776	7.8118	7.7433	64.3566	7.7663	0.42043
12	8	1501	0.068131	0.083435	0.048173	0.57905	0.03969	0.055385
12	4	1801	0.30503	0.33921	0.24495	2.6813	0.24049	0.18766
12	5	1801	1.0244	1.0791	0.93632	8.6699	0.74842	0.69957
12	**6**	**1801**	**0.07754**	**0.095055**	**0.038409**	**0.70861**	**0.034802**	**0.069302**
12	7	1801	0.10034	0.12331	0.048562	0.85987	0.051821	0.08594
12	8	1801	0.26226	0.33004	0.092335	2.2502	0.15067	0.21469
13	4	1200	6.5354	6.4513	6.5909	53.8924	6.2991	1.7418
13	5	1200	0.12091	0.17572	0.061522	1.035	0.05951	0.10526
13	6	1200	0.13152	0.15096	0.11679	1.1204	0.1015	0.083656
13	7	1200	9.1264	9.0765	9.1595	75.682	8.226	3.9534
13	8	1200	0.12625	0.1864	0.058403	1.2959	0.048202	0.11671
13	4	1501	0.11394	0.15435	0.046155	0.97906	0.042831	0.1056
13	5	1501	0.21434	0.21912	0.20944	1.8191	0.16612	0.13547
13	6	1501	0.34614	0.44288	0.20843	2.9776	0.18655	0.29163

In summary the best performance is achieved by the network with the following parameters:

Table 3.12: Best MLP-NARX for modeling of compressor pressure in terms of $RMSE_{test}$.

# neurons	# delays	# training samples	RMSE$^{P_C}_{total}$	RMSE$^{P_C}_{training}$	RMSE$^{P_C}_{test}$	%RMSE$^{P_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
12	**6**	**1801**	**0.07754**	**0.095055**	**0.038409**	**0.70861**	**0.034802**	**0.069302**

The engine output and the trained network output for both training and testing data are shown in Figure 3.16.

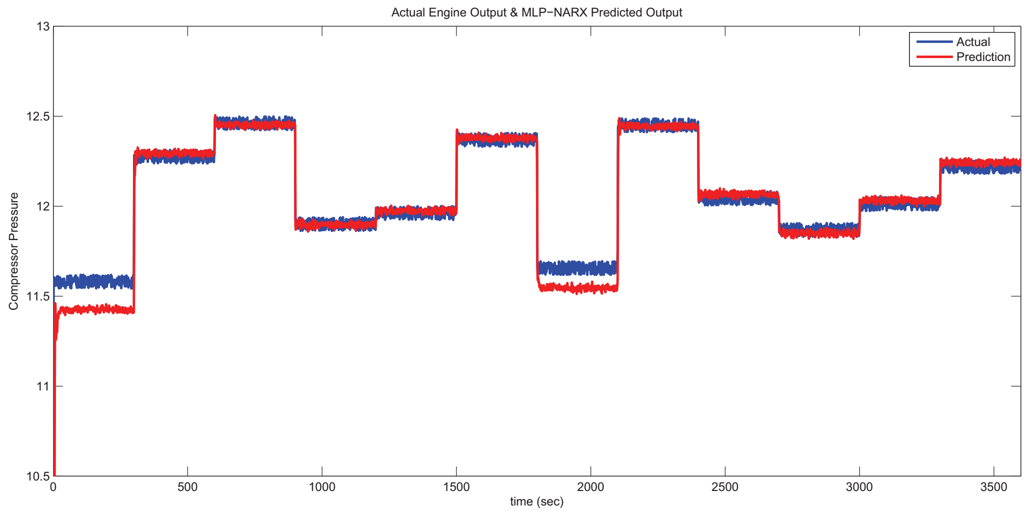


Figure 3.16: MLP-NARX model prediction and actual engine output (compressor pressure).

3.4.3 MLP-NARX Model of the Rotational Speed

This section summarizes the extensive simulations performed for modeling the engine rotational speed using MLP-NARX model. Like before, three different sizes of training data are selected (40%, 50%, 60% of the available data). We start with a relatively small neural network structure, and then construct more complex networks by adding neurons and delays. To avoid too complex networks we limit the number of hidden neurons to 20 and the number delays to 10 ($d_u \quad 10, d_y$

10). Also, as previously discussed in Remark 3.3, we assume $d_y = d_u$ in order to limit the number of trials. The architecture for training the neural network is shown in Figure 3.12.

The summary of the network construction is shown in Table 3.13.

Table 3.13: Summary of construction of MLP-NARX for modelling the rotational speed (see the appendix for extensive summary).

# neurons	# delays	# training samples	RMSE$^N_{total}$	RMSE$^N_{training}$	RMSE$^N_{test}$	%RMSE$^N_{total}$	mean (μ_{ae})	Std (σ_{ae})
10	5	1200	35.5009	37.8877	33.8172	0.29961	28.6901	20.9126
10	6	1200	36.9265	40.1687	34.598	0.31214	29.6183	22.0564
10	7	1200	296.6163	300.5436	293.9705	2.4957	294.5564	34.9023
10	8	1200	35.0706	36.4016	34.1549	0.29608	28.9322	19.8243
10	4	1501	450.7895	447.4808	454.0762	3.7936	449.116	38.8127
10	5	1501	36.5406	40.6434	31.9113	0.30842	29.8534	21.0745
10	6	1501	400.5267	398.5496	402.4953	3.3711	399.6232	26.892
10	7	1501	37.6383	42.1131	32.5504	0.31812	30.6131	21.9006
10	8	1501	46.1702	50.8953	40.899	0.38999	32.591	32.7088
10	4	1801	33.0561	35.247	29.4638	0.27903	27.0061	19.0656
10	**5**	**1801**	**27.2712**	**32.668**	**16.0695**	**0.23157**	**16.9852**	**21.3395**
10	6	1801	33.9831	35.9672	30.7663	0.28692	28.2217	18.9344
10	7	1801	33.5627	35.5145	30.3992	0.28338	27.7218	18.923
10	8	1801	43.4787	50.0322	31.1554	0.36597	30.8818	30.6108
11	4	1200	29.9508	32.6795	27.9854	0.25259	24.7664	16.8455
11	5	1200	37.26	38.6579	36.2987	0.31462	30.0035	22.0966
11	6	1200	487.0859	489.2236	485.6564	4.0997	486.0575	31.6409
11	7	1200	34.7142	42.2669	28.5952	0.29399	24.9275	24.1639
11	8	1200	32.9792	34.8175	31.6952	0.27836	27.0523	18.8659

Table 3.13: Summary of construction of MLP-NARX for modelling the rotational speed (see the appendix for extensive summary).

# neurons	**# delays**	**# training samples**	**RMSE$^N_{total}$**	**RMSE$^N_{training}$**	**RMSE$^N_{test}$**	**%RMSE$^N_{total}$**	**mean (μ_{ae})**	**Std (σ_{ae})**
11	4	1501	34.5558	38.073	30.6348	0.29169	28.7501	19.1745
11	5	1501	37.1712	41.6389	32.0838	0.31366	30.0774	21.8451

Table 3.13 shows network structures as well as validation results (e.g. training, testing, and total RMSE). The best performance (in terms of testing RMSE) is achieved by the network with the following parameters:

Table 3.14: Best MLP-NARX for modeling the rotational speed in terms of $RMSE_{test}$.

# neurons	**# delays**	**# training samples**	**RMSE$^N_{total}$**	**RMSE$^N_{training}$**	**RMSE$^N_{test}$**	**%RMSE$^N_{total}$**	**mean (μ_{ae})**	**Std (σ_{ae})**
10	**5**	**1801**	**27.2712**	**32.668**	**16.0695**	**0.23157**	**16.9852**	**21.3395**

The engine output and the trained network output for both training and testing data are shown in Figure 3.17.

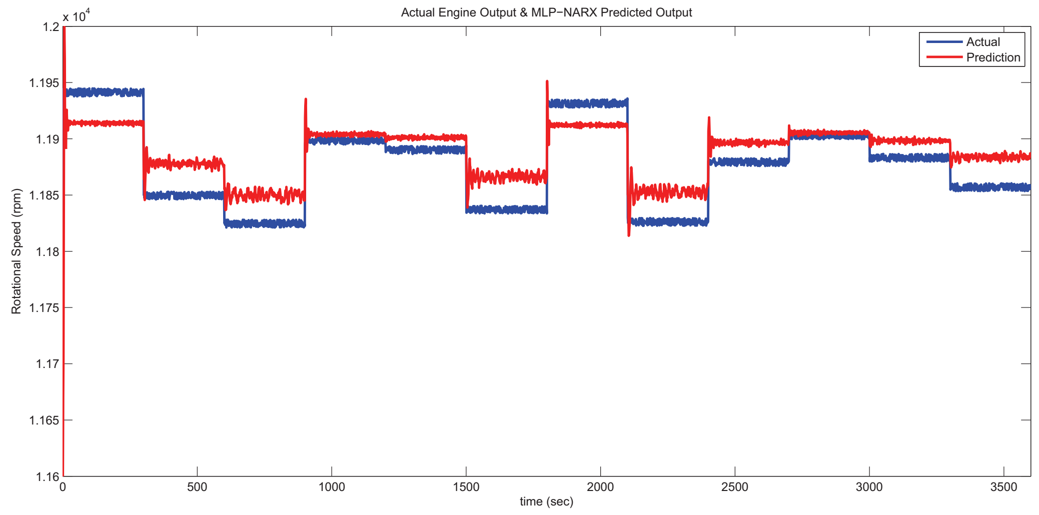


Figure 3.17: MLP-NARX model prediction and actual engine output (rotational speed).

3.4.4 MLP-NARX Model of the Turbine Temperature

This section summarizes the construction of the MLP-NARX model for identifying the dynamics of turbine temperature. Like before, three different sizes are selected for training data set (40%, 50%, 60% of the available data). The construction starts with small network and continues to more complex networks. In order to avoid too complex networks, we limit the number of hidden neurons to 20 and the number delays to 10 ($d_u \quad 10, d_y \quad 10$). Also, as previously discussed in Remark 3.3, we assume $d_y = d_u$ in order to limit the number of construction trials. The architecture for training the neural network is shown in Figure 3.12. The network is trained by using backpropagation as previously described in Chapter 2.

Table 3.15 summarises the construction and validation results of the trained MLP-NARX models for turbine temperature.

Table 3.15: Summary of construction of MLP-NARX for modelling the turbine temperature (see the appendix for extensive summary).

# neurons	# delays	# training samples	RMSE$^{T_T}_{total}$	RMSE$^{T_T}_{training}$	RMSE$^{T_T}_{test}$	%RMSE$^{T_T}_{total}$	mean (μ_{ae})	Std (σ_{ae})
12	5	1801	460.9805	454.8895	469.9741	24.6622	449.3605	102.8677
12	6	1801	768.8118	771.8882	764.1713	42.5231	709.318	296.5953
12	7	1801	155.7693	159.994	149.2043	8.8018	129.8104	86.1151
12	8	1801	159.2104	154.2648	166.3573	8.7893	150.7838	51.1184
13	4	1200	222.7684	227.2427	219.7365	12.5666	194.0663	109.3984
13	5	1200	1054.4226	1036.3916	1066.2673	57.1032	993.248	353.9877
13	6	1200	74.6131	73.1871	75.5483	4.1297	67.4487	31.908
13	7	1200	1616.5065	1609.1162	1621.4119	88.0136	1392.8149	820.6
13	8	1200	346.7517	336.1981	353.6087	18.9061	321.841	129.0761
13	4	1501	652.3556	655.5736	649.1194	35.4576	613.8091	220.9581
13	**5**	**1501**	**41.9615**	**40.4995**	**43.3752**	**2.317**	**37.1441**	**19.5246**
13	6	1501	650.2679	653.681	646.8345	34.8309	604.7047	239.1645
13	7	1501	1060.3711	1066.0517	1054.656	58.2532	1048.166	160.4476
13	8	1501	1048.6294	1058.7777	1038.3751	56.7266	1042.463	113.573
13	4	1801	1218.9509	1213.006	1227.8192	66.3182	983.1694	720.6888
13	5	1801	260.3611	258.2062	263.5621	14.482	241.341	97.7015
13	6	1801	243.6969	227.401	266.2894	13.4599	220.9837	102.7516
13	7	1801	878.8761	874.1466	885.927	48.1715	777.9259	409.0358
13	8	1801	239.4609	243.237	233.6791	13.1819	219.1003	96.6421
14	4	1200	130.0401	123.4004	134.282	7.1211	117.3991	55.9367
14	5	1200	173.9732	168.3939	177.5934	9.6026	158.1745	72.4519

In summary, the best performance (in terms of the testing RMSE) is archived by the following parameters.

Table 3.16: Best MLP-NARX for modeling the turbine temperature in terms of $RMSE_{test}$.

# neurons	# delays	# training samples	RMSE$^{T_T}_{total}$	RMSE$^{T_T}_{training}$	RMSE$^{T_T}_{test}$	%RMSE$^{T_T}_{total}$	mean (μ_{ae})	Std (σ_{ae})
13	**5**	**1501**	**41.9615**	**40.4995**	**43.3752**	**2.317**	**37.1441**	**19.5246**

The engine output and the trained network output for both training and testing data are shown in Figure 3.18.

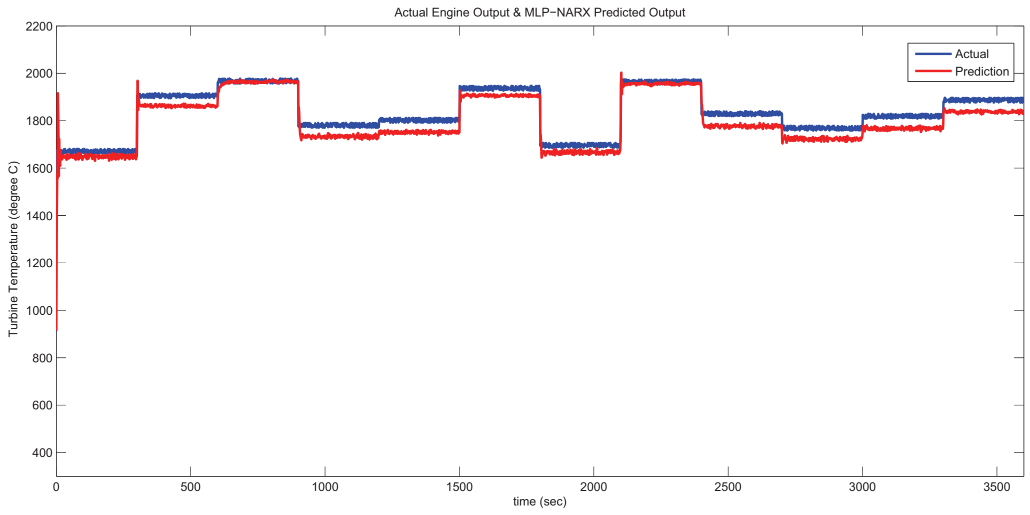


Figure 3.18: MLP-NARX model prediction and actual engine output (turbine temperature).

3.4.5 MLP-NARX Model of the Turbine Pressure

This section summarizes the construction of the MLP-NARX model for identifying the dynamics of turbine pressure. Like before, three different sizes are selected for training data set (40%, 50%, 60% of the available data). The construction starts with small network and continues to more complex networks. In order to avoid too complex networks, we limit the number of hidden neurons to 20 and the number delays to 10 ($d_u \quad 10, d_y \quad 10$). Also, as previously discussed in Remark 3.3, we assume $d_y = d_u$ in order to limit the number of construction trials. The architecture for training the neural network is shown in Figure 3.12. The network is trained by using backpropagation as previously described in Chapter 2.

Table 3.17 summarises the construction and validation results of trained MLP-NARX models for turbine temperature.

Table 3.17: Summary of construction of MLP-NARX for modelling the turbine pressure (see the appendix for extensive summary).

# neurons	# delays	# training samples	RMSE$^{P_T}_{total}$	RMSE$^{P_T}_{training}$	RMSE$^{P_T}_{test}$	%RMSE$^{P_T}_{total}$	Mean (μ_{ae})	Std(σ_{ae})
12	8	1200	1.5321	1.7145	1.3974	33.2073	1.1832	0.97346
12	4	1501	6.3269	6.5764	6.067	138.2268	3.6285	5.1839
12	5	1501	0.858	0.98273	0.71164	19.1013	0.67116	0.53461
12	6	1501	0.92128	1.2058	0.49312	20.2576	0.4038	0.82821
12	7	1501	16.0708	16.1816	15.9593	330.8824	16.0099	1.3979
12	8	1501	10.0099	10.1097	9.909	205.8049	9.9552	1.0452
12	4	1801	0.49299	0.44647	0.55554	10.3286	0.4192	0.25949
12	5	1801	0.33884	0.40479	0.20299	8.3306	0.20063	0.2731
12	6	1801	9.4812	9.0519	10.0912	198.5974	7.7466	5.4674
12	7	1801	0.38883	0.46908	0.21877	8.517	0.2126	0.32562
12	**8**	**1801**	**0.50515**	**0.64118**	**0.14539**	**11.2103**	**0.16013**	**0.47917**
13	4	1200	11.9467	11.7735	12.0607	244.2561	11.4902	3.2713
13	5	1200	30.3314	30.205	30.4153	626.7339	30.2972	1.4392
13	6	1200	0.89159	0.96697	0.83761	19.4837	0.73062	0.5111
13	7	1200	0.3794	0.51465	0.25175	8.2715	0.22773	0.3035
13	8	1200	4.3377	6.411	1.9915	95.4077	2.5099	3.5384

Table 3.17: Summary of construction of MLP-NARX for modelling the turbine pressure (see the appendix for extensive summary).

# neurons	**# delays**	**# training samples**	**RMSE**$^{P_T}_{total}$	**RMSE**$^{P_T}_{training}$	**RMSE**$^{P_T}_{test}$	**%RMSE**$^{P_T}_{total}$	**Mean** (μ_{ae})	**Std**(σ_{ae})
13	4	1501	0.87233	1.08	0.59608	22.0664	0.45508	0.74434
13	5	1501	34.6874	34.7931	34.5814	717.0989	34.6576	1.4377
13	6	1501	0.29812	0.34907	0.23639	6.4572	0.20653	0.21502
13	7	1501	9.6314	9.8002	9.4594	197.1338	9.506	1.5491
13	8	1501	0.66389	0.80207	0.48789	14.5958	0.39815	0.53133

In summary, the best performance (in terms of the testing RMSE) is archived by the following parameters.

Table 3.18: Best MLP-NARX for modeling the turbine pressure in terms of $RMSE_{test}$**.**

# neurons	**# delays**	**# training samples**	**RMSE**$^{P_T}_{total}$	**RMSE**$^{P_T}_{training}$	**RMSE**$^{P_T}_{test}$	**%RMSE**$^{P_T}_{total}$	**Mean** (μ_{ae})	**Std**(σ_{ae})
12	**8**	**1801**	**0.50515**	**0.64118**	**0.14539**	**11.2103**	**0.16013**	**0.47917**

The engine output and the trained network output for both training and testing data are shown in Figure 3.19.

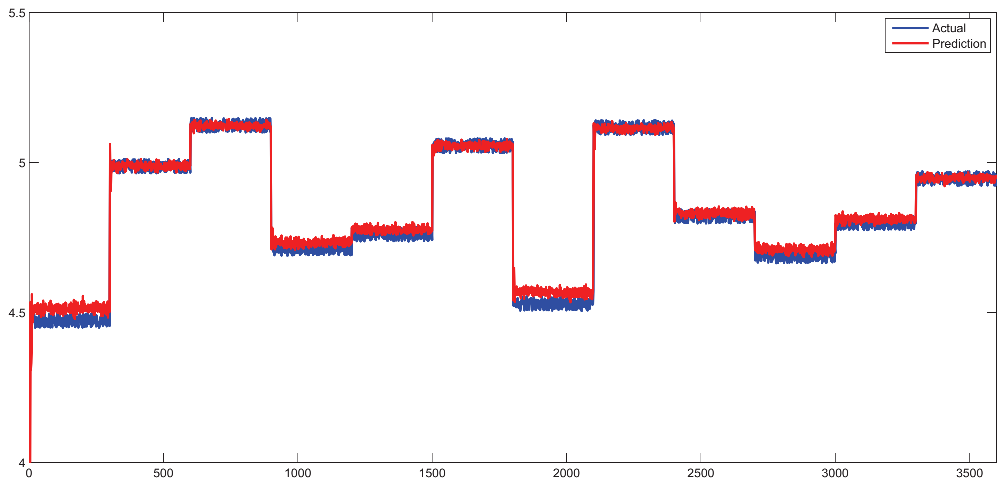


Figure 3.19: MLP-NARX model prediction and actual engine output (turbine pressure).

3.5 Jet Engine Dynamics Identification using RBF-NARX

The use of RBF neural networks in NARX model for dynamical systems identification has been reported in several publications including but not limited to [8], [10], [11], [12], [13], [14], [15], [16]. This section describes construction of the RBF-NARX model to identify jet engine dynamics.

As previously described in Chapter 2 a series-parallel architecture is used for training of RBF-NARX model as shown in Figure 3.21. In this architecture the engine output is fed back into the RBF-NARX model, assuming that the engine is fault-free during the training stage.

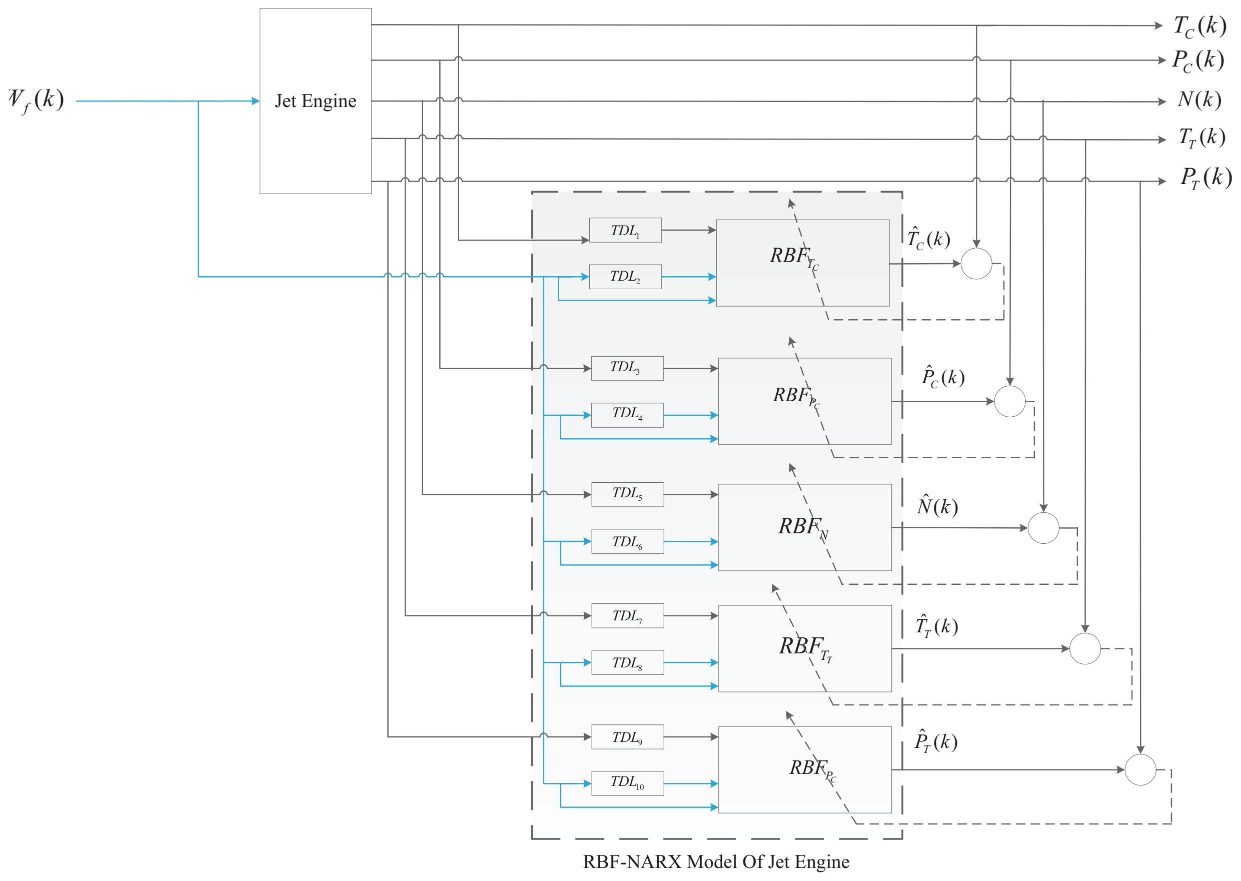


Figure 3.20: The architecture of the RBF-NARX model of jet engine during training phase.

Once the training stage is completed, the series-parallel architecture would be replaced with a parallel architecture. In this architecture, the outputs of the RBF-NARX model are fed back into the model rather than actual engine outputs. This is essential as the engine is prone to faults during testing stage. Figures 3.20 and 3.21 show the architectures of the RBF-NARX model used in this section.

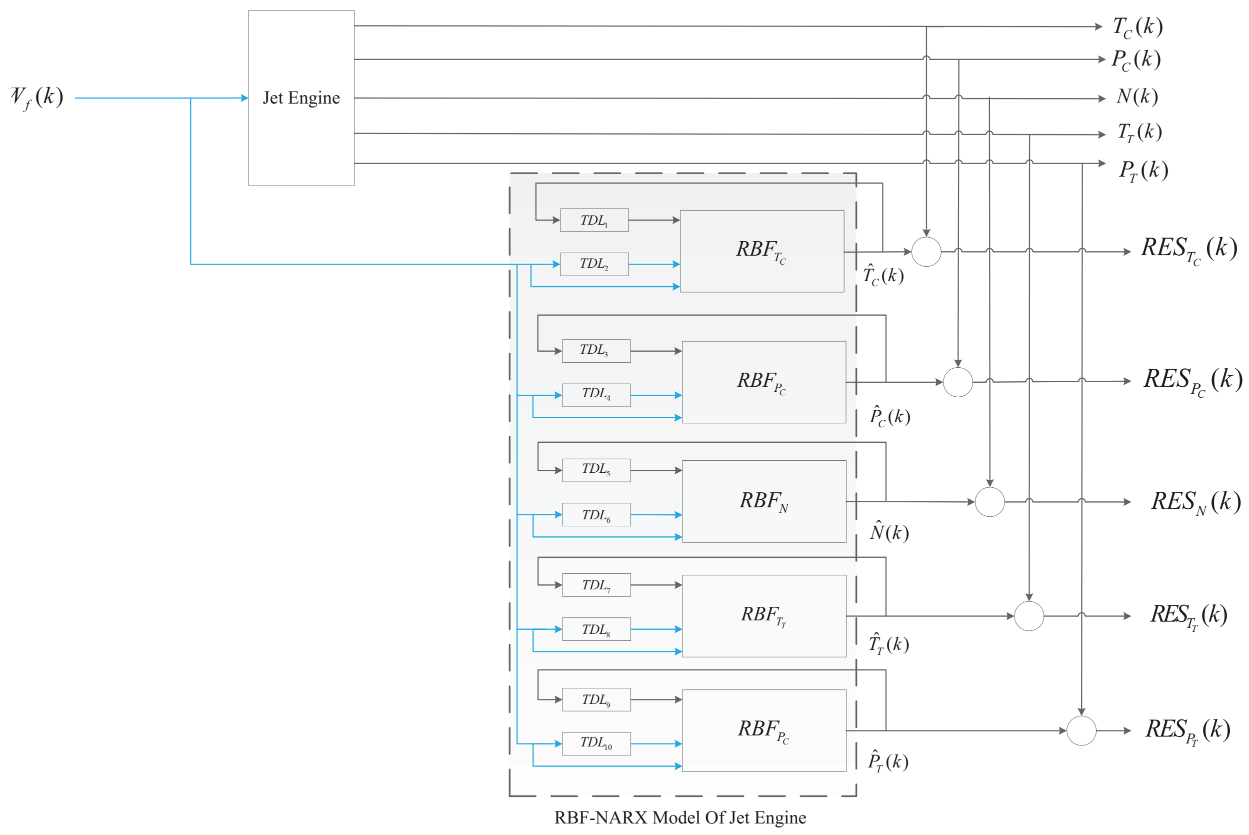


Figure 3.21: The architecture of the RBF-NARX model of jet engine during testing phase.

3.5.1 RBF-NARX Model of the Compressor Temperature

This section summarizes the construction of the RBF-NARX model for identifying the dynamics of the compressor temperature. Three different sizes are selected for training data set. First, several neural networks are constructed using 40% of available data (1200 out of 3001 samples). Second, we construct several networks using 50% of available data (1501 out of 3001 samples), and finally, we construct the RBF-NARX neural networks using 60% (1801 out of 3001 samples).

Each case starts with a small neural network structure, and then adds to the complexity by adding neurons and delays. To avoid complex networks we limit the number of hidden neurons to 20 and the number delays to 10 ($d_u \quad 10, d_y \quad 10$). Also, as previously discussed in Remark 3.3, we assume $d_y = d_u$ in order to limit the number of construction trials.

The summary of network construction is shown in Table 3.19.

Table 3.19: Summary of construction of RBF-NARX for modelling the compressor temperature (see the appendix for extensive summary).

# neurons	**# delays**	**# training samples**	**RMSE$^{T_C}_{total}$**	**RMSE$^{T_C}_{training}$**	**RMSE$^{T_C}_{test}$**	**%RMSE$^{T_C}_{total}$**	**mean (μ_{ae})**	**Std (σ_{ae})**
10	7	1200	2.2127	2.5475	1.9581	0.33685	1.7548	1.3482
10	8	1200	2.4748	2.8173	2.2174	0.37669	2.001	1.4565
10	4	1501	2.1221	2.1281	2.116	0.32288	1.7637	1.1803
10	5	1501	2.7501	2.9006	2.5908	0.41872	2.1888	1.6654
10	6	1501	1.6867	1.7608	1.6091	0.25666	1.3559	1.0034
10	7	1501	2.8032	2.8708	2.7339	0.42667	2.2943	1.6109
10	8	1501	1.5578	1.6449	1.4655	0.23702	1.2465	0.93455
10	4	1801	1.6739	1.8018	1.4611	0.25456	1.4078	0.90572
10	5	1801	2.7066	3.0159	2.1608	0.4121	2.1352	1.6635
10	6	1801	1.2448	1.3573	1.0536	0.18931	1.0106	0.72697
10	**7**	**1801**	**0.56069**	**0.60188**	**0.49245**	**0.085012**	**0.44584**	**0.34007**
10	8	1801	0.76254	0.82931	0.64958	0.11582	0.60207	0.46803
11	4	1200	45.4889	45.4371	45.5234	6.8836	45.4588	1.6552
11	5	1200	3.0822	3.5464	2.7294	0.46925	2.4649	1.8507
11	6	1200	1.4912	1.7501	1.2901	0.22692	1.1631	0.93331
11	7	1200	1.6108	1.8436	1.4348	0.24512	1.2955	0.95744
11	8	1200	1.2776	1.4696	1.1317	0.19436	1.0121	0.77985
11	4	1501	54.1284	54.0921	54.1647	8.191	54.095	1.9

Table 3.19: Summary of construction of RBF-NARX for modelling the compressor temperature (see the appendix for extensive summary).

# neurons	# delays	# training samples	RMSE$_{total}^{T_C}$	RMSE$_{training}^{T_C}$	RMSE$_{test}^{T_C}$	%RMSE$_{total}^{T_C}$	mean (μ_{ae})	Std (σ_{ae})
11	5	1501	3.1071	3.2399	2.9683	0.47296	2.5174	1.8215
11	6	1501	1.7994	1.9042	1.688	0.27388	1.4164	1.1099
11	7	1501	1.4374	1.5362	1.3312	0.21875	1.1301	0.88849

In summary the best performance is achieved by the network with following parameters:

Table 3.20: Best RBF-NARX for modeling the compressor temperature in terms of $RMSE_{test}$.

# neurons	# delays	# training samples	RMSE$_{total}^{T_C}$	RMSE$_{training}^{T_C}$	RMSE$_{test}^{T_C}$	%RMSE$_{total}^{T_C}$	mean (μ_{ae})	Std (σ_{ae})
10	**7**	**1801**	**0.56069**	**0.60188**	**0.49245**	**0.085012**	**0.44584**	**0.34007**

The engine output and the trained network output for both training and testing data are shown in Figure 3.22.

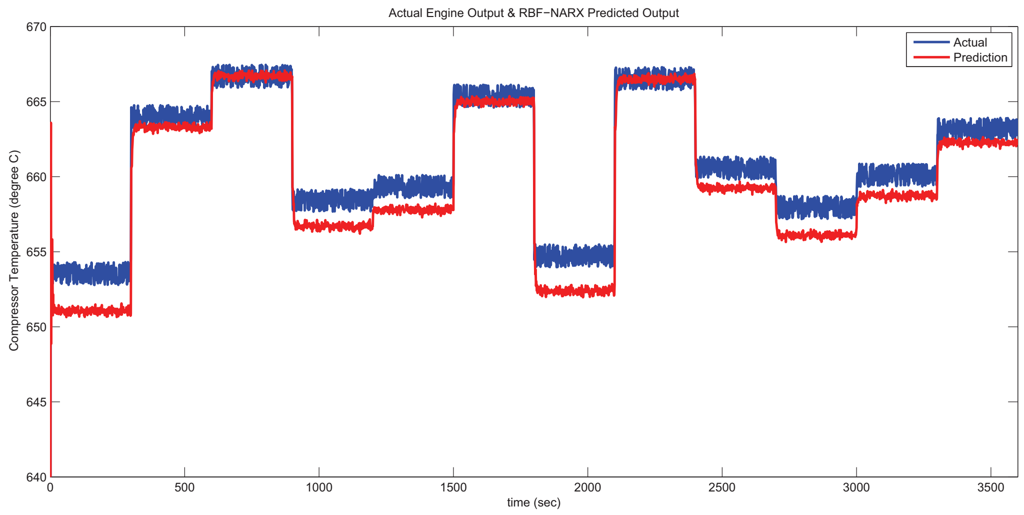


Figure 3.22: RBF-NARX model prediction and actual engine output (compressor temperature).

3.5.2 RBF-NARX Model of the Compressor Pressure

This section summarizes the construction of the RBF-NARX model for identifying the dynamics of the compressor pressure. Three different sizes are selected for training data set. First, several neural networks are constructed using 40% of available data (1200 out of 3001 samples). Second, we construct several networks using 50% of available data (1501 out of 3001 samples), and finally, we construct the RBF-NARX neural networks using 60% (1801 out of 3001 samples).

Each case starts with a small neural network structure, and then adds to the complexity by adding neurons and delays. To avoid complex networks we limit the number of hidden neurons to 20 and the number delays to 10 ($d_u \quad 10, d_y \quad 10$). Also, as previously discussed in Remark 3.3, we assume $d_y = d_u$ in order to limit the number of construction trials.

The summary of network construction is shown in Table 3.21.

Table 3.21: Summary of construction of the RBF-NARX for modelling the compressor pressure (see the appendix for extensive summary).

# neurons	# delays	# training samples	RMSE$^{P_C}_{total}$	RMSE$^{P_C}_{training}$	RMSE$^{P_C}_{test}$	%RMSE$^{P_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
12	7	1200	0.064396	0.07691	0.054486	0.54703	0.048821	0.042
12	8	1200	0.056249	0.067569	0.047225	0.47793	0.04124	0.038259
12	4	1501	1.8216	1.8187	1.8245	15.1073	1.8189	0.10028
12	5	1501	0.076144	0.079472	0.072662	0.64574	0.0601	0.046762
12	6	1501	0.093508	0.10086	0.08552	0.79507	0.071535	0.06023
12	7	1501	0.041526	0.047645	0.034328	0.35198	0.02924	0.029492
12	8	1501	0.10399	0.10738	0.10049	0.88305	0.083082	0.062552
12	4	1801	0.028344	0.028788	0.027662	0.23564	0.02311	0.016413
12	5	1801	0.051061	0.054583	0.045264	0.43124	0.041898	0.029191
12	6	1801	0.03562	0.040527	0.026609	0.30139	0.023059	0.027154

Table 3.21: Summary of construction of the RBF-NARX for modelling the compressor pressure (see the appendix for extensive summary).

# neurons	**# delays**	**# training samples**	**RMSE$^{P_C}_{total}$**	**RMSE$^{P_C}_{training}$**	**RMSE$^{P_C}_{test}$**	**%RMSE$^{P_C}_{total}$**	**mean (μ_{ae})**	**Std (σ_{ae})**
12	**7**	**1801**	**0.035583**	**0.041004**	**0.025357**	**0.30124**	**0.022888**	**0.027249**
12	8	1801	0.056521	0.064461	0.041869	0.48031	0.04089	0.039028
13	4	1200	1.2741	1.2871	1.2653	10.5876	0.70762	1.0597
13	5	1200	0.033538	0.041606	0.02685	0.28245	0.022606	0.024779
13	6	1200	0.083601	0.09174	0.077706	0.70904	0.068072	0.048539
13	7	1200	0.032045	0.037664	0.027674	0.26878	0.023128	0.022184
13	8	1200	0.050918	0.06243	0.041511	0.43244	0.036921	0.03507
13	4	1501	0.032368	0.035112	0.029368	0.27105	0.02435	0.021329
13	5	1501	0.045542	0.05078	0.039613	0.38571	0.033105	0.03128
13	6	1501	0.072641	0.075026	0.070174	0.61587	0.058254	0.043404
13	7	1501	0.069493	0.07567	0.062705	0.59061	0.052806	0.045183

In summary the best performance is achieved by the network with the following parameters:

Table 3.22: Best RBF-NARX for modeling the compressor pressure in terms of $RMSE_{test}$.

# neurons	**# delays**	**# training samples**	**RMSE$^{P_C}_{total}$**	**RMSE$^{P_C}_{training}$**	**RMSE$^{P_C}_{test}$**	**%RMSE$^{P_C}_{total}$**	**mean (μ_{ae})**	**Std (σ_{ae})**
12	**7**	**1801**	**0.035583**	**0.041004**	**0.025357**	**0.30124**	**0.022888**	**0.027249**

The engine output and the trained network output for both training and testing data are shown in Figure 3.23.

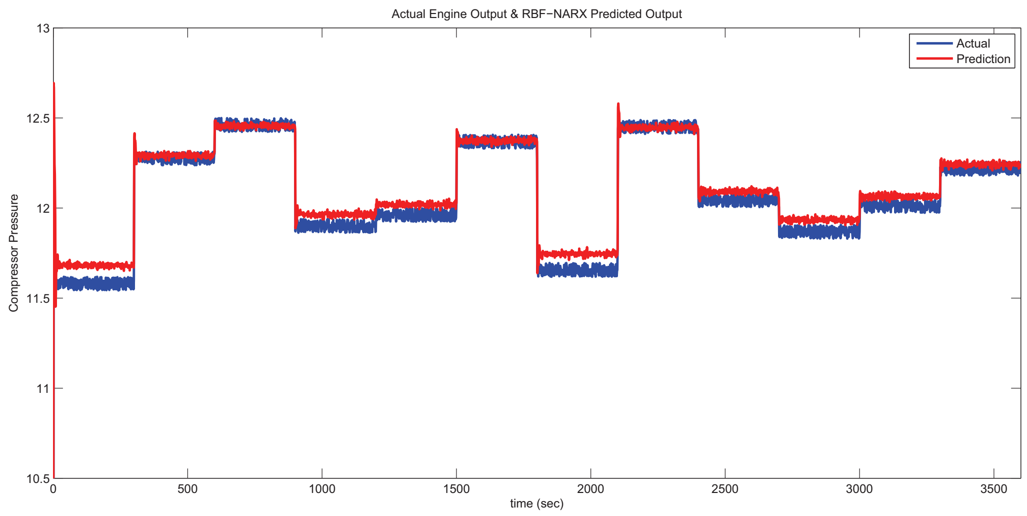


Figure 3.23: RBF-NARX model prediction and actual engine output (compressor pressure)

3.5.3 RBF-NARX Model of the Rotational Speed

This section summarizes the construction of the RBF-NARX model for identifying the jet engine rotational speed. Three different sizes are selected for training data set. First, several neural networks are constructed using 40% of available data (1200 out of 3001 samples). Second, we construct several networks using 50% of available data (1501 out of 3001 samples), and finally, we construct the RBF-NARX neural networks using 60% (1801 out of 3001 samples).

Each case starts with a small neural network structure, and then adds to the complexity by adding neurons and delays. To avoid complex networks we limit the number of hidden neurons to 20 and the number delays to 10 ($d_u \quad 10, d_y \quad 10$). Also, as previously discussed in Remark 3.3, we assume $d_y = d_u$ in order to limit the number of construction trials.

The summary of network construction is shown in Table 3.23.

Table 3.23: Summary of construction of the RBF-NARX for modelling the rotational speed (see appendix for extensive summary).

# neurons	# delays	# training samples	RMSE$^N_{total}$	RMSE$^N_{training}$	RMSE$^N_{test}$	%RMSE$^N_{total}$	mean (μ_{ae})	Std (σ_{ae})
17	8	1501	114.1124	114.047	114.1777	0.96081	114.0516	3.7236
17	4	1801	108.6896	108.6528	108.745	0.91517	108.6376	3.3633
17	5	1801	106.5698	106.4812	106.7026	0.89733	106.4511	5.0283
17	6	1801	115.2259	115.1704	115.3091	0.9702	115.1513	4.1453
17	7	1801	176.9808	176.9117	177.0844	1.4902	176.8843	5.8453
17	8	1801	161.857	161.8074	161.9314	1.3628	161.7737	5.1925
18	4	1200	109.285	109.1958	109.3444	0.92018	109.2308	3.4437
18	5	1200	102.3664	102.29	102.4172	0.86192	102.3211	3.0438
18	6	1200	169.5445	169.3732	169.6585	1.4276	169.4363	6.0576
18	7	1200	126.7108	126.6152	126.7745	1.0669	126.6515	3.8752
18	**8**	**1200**	**134.4747**	**212.6346**	**2.6076**	**1.1263**	**7.0685**	**134.3112**
18	4	1501	99.7465	99.6975	99.7955	0.83987	99.699	3.0786
18	5	1501	43.4613	52.6548	31.6962	0.36598	30.5798	30.8881
18	6	1501	180.9963	180.9068	181.0859	1.524	180.9185	5.3067
18	7	1501	120.5452	120.4738	120.6166	1.015	120.4874	3.7331
18	8	1501	158.1591	157.9758	158.3422	1.3317	158.0073	6.9275
18	4	1801	31.6943	34.3004	27.3204	0.2673	26.1631	17.8921
18	5	1801	107.6138	107.5772	107.6686	0.9061	107.562	3.3388
18	6	1801	125.7658	125.7153	125.8417	1.0589	125.6994	4.0889
18	7	1801	109.3846	109.3492	109.4377	0.92101	109.336	3.2609
18	8	1801	133.667	133.8736	133.3563	1.1254	94.2531	94.7957

In summary the best performance is achieved by the network with following parameters:

Table 3.24: **Best RBF-NARX for modeling the rotational speed in terms of $RMSE_{test}$.**

# neurons	# delays	# training samples	RMSE$^N_{total}$	RMSE$^N_{training}$	RMSE$^N_{test}$	%RMSE$^N_{total}$	mean (μ_{ae})	Std (σ_{ae})
18	**8**	**1200**	**134.4747**	**212.6346**	**2.6076**	**1.1263**	**7.0685**	**134.3112**

The engine output and the trained network output for both training and testing data are shown in Figure 3.24.

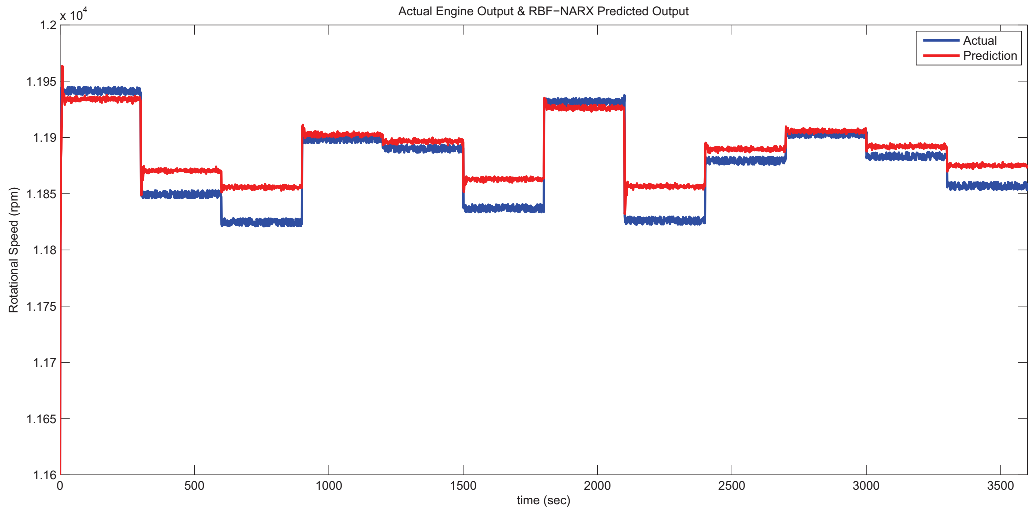


Figure 3.24: RBF-NARX model prediction and actual engine output (rotational speed).

3.5.4 RBF-NARX Model of the Turbine Temperature

This section summarizes the construction of the RBF-NARX model for identifying the jet engine turbine temperature. Three different sizes are selected for training data set. First, several neural networks are constructed using 40% of available data (1200 out of 3001 samples). Second, we construct several networks using 50% of available data (1501 out of 3001 samples), and finally, we construct the RBF-NARX neural networks using 60% (1801 out of 3001 samples).

Each case starts with a small neural network structure, and then adds to the complexity by adding neurons and delays. To avoid complex networks we limit the number of hidden neurons to 20 and the number delays to 10 ($d_u \quad 10, d_y \quad 10$). Also, as previously discussed in Remark 3.3, we assume $d_y = d_u$ in order to limit the number of construction trials.

The summary of network construction is shown in Table 3.25.

Table 3.25: Summary of construction of the RBF-NARX for modelling the turbine temperature (see the appendix for extensive summary).

# neurons	# delays	# training samples	RMSE$^{T_T}_{total}$	RMSE$^{T_T}_{training}$	RMSE$^{T_T}_{test}$	%RMSE$^{T_T}_{total}$	mean (μ_{ae})	Std (σ_{ae})
10	4	1200	248.0781	179.3072	284.8268	15.7583	153.0082	195.3048
10	5	1200	207.1761	148.416	238.4197	13.2638	175.6342	109.9026
10	6	1200	116.529	106.6732	122.657	9.5312	77.1716	87.3275
10	7	1200	107.7162	99.2078	113.0304	8.6593	80.7307	71.3234
10	8	1200	172.1691	174.8978	170.3266	13.1648	120.1952	123.2898
10	4	1501	147.9648	147.6238	148.3052	9.397	147.6888	9.0341
10	5	1501	210.1373	171.3942	242.7947	13.4563	143.7779	153.2757
10	6	1501	183.362	149.3543	212.0001	11.7609	159.1257	91.123
10	7	1501	191.311	154.2249	222.3129	12.2994	165.1711	96.5481
10	8	1501	116.2038	109.0299	122.9642	9.4428	77.3203	86.7606
10	**4**	**1801**	**178.8614**	**230.875**	**2.4151**	**9.2798**	**47.9622**	**172.3396**
10	5	1801	131.9345	130.7706	133.6623	7.821	61.2848	116.8565
10	6	1801	180.3875	158.4	209.0921	11.5788	156.4552	89.8003
10	7	1801	152.1321	130.5643	179.7081	9.8619	129.5118	79.8311
10	8	1801	123.4252	113.3757	137.1327	9.9988	84.0548	90.3952
11	4	1200	220.7185	159.9259	253.2489	14.0443	192.6668	107.7034
11	5	1200	211.3474	150.6581	243.5286	13.4975	147.5283	151.363

Table 3.25: Summary of construction of the RBF-NARX for modelling the turbine temperature (see the appendix for extensive summary).

# neurons	**# delays**	**# training samples**	**RMSE$^{T_T}_{total}$**	**RMSE$^{T_T}_{training}$**	**RMSE$^{T_T}_{test}$**	**%RMSE$^{T_T}_{total}$**	**mean (μ_{ae})**	**Std (σ_{ae})**
11	6	1200	181.9475	127.4505	210.5694	11.68	157.7301	90.7129
11	7	1200	185.3197	129.6805	214.5254	11.8944	160.6223	92.4485
11	8	1200	116.0043	110.1355	119.7551	9.423	79.1754	84.7977
11	4	1501	242.1402	198.9689	278.7254	15.5464	152.3787	188.2138

In summary the best performance is achieved by the network with following parameters:

Table 3.26: Best RBF-NARX for modeling the turbine temperature in terms of $RMSE_{test}$.

# neurons	**# delays**	**# training samples**	**RMSE$^{T_T}_{total}$**	**RMSE$^{T_T}_{training}$**	**RMSE$^{T_T}_{test}$**	**%RMSE$^{T_T}_{total}$**	**mean (μ_{ae})**	**Std (σ_{ae})**
10	**4**	**1801**	**178.8614**	**230.875**	**2.4151**	**9.2798**	**47.9622**	**172.3396**

The engine output and the trained network output for both training and testing data are shown in Figure 3.25.

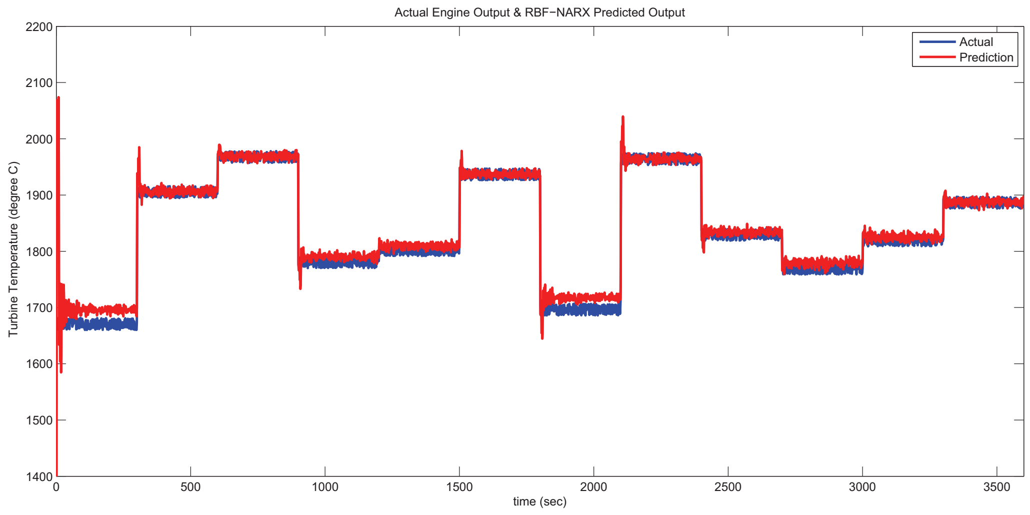


Figure 3.25: RBF-NARX model prediction and actual engine output (turbine temperature)

3.5.5 RBF-NARX Model of the Turbine Pressure

This section summarizes the construction of the RBF-NARX model for identifying the jet engine turbine pressure. Three different sizes are selected for training data set. First, several neural networks are constructed using 40% of available data (1200 out of 3001 samples). Second, we construct several networks using 50% of available data (1501 out of 3001 samples), and finally, we construct the RBF-NARX neural networks using 60% (1801 out of 3001 samples).

Each case starts with a small neural network structure, and then adds to the complexity by adding neurons and delays. To avoid complex networks we limit the number of hidden neurons to 20, and the number delays to 10 ($d_u \quad 10, d_y \quad 10$). Also, as previously discussed in Remark 3.3, we assume $d_y = d_u$ in order to limit the number of construction trials.

The summary of the network construction is shown in Table 3.27.

Table 3.27: Summary of construction of the RBF-NARX for modelling the turbine pressure (see the appendix for extensive summary).

# neurons	# delays	# training samples	RMSE$^{P_T}_{total}$	RMSE$^{P_T}_{training}$	RMSE$^{P_T}_{test}$	%RMSE$^{P_T}_{total}$	Mean (μ_{ae})	Std(σ_{ae})
10	6	1200	0.11497	0.13096	0.10294	2.4936	0.091399	0.069754
10	7	1200	0.073515	0.089074	0.060983	1.6013	0.054738	0.049082
10	8	1200	0.094749	0.1096	0.083398	2.0586	0.073986	0.059199
10	4	1501	0.03267	0.030641	0.034581	0.68331	0.027145	0.018182
10	5	1501	0.047137	0.054334	0.038616	1.0301	0.032015	0.034603
10	6	1501	0.047366	0.044367	0.050188	1.0006	0.041462	0.022904
10	7	1501	0.047904	0.054466	0.040282	1.0461	0.033841	0.033912
10	8	1501	0.088548	0.097417	0.078678	1.9289	0.066065	0.058968
10	4	1801	0.030375	0.029691	0.031373	0.63827	0.024336	0.018179
10	5	1801	0.065731	0.066521	0.064526	1.3976	0.057415	0.032005
10	**6**	**1801**	**0.026215**	**0.030958**	**0.01674**	**0.56995**	**0.015856**	**0.02088**
10	7	1801	0.036441	0.04369	0.021358	0.7974	0.022423	0.028731
10	8	1801	0.04342	0.052328	0.024601	0.95268	0.027342	0.033735
11	4	1200	1.4267	1.4251	1.4277	29.5852	1.4257	0.051337
11	5	1200	0.030405	0.029495	0.030997	0.63677	0.025548	0.016489
11	6	1200	0.040449	0.050807	0.031723	0.88388	0.027711	0.02947
11	7	1200	0.12818	0.14652	0.11433	2.7813	0.10118	0.078712
11	8	1200	0.034697	0.046994	0.023121	0.76037	0.021102	0.027548
11	4	1501	0.071209	0.067685	0.074568	1.5146	0.062443	0.034233
11	5	1501	0.072483	0.069072	0.075743	1.5416	0.063722	0.034551
11	6	1501	0.064206	0.063003	0.065387	1.3644	0.055589	0.032133

In summary the best performance is achieved by the network with the following parameters:

Table 3.28: Best RBF-NARX for modeling the turbine pressure in terms of $RMSE_{test}$.

# neurons	# delays	# training samples	RMSE$^{P_T}_{total}$	RMSE$^{P_T}_{training}$	RMSE$^{P_T}_{test}$	%RMSE$^{P_T}_{total}$	Mean (μ_{ae})	Std(σ_{ae})
10	**6**	**1801**	**0.026215**	**0.030958**	**0.01674**	**0.56995**	**0.015856**	**0.02088**

The engine output and the trained network output for both training and testing data are shown in Figure 3.26.

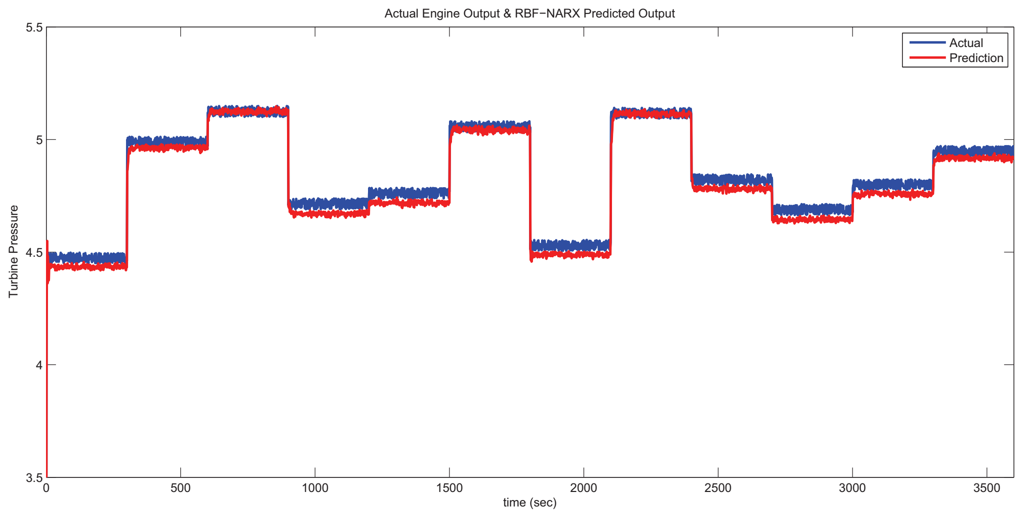


Figure 3.26: RBF-NARX model prediction and the actual engine output (turbine pressure).

3.6 Jet Engine Dynamics Identification using SVM-NARX

The use of SVR-NARX model in system identification has been reported in various publications. A general framework for nonlinear system identification with SVR based on NARX model is presented in [1]. In another framework, [2] combines Least-Square Support Vector Machines (LS-SVM) with NARX model for identification of Weiner-Hammerstein systems. Other SVR-based system identification methods together with ARX models are reported in [4, 3, 5, 6]. In this research, support vector regression is used in an NARX model to identify the jet engine dynamics. Figure 2.10 shows the structure of the SVR-NARX system identification algorithm.

As previously described in Chapter 2 a series-parallel architecture is used for training of SVM-NARX model as shown in Figure 3.27. In this architecture the engine output is fed back into the SVM-NARX model, assuming that the engine is fault-free during the training stage.

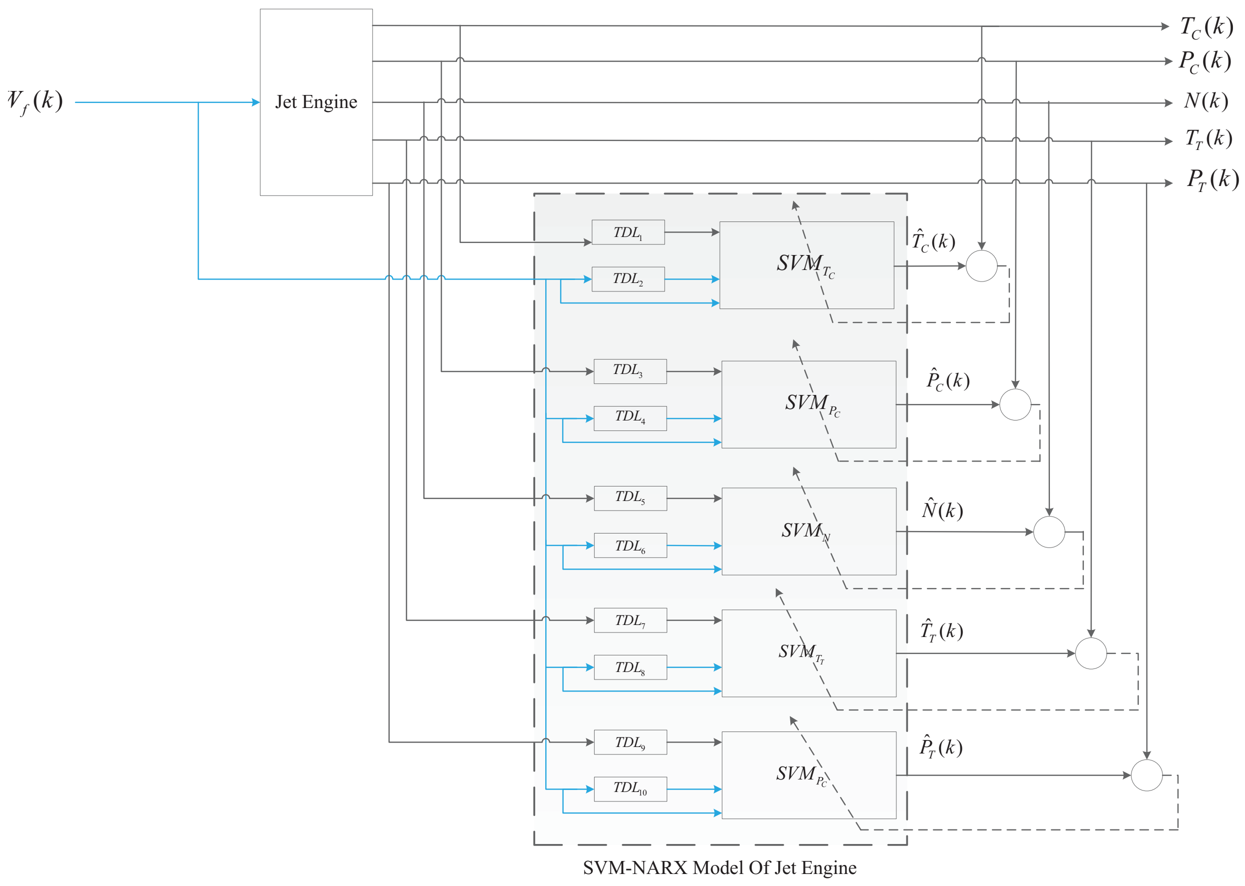


Figure 3.27: The architecture of the SVM-NARX model of jet engine during training phase.

Once the training stage is completed, the series-parallel architecture would be replaced with a parallel architecture. In this architecture, the outputs of SVM-NARX model are fed back into the model rather than actual engine outputs. This is essential as the engine is prone to fault during testing stage. Figure 3.28 shows the parallel architecture of SVM-NARX model used in this section.

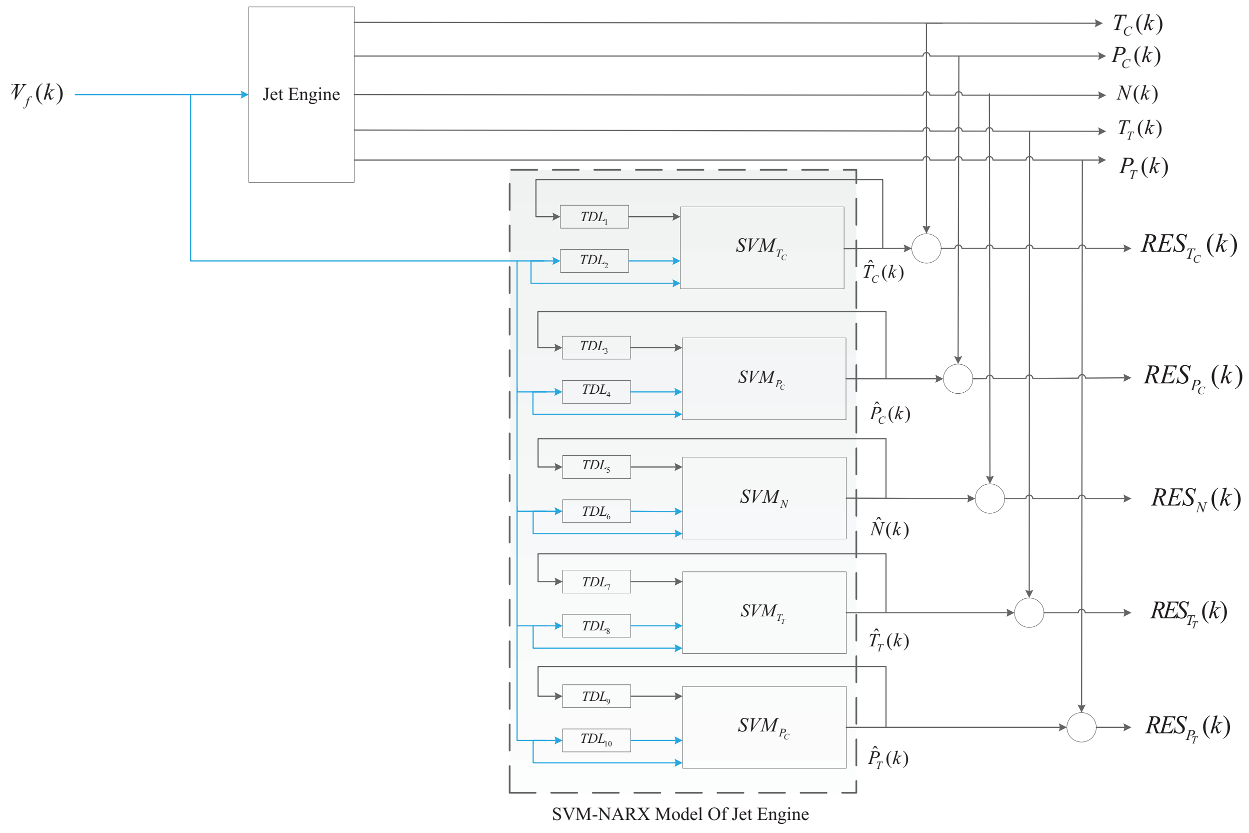


Figure 3.28: The architecture of the SVM-NARX model of jet engine during testing phase.

3.6.1 SVM-NARX Model of the Compressor Temperature

This section summarizes the construction of the SVM-NARX model for identifying the dynamics of the compressor temperature. Three different sizes are selected for training data set. First, several support vector regressions are constructed using 40% of available data (1200 out of 3001 samples).

Second, we construct several SVRs using 50% of available data (1501 out of 3001 samples), and finally, we construct the SVM-NARX models using 60% (1801 out of 3001 samples).

We put the goal to achieve an RMSE percentage less than $\epsilon \quad 1\%$. Initially we limit the delays to $(d_u \quad 10, d_y \quad 10)$. If it does not satisfy our constraint, then we would increase the number of delays. Also, as previously discussed in Remark 3.3, we assume $d_y = d_u$ in order to limit the number of construction trials.

The summary of construction is shown in Table 3.29.

Table 3.29: Summary of construction of the SVM-NARX for modelling the compressor temperature (see the appendix for extensive summary).

# delays	# training samples	RMSE$^{T_C}_{total}$	RMSE$^{T_C}_{training}$	RMSE$^{T_C}_{test}$	%RMSE$^{T_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
4	1200	5.785	6.1078	5.5595	0.88026	4.7571	3.2922
5	1200	3.2587	3.4575	3.1191	0.49584	2.5706	2.0029
6	1200	6.7519	6.9366	6.626	1.0262	5.8765	3.3252
7	**1200**	**2.6112**	**2.7304**	**2.5287**	**0.39709**	**2.0356**	**1.6357**
8	1200	8.5326	8.646	8.4563	1.2962	7.6082	3.863
4	1501	10.0526	10.2897	9.8096	1.5335	7.1987	7.0172
5	1501	2.7116	2.6177	2.8024	0.41237	2.1957	1.5913
6	1501	8.0053	8.0475	7.9629	1.22	6.1615	5.1112
7	1501	3.1256	3.0234	3.2246	0.47547	2.4852	1.8958
8	1501	10.1568	10.4808	9.8219	1.5495	7.2417	7.1222
4	1801	4.0933	3.9579	4.2883	0.6212	3.2124	2.537
5	1801	5.1256	5.1964	5.0174	0.77866	4.3276	2.7467
6	1801	25.5922	27.3954	22.6185	3.8929	19.709	16.3266
7	1801	12.9035	13.9279	11.192	1.9636	10.4707	7.5416
8	1801	4.8782	5.1378	4.4605	0.74155	3.5143	3.3836

In summary the best performance is achieved by the network with the following parameters:

Table 3.30: Best SVM-NARX for modeling the compressor temperature in terms of $RMSE_{test}$.

# delays	# training samples	RMSE$^{T_C}_{total}$	RMSE$^{T_C}_{training}$	RMSE$^{T_C}_{test}$	%RMSE$^{T_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
7	**1200**	**2.6112**	**2.7304**	**2.5287**	**0.39709**	**2.0356**	**1.6357**

The engine output and the trained network output for both training and testing data are shown in Figure 3.29.

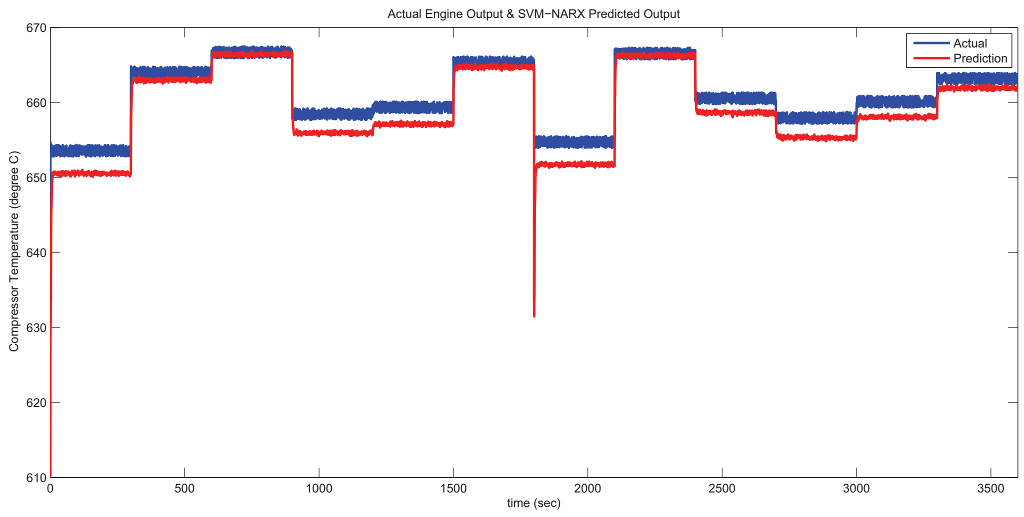


Figure 3.29: SVM-NARX model prediction and the actual engine output (compressor temperature).

3.6.2 SVM-NARX Model of the Compressor Pressure

This section summarizes the construction of the SVM-NARX model for identifying the dynamics of compressor pressure. Three different sizes are selected for training data set. First, several support vector regressions are constructed using 40% of available data (1200 out of 3001 samples).

Second, we construct several SVRs using 50% of available data (1501 out of 3001 samples), and finally, we construct the SVM-NARX models using 60% (1801 out of 3001 samples).

We put the goal to achieve an RMSE percentage less than $\epsilon \quad 1\%$. Initially we limit the delays to $(d_u \quad 10, d_y \quad 10)$. If it does not satisfy our constraint, then we would increase the number of delays. Also, as previously discussed in Remark 3.3, we assume $d_y = d_u$ in order to limit the number of construction trials.

The summary of construction is shown in Table 3.31.

Table 3.31: Summary of construction of the SVM-NARX for modelling the compressor pressure (see the appendix for extensive summary).

# delays	**# training samples**	**RMSE**$^{P_C}_{total}$	**RMSE**$^{P_C}_{training}$	**RMSE**$^{P_C}_{test}$	**%RMSE**$^{P_C}_{total}$	**mean** (μ_{ae})	**Std** (σ_{ae})
4	1200	0.04434	0.044539	0.044207	0.37271	0.023476	0.037618
5	1200	0.06586	0.074462	0.059439	0.56046	0.042214	0.050556
6	1200	0.054944	0.058191	0.052668	0.46561	0.032905	0.044004
7	1200	0.051974	0.051877	0.052039	0.43826	0.022331	0.046936
8	1200	0.070633	0.067559	0.072611	0.59467	0.029654	0.064113
4	1501	0.047934	0.046986	0.048864	0.40478	0.030515	0.03697
5	1501	0.046661	0.046592	0.04673	0.39495	0.029562	0.036105
6	1501	0.059857	0.058357	0.061321	0.50837	0.033006	0.049939
7	1501	0.10016	0.10284	0.097396	0.85469	0.065978	0.075359
8	1501	0.047577	0.047181	0.04797	0.40371	0.025682	0.040053
4	**1801**	**0.041185**	**0.037001**	**0.046766**	**0.34187**	**0.028081**	**0.030129**
5	1801	0.16758	0.19602	0.11212	1.4287	0.11603	0.12093
6	1801	0.064463	0.04931	0.082111	0.53039	0.022916	0.060257
7	1801	0.073084	0.078509	0.064088	0.61647	0.04068	0.060721
8	1801	0.058766	0.044294	0.07544	0.48336	0.02175	0.054597

In summary the best performance is achieved by the network with the following parameters:

Table 3.32: Best SVM-NARX for modeling the compressor pressure in term of $RMSE_{test}$.

# delays	# training samples	RMSE$^{P_C}_{total}$	RMSE$^{P_C}_{training}$	RMSE$^{P_C}_{test}$	%RMSE$^{P_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
4	**1200**	**0.04434**	**0.044539**	**0.044207**	**0.37271**	**0.023476**	**0.037618**

The engine output and the trained network output for both training and testing data are shown in Figure 3.30.

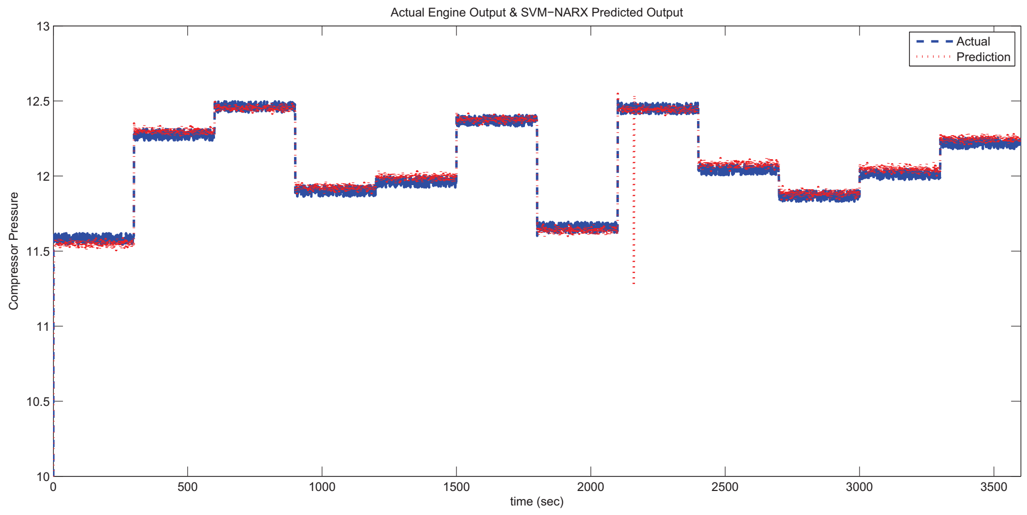


Figure 3.30: SVM-NARX model prediction and actual engine output (compressor pressure).

3.6.3 SVM-NARX Model of the Rotational Speed

This section summarizes the construction of the SVM-NARX model for identifying the dynamics of the rotational speed. Three different sizes are selected for training data set. First, several support vector regressions are constructed using 40% of available data (1200 out of 3001 samples). Second, we construct several SVRs using 50% of available data (1501 out of 3001 samples), and finally, we construct the SVM-NARX models using 60% (1801 out of 3001 samples).

We put the goal to achieve an RMSE percentage less than $\epsilon \quad 1\%$. Initially we limit the delays to $(d_u \quad 10, d_y \quad 10)$. If it does not satisfy our constraint, then we would increase the number of delays. Also, as previously discussed in Remark 3.3, we assume $d_y = d_u$ in order to limit the number of construction trials.

The summary of construction is shown in Table 3.33.

Table 3.33: Summary of construction of the SVM-NARX for modelling the rotational speed (see the appendix for extensive summary).

# delays	# training samples	RMSE$^N_{total}$	RMSE$^N_{training}$	RMSE$^N_{test}$	%RMSE$^N_{total}$	mean (μ_{ae})	Std (σ_{ae})
4	1200	26.3914	26.3745	26.4026	0.22282	22.6819	13.4932
5	1200	27.3764	27.3992	27.3612	0.23114	23.5228	14.0064
6	1200	27.8955	27.874	27.9099	0.23553	23.9589	14.2887
7	1200	27.4287	27.3904	27.4543	0.23158	23.7126	13.7868
8	1200	26.1403	26.1954	26.1035	0.2207	22.3773	13.513
4	**1501**	**24.6728**	**26.5164**	**22.6791**	**0.20831**	**21.236**	**12.562**
5	1501	26.7274	28.7473	24.5411	0.22566	22.9553	13.6908
6	1501	28.2652	30.4459	25.9008	0.23865	24.18	14.6386
7	1501	26.9222	28.8732	24.8176	0.2273	23.1971	13.665
8	1501	26.0438	27.9951	23.9333	0.21988	22.4468	13.2081
4	1801	26.545	27.0505	25.7681	0.22412	22.7997	13.5956
5	1801	27.4482	27.9494	26.6783	0.23175	23.5607	14.0829
6	1801	25.4999	25.9773	24.7663	0.21528	21.9856	12.9192
7	1801	25.2779	25.7547	24.545	0.21342	21.7604	12.864
8	1801	26.1525	26.6325	25.4153	0.2208	22.5576	13.2339

In summary the best performance is achieved by the network with the following parameters:

Table 3.34: Best SVM-NARX for modeling the rotational speed in terms of $RMSE_{test}$.

# delays	# training samples	RMSE$^N_{total}$	RMSE$^N_{training}$	RMSE$^N_{test}$	%RMSE$^N_{total}$	mean (μ_{ae})	Std (σ_{ae})
4	**1501**	**24.6728**	**26.5164**	**22.6791**	**0.20831**	**21.236**	**12.562**

The engine output and the trained network output for both training and testing data are shown in Figure 3.31.

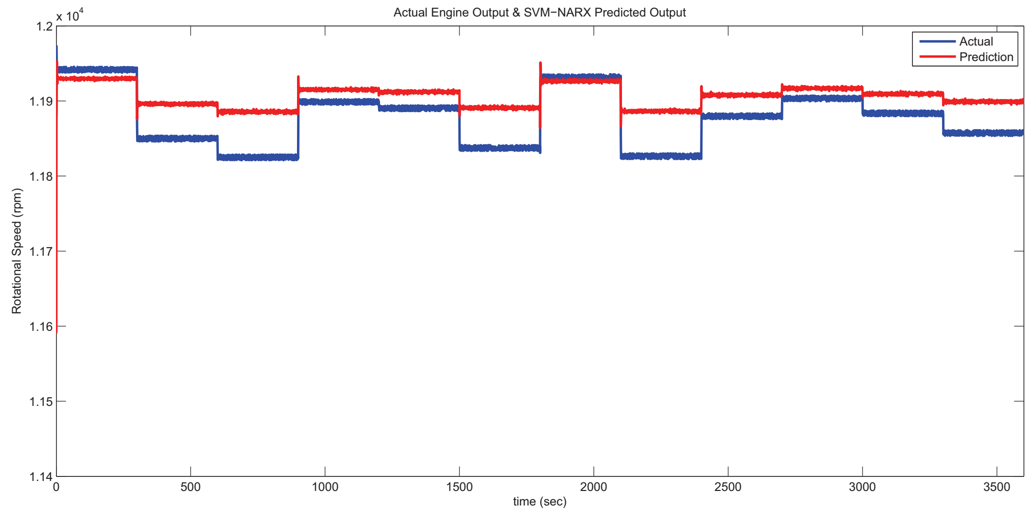


Figure 3.31: The SVM-NARX model prediction and the actual engine output (rotational speed).

3.6.4 SVM-NARX Model of the Turbine Temperature

This section summarizes the construction of the SVM-NARX model for identifying the dynamics of the turbine temperature. Three different sizes are selected for training data set. First, several support vector regressions are constructed using 40% of available data (1200 out of 3001 samples). Second, we construct several SVRs using 50% of available data (1501 out of 3001 samples), and finally, we construct the SVM-NARX models using 60% (1801 out of 3001 samples).

We put the goal to achieve an RMSE percentage less than $\epsilon \quad 1\%$. Initially we limit the delays to $(d_u \quad 10, d_y \quad 10)$. If it does not satisfy our constraint, then we would increase the number of delays. Also, as previously discussed in Remark 3.3, we assume $d_y = d_u$ in order to limit the number of construction trials.

The summary of construction is shown in Table 3.35.

Table 3.35: Summary of construction of the SVM-NARX for modelling the turbine temperature (see the appendix for extensive summary).

# delays	**# training samples**	**RMSE$^{T_T}_{total}$**	**RMSE$^{T_T}_{training}$**	**RMSE$^{T_T}_{test}$**	**%RMSE$^{T_T}_{total}$**	**mean (μ_{ae})**	**Std (σ_{ae})**
4	1200	183.6261	189.9563	179.2832	10.4182	160.2343	89.6932
5	1200	166.4114	175.3166	160.2015	9.4955	141.1499	88.1518
6	**1200**	**104.3983**	**107.0395**	**102.6002**	**5.9073**	**90.6818**	**51.7326**
7	1200	136.832	133.9356	138.7288	7.6189	123.4487	59.0255
8	1200	166.3491	161.8485	169.2822	9.2006	154.693	61.1777
4	1501	293.6453	273.8494	312.1944	16.4391	262.022	132.5706
5	1501	282.3101	269.7902	294.3017	15.649	259.5313	111.1059
6	1501	215.896	209.24	222.355	11.9547	202.3673	75.2297
7	1501	249.4072	235.4363	262.6403	13.9182	226.8929	103.5634
8	1501	236.7342	224.6235	248.2584	13.3207	212.1327	105.0933
4	1801	408.2675	398.3448	422.7191	22.527	380.7186	147.4427
5	1801	307.8869	291.5684	330.865	17.1951	274.3306	139.7866
6	1801	202.5877	190.7599	219.1402	11.2666	183.4603	85.938
7	1801	272.9621	258.0228	293.9563	15.1594	248.0573	113.9212
8	1801	205.2928	208.9996	199.6019	11.6387	174.1967	108.6398

In summary the best performance is achieved by the network with the following parameters:

Table 3.36: Best SVM-NARX for modeling the turbine temperature in terms of $RMSE_{test}$.

# delays	# training samples	RMSE$^{T_T}_{total}$	RMSE$^{T_T}_{training}$	RMSE$^{T_T}_{test}$	%RMSE$^{T_T}_{total}$	mean (μ_{ae})	Std (σ_{ae})
6	1200	104.3983	107.0395	102.6002	5.9073	90.6818	51.7326

The engine output and the trained network output for both training and testing data are shown in Figure 3.32.

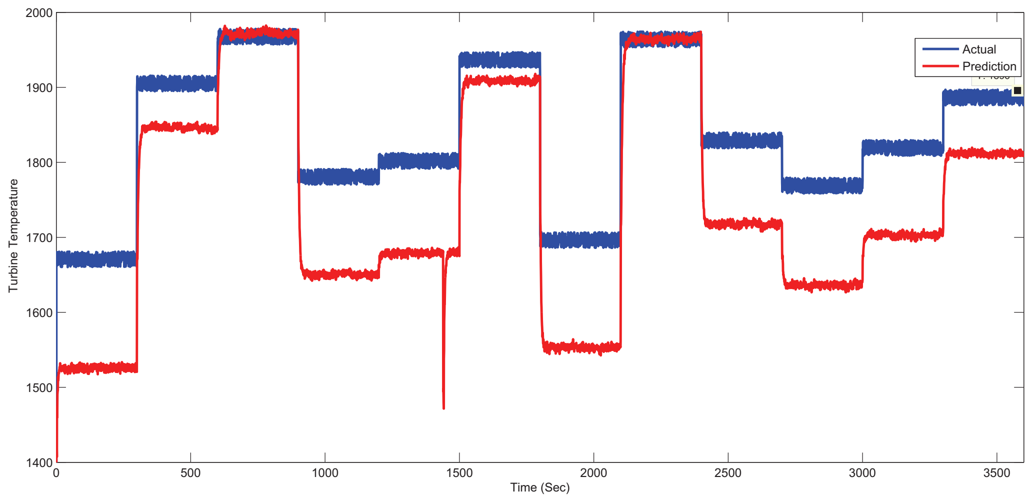


Figure 3.32: SVM-NARX model prediction and actual engine output (turbine temperature).

3.6.5 SVM-NARX Model of the Turbine Pressure

This section summarizes the construction of the SVM-NARX model for identifying the dynamics of turbine pressure. Three different sizes are selected for training data set. First, several support vector regressions are constructed using 40% of available data (1200 out of 3001 samples). Second, we construct several SVRs using 50% of available data (1501 out of 3001 samples), and finally, we construct the SVM-NARX models using 60% (1801 out of 3001 samples).

We put the goal to achieve an RMSE percentage less than $\epsilon \quad 1\%$. Initially we limit the delays to $(d_u \quad 10, d_y \quad 10)$. If it does not satisfy our constraint, then we would increase the number of delays. Also, as previously discussed in Remark 3.3, we assume $d_y = d_u$ in order to limit the number of construction trials.

The summary of construction is shown in Table 3.37.

Table 3.37: Summary of construction of the SVM-NARX for modelling the turbine pressure (see the appendix for extensive summary).

# delays	# training samples	RMSE$^{P_T}_{total}$	RMSE$^{P_T}_{training}$	RMSE$^{P_T}_{test}$	%RMSE$^{P_T}_{total}$	Mean (μ_{ae})	Std(σ_{ae})
4	1200	0.14436	0.166	0.12793	3.164	0.099075	0.10501
5	1200	0.22767	0.25312	0.209	4.9621	0.16848	0.15314
6	1200	0.13213	0.14305	0.12432	2.858	0.10348	0.08217
7	1200	0.36862	0.40489	0.34232	8.0233	0.27583	0.24456
8	1200	0.26315	0.27079	0.25794	5.6279	0.22447	0.13736
4	1501	0.15671	0.15761	0.1558	3.4091	0.11986	0.10096
5	1501	0.14812	0.14611	0.15011	3.203	0.11887	0.088379
6	1501	0.15141	0.14926	0.15353	3.2731	0.12131	0.090617
7	1501	0.11376	0.10654	0.12055	2.4167	0.097374	0.058823
8	1501	0.11595	0.11134	0.12038	2.4909	0.094515	0.067168
4	**1801**	**0.088902**	**0.084001**	**0.095787**	**1.8751**	**0.075015**	**0.047715**
5	1801	0.14606	0.14218	0.15169	3.0978	0.12598	0.073916
6	1801	0.12097	0.11917	0.12363	2.5699	0.10214	0.064829
7	1801	0.18853	0.20295	0.16453	4.0615	0.14994	0.1143
8	1801	0.27965	0.3162	0.21338	6.0711	0.21025	0.18441

In summary the best performance is achieved by the network with the following parameters:

Table 3.38: Best SVM-NARX for modeling the turbine pressure in terms of $RMSE_{test}$.

# delays	# training samples	RMSE$^{P_T}_{total}$	RMSE$^{P_T}_{training}$	RMSE$^{P_T}_{test}$	%RMSE$^{P_T}_{total}$	Mean (μ_{ae})	Std(σ_{ae})
4	**1801**	**0.088902**	**0.084001**	**0.095787**	**1.8751**	**0.075015**	**0.047715**

The engine output and the trained network output for both training and testing data are shown in Figure 3.33.

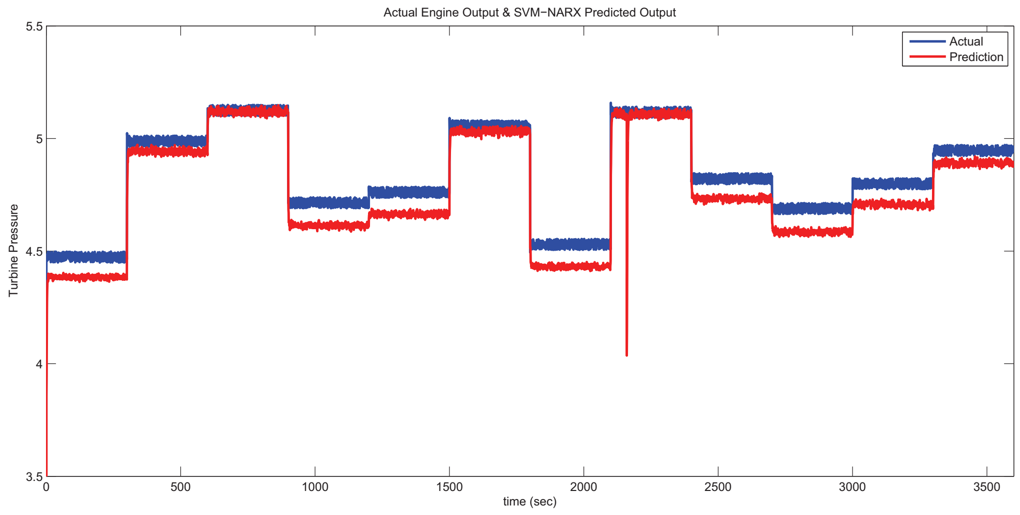


Figure 3.33: The SVM-NARX model prediction and the actual engine output (turbine pressure).

3.7 Jet Engine Dynamic Identification with Ensemble Learning

This section describes identification of jet engine dynamics using ensemble methods. Training an ensemble system can be generally divided into three steps [201] as shown in Figure 3.34. The first step is *ensemble generation*, which consists of generating a set of models. It often happens that a number of redundant models are generated during ensemble generation. The next step is *ensemble pruning* where the pool of generated models are trimmed in order to achieve maximum diversity among the learners. Finally, the models are combined in the ensemble integration step, where the final prediction is formed based on the models' prediction. Different ensemble architectures can be made by considering different methodologies for each of these three steps.

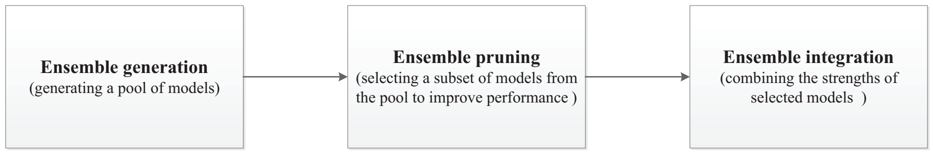

Ensemble generation
(generating a pool of models)
Ensemble pruning
(selecting a subset of models from the pool to improve performance)
Ensemble integration
(combining the strengths of selected models)

Figure 3.34: Ensemble learning stages [223].

In this section three different methodologies are considered for creating ensemble model of the jet engine dynamics. Different strategies are first discussed, and then applied toward jet engine identification problem. The simulation results are presented at the end of this section.

As previously described in Chapter 2 a series-parallel architecture is used for training of ensemble model as shown in Figure 3.35.

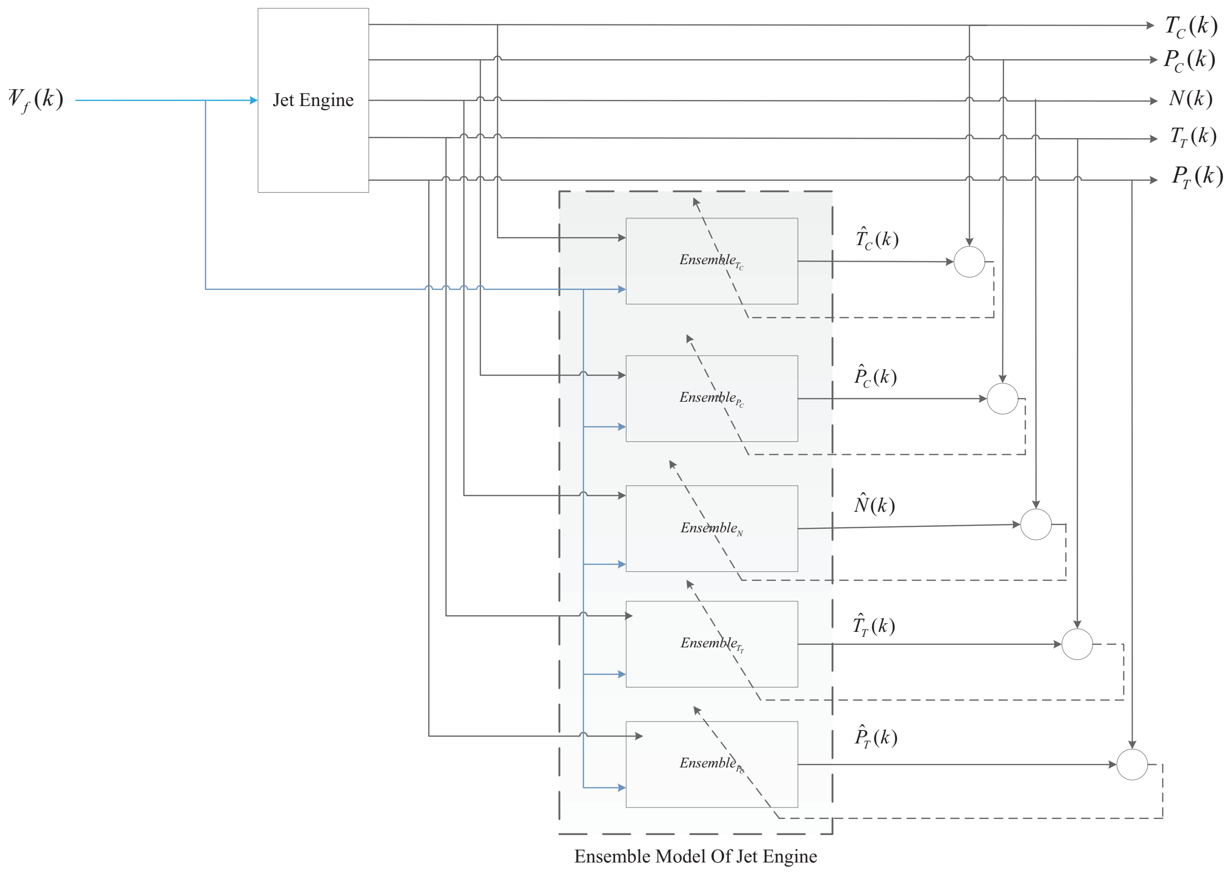


Figure 3.35: Architecture of ensemble during the training phase.

Once the training stage is completed, the series-parallel architecture would be replaced with a parallel architecture. This assumption is valid since the a trained model outputs replicate the outputs of the actual jet engine. The architecture of the ensemble during the testing phase is shown in Figure 3.36.

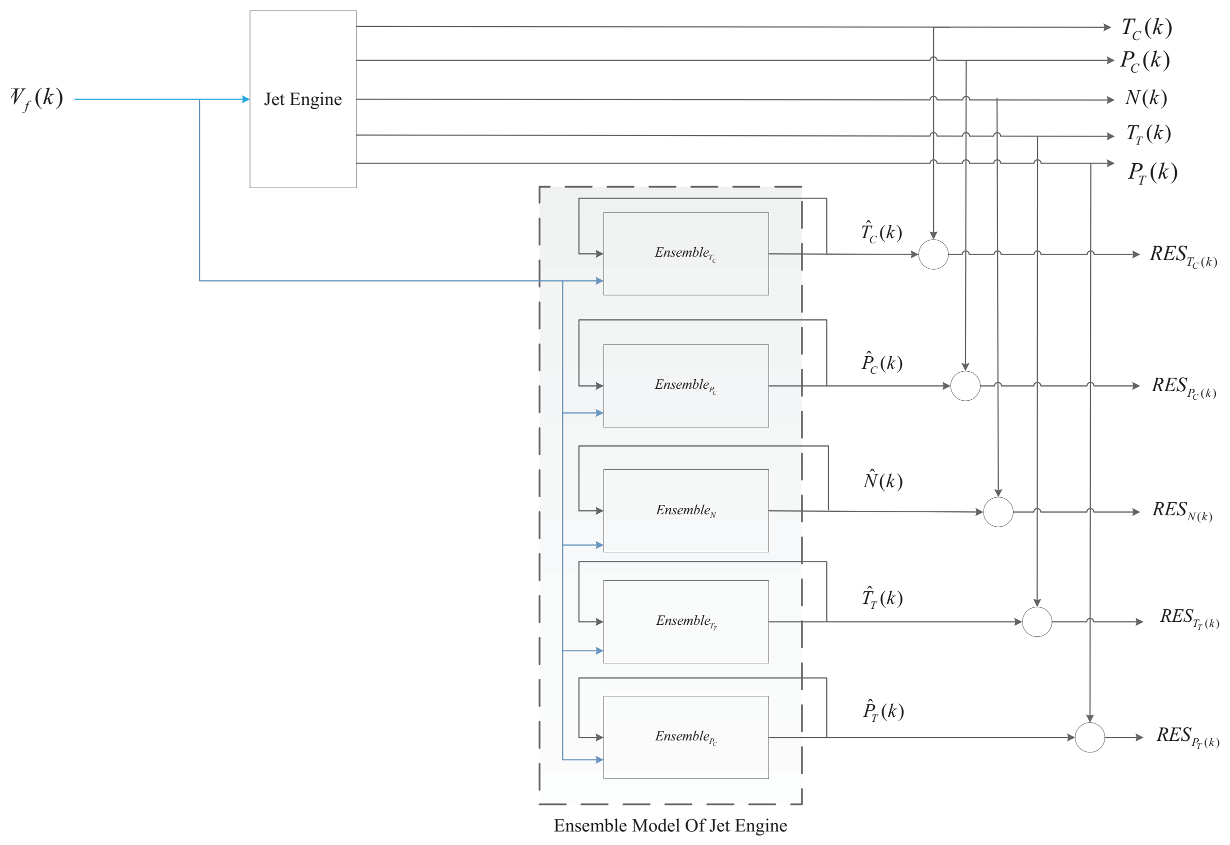


Figure 3.36: Architecture of the ensemble during testing phase.

Figure 3.37 shows inside of each ensemble model. Note that each model has its own parameters such as the number of neurons (in case of neural networks) or time delays.

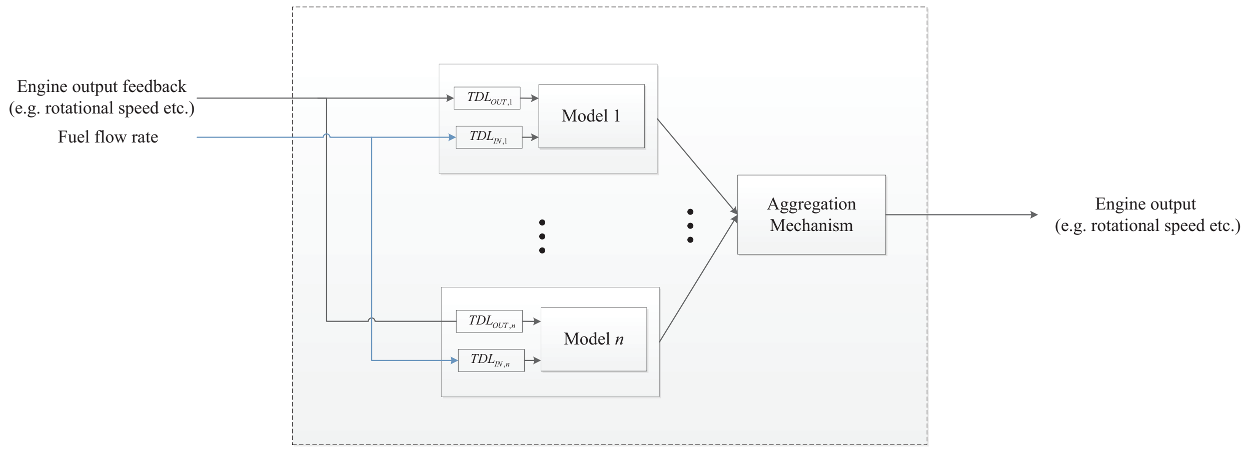


Figure 3.37: Inside of an ensemble model (refer to Figures 3.36and 3.35).

3.7.1 Ensemble Generation

Definition 3.1. Ensemble generation approaches are divided into *homogeneous* where all the models are generated using the same learning algorithm, and *heterogeneous* where different learning algorithms are used for training of ensemble members [133].

As previously discussed in Chapter 2, creating diverse set of learners is the key to successfully train an ensemble of regressors or classifiers. Intuitively, we know that if all ensemble learners provide the same output, there would be nothing to benefit from their combination. The importance of diversity for ensemble systems is well established in [193], [194]. Ideally, we would like the individual learners to be independent or even negatively correlated [121], [195]. Below, we describe two different approaches that can be adopted to create diversity among ensemble.

Homogeneous ensemble generation is the best covered area of ensemble learning in the literature [219]. In this approach, the ensemble members are generated using the same learning algorithm (e.g. the same kind of neural networks), and the diversity among them are to be generated by *altering the training data*. Alternatively, we may diversify the homogeneous models by using a same learning method with different setting of parameters (e.g. neural networks with different number of hidden units). Comparative studies between these two approaches conclude that altering training data is generally more effective than altering model parameters [221], [222]. Several approaches are suggested in the literature for training ensemble systems by manipulating the training data. Bagging (bootstrap aggregating) is the most extensively homogeneous ensemble method used in the literature [223]. In this approach the original training data is resampled by the bootstrap sampling in order to create several training set of the training data. The authors in [128] and [129] give insight about why bagging works.

Boosting is another algorithm which uses manipulation of training data to generate diversity. Similar to bagging, several training data sets are generated by resampling of the training data; however, unlink bagging the probability of being selected in not necessarily the same for different samples. In fact, the probability of being selected is initially equal for all the samples, but in the subsequent iterations samples with more inaccurate predictions would have higher chance of being selected. Boosting has been originally developed for classification problems. Although several modifications of it are proposed for regression but none of them has demonstrated as promising results as bagging [130].

In *heterogeneous* ensemble on the other hand the models are trained using the same training data. The number of works into using different architectures for ensemble systems is relatively small, and thus it requires more attention [223]. The diversity among the models is generated by different learning algorithms. This approach is studied less in the literature; however, some very good results are reported using heterogeneous ensembles [30]. The diversity in this approach is obtained by inherent properties of different learning algorithms. The problem is the lack of control on the diversity of the ensemble during the generation phase. This approach is discussed further in the following.

3.7.2 Ensemble Pruning

The ensemble pruning algorithms are previously discussed in Chapter 2 in details. Ensemble pruning is the procedure for trimming the pool of trained models, with the goal of improving generalization error of the ensemble. It is also used to reduce the complexity of the ensemble system. Several pruning methods are addressed in the literature. Three different pruning approaches are compared in [222]. The first approach is ranking based on the accuracy, second method is the forward selective search (FSS) algorithm, and the third using genetic algorithm. The authors test the above mentioned approaches and conclude that FSS gives the best result.

Roli *et al.* [201] conduct a benchmark in order to compare different pruning algorithms where they compare FSS by selecting the best model in the first iteration, backward selective search (BSS), and tabu search. They conclude FSS that selects the best model in the first iteration outperforms the other approaches.

Coelho *et al.* [134] compare FSS with ranking (FSSwR, starting with best members), FSS, BSS with Ranking (BSSwR) and BSS. Each one of these algorithms are tested with a different integration methodologies. The authors conclude that FSS and BSS gives higher diversity in general.

All of these benchmark studies address the ensemble classification problem; however the proposed approaches are general and can be applied to regression problems as well. It seems that more sophisticated algorithms such as FSS, BSS or clustering algorithms are able to give better results in term of accuracy, as expected when compared with more primitive algorithms discussed in Chapter 2 such as exponential pruning algorithms and randomized pruning algorithms.

3.7.3 Ensemble Combination

Ensemble integration is the last step in training an ensemble. Ensemble integration combines the predictions made by various models in the ensemble to generate the final ensemble prediction. For regression problems all integration mechanisms combine models using a linear combination of predictors which can be described by the following equation [223]:

$$f_{ensemble}(x) = \sum_{i}^{n} \alpha_i f_i(x)$$

where $f_{ensemble}(x)$ is the output of ensemble model for the instant x, $f_i(x)$ is the output of the i^{th} model for the instance x, α_i is the averaging weight for the i^{th} model, and n is the number of ensemble members.

In other words, the ensemble combination for a regression problem can be restated as estimating the averaging weights α_is. Merz *et al.* [224] conducted a comparative study in order to determine the most effective ensemble combination techniques. The authors studied several ensemble combination techniques including Generalized Ensemble Method (GEM) [192], Basic Ensemble Method (BEM) [192], Linear Regression (LR), Gradient Descent, and Exponential Gradient Descent [225]. For this purpose the authors conducted different comparative studies between the aforementioned combination methods using eight different data sets. The study concludes that optimizing the averaging weights using *gradient descent method* and *generalized ensemble method* results in a better generalization performance.

Remark 3.5. *We should note that the optimization of weights would be conducted on the training* *data. Assume that the members of the ensemble are trained separately using different training data set that is x_i, $i = 1, ..., n$ (i.e. training input vector of the i^{th} model) and t_i, $i = 1, ..., n$ (i.e. training target vector of the i^{th} model) where n is the number of models in the ensemble. Then the training data of the ensemble system would be the union of all the training data used for training of individual models. In other words:*

$$\begin{aligned} \textit{Training input vector: } x_{training} &= x_1 \quad x_2 \quad ... \quad x_n \\ \textit{Training target vector: } t_{training} &= t_1 \quad t_2 \quad ... \quad t_n \end{aligned}$$

Generalized Ensemble Method

Generalized ensemble method (GEM) is an ensemble combination technique which was first presented in [192]. In this method the averaging weights (α_is) are optimized using the method of Lagrange multipliers. The GEM estimator defines the $f_{GEM}(x)$, as follows:

$$f_{GEM}(x) = \sum_{i=1}^{n} \alpha_i f_i(x)$$

where $f_{GEM}(x)$ is the output of the ensemble model combined using GEM, $f_i(x)$ is the i^{th} model regression estimates, $f(x)$ is the actual estimation (i.e. target value), α_i is the averaging weight for the i^{th} model, n is the number of ensemble members. Defining $m_i(x) = f(x) - f_i(x)$ (i.e. the error of the i^{th} model) we have:

$$f_{GEM}(x) = \sum_{i=1}^{n} \alpha_i f_i(x) = \sum_{i=1}^{n} \alpha_i \; f(x) \quad m_i(x)$$

Assuming that $\sum_{i=1}^{n} \alpha_i = 1$ we have $\sum_{i=1}^{n} \alpha_i f(x) = f(x)$. Then we can write the above equation as follows:

$$f_{GEM}(x) = \sum_{i=1}^{n} \alpha_i f_i(x) = \sum_{i=1}^{n} \alpha_i \;\; f(x) \quad m_i(x) \;\; = f(x) \quad \sum_{i=1}^{n} \alpha_i m_i(x)$$

The ensemble absolute error is defined as:

$$f_{GEM}(x) - f(x) \; = \sum_{i=1}^{n} \alpha_i \, m_i(x)$$

Alternatively, the ensemble absolute error can be rewritten as:

$$f_{GEM}(x) - f(x) \; = \sum_{j=1}^{n} \alpha_j \, m_j(x)$$

There the mean of squared error of the ensemble can be formulated as:

$$\begin{aligned} MSE \;\; f_{GEM}(x) &= E \;\; f_{GEM}(x) - f(x)^{\,2} \\ &= E \; \sum_{i=1}^{n} \alpha_i \, m_i(x) \; \sum_{j=1}^{n} \alpha_j \, m_j(x) \\ &= \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j E \;\; m_i(x) \;\; m_j(x) \end{aligned}$$

where the expected value is taken over all instances of input vector x. Since we use the training data to determine the α_is then window size of the expected value would be size of training data. Defining the symmetric correlation matrix $C_{ij}(x) = E[\; m_i(x) \;\; m_j(x)\;]$ the above equation simplifies to the following equation:

$$MSE\ \ f_{GEM}(x)\ = \sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j C_{ij}(x)$$

The authors in [192] then formulate the following optimization problem with the objective of minimizing the MSE.

$$\begin{aligned} &\textit{minimize} && MSE\ \ f_{GEM}(x_{training})\ = \sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j C_{ij}(x_{training}) \\ &\textit{subject to} && \sum_{i=1}^{n}\alpha_i = 1 \end{aligned}$$

Using the method of Lagrange multipliers to solve for α_k we want α_k such that for $\ \ k$:

$$\frac{\partial}{\partial\alpha_k}[\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j C_{ij}(x_{training}) - 2\lambda(\sum_{i=1}^{n}\alpha_i)] = 0$$

Taking the partial derivatives the above equation simplifies to the following condition:

$$\sum_{k=1}^{n}\alpha_k C_{kj}(x_{training}) = \lambda, \quad k >\ \ 1, ..., n$$

Imposing the constraint $\sum_{i=1}^{n}\alpha_i = 1$ we can solve the set of equations and determine the optimal α_is as follows [192]:

$$\alpha_i = \sum_{j} C_{ij}(x_{training})^{-1}\ \ \sum_{k}\sum_{j} C_{kj}(x_{training})^{-1\ \ -1} \tag{3.3}$$

Gradient Decent Optimization

Another integration method which was suggested in [225], and [224] is the very well–known gradient decent optimization scheme. One advantage of this method is the possibility to adopt different objective functions for the optimization problem (i.e. minimizing RMSE, mean of error, etc.).

In this research both the *GEM* and *gradient descent* are adopted for combining individual models in an ensemble. Our experiments show that the gradient descent performs significantly better than *GEM*. The objective function which is minimized by using the gradient descent is RMSE of the ensemble as follows:

$$\min_{\alpha_i} \quad RMSE_{ensemble}(\alpha) = \sum_{i=1}^{n} \alpha_i f_i(x_{training}) - f(x_{training}) \tag{3.4}$$

$$\textit{Updating rule:} \quad \alpha_{k+1} = \alpha_k - \gamma \quad RMSE_{ensemble}(\alpha_k) \tag{3.5}$$

where $\alpha = [\alpha_1, ..., \alpha_n]$, $f_i(x_{training})$ is the prediction of the i^{th} model for the training data set, $f(x_{training})$ is the target value of the training data, α_k is the value of α at the k^{th} iteration, α_{k+1} is the value of α at the $k \quad 1^{th}$ iteration, $\quad RMSE_{ensemble}(\alpha_k)$ is the gradient of $RMSE_{ensemble}$ at the k^{th} iteration, and γ is the step size. The step size should be carefully selected as with a too large step size the gradient descent may diverge, and with a too small step size it may take a long time to converge. Note that the step size can be either fixed or adaptive (i.e. changing at each iteration). Adaptive step size is initialized with a bigger step size and it becomes smaller as the gradient of the function gets smaller. After trying different step sizes, we picked $\gamma = 0.1$ with which the optimization problem converges within a reasonable time.

Remark 3.6. *In order to have an ensemble whose performance is better than all the individual* *models (or at least as well as the best model in the pool), the weights of the gradient descent would be initialized as follows:*

$$\alpha_i = \begin{cases} 1 & \textit{if model } \mathrm{i} \textit{ is the best model in the pool} \\ 0 & \textit{otherwise} \end{cases}$$

This way if the gradient descent gets trapped in a local minimum, we are guaranteed that the result is at least as good as the best member of the ensemble. In other words, the RMSE with 100% contribution of the best ensemble member is either a local minimum of the error function or, if not, we can ˇnd another point where the error function is smaller.

In the next subsection we present the simulation results for different ensemble techniques applied for the jet engine modeling problem.

3.7.4 Ensemble I: Heterogeneous Ensemble with Ranked Pruning

Heterogeneous ensemble with ranked pruning is reported in several papers in the literature including but not limited to [222], [132], and [133]. In this approach first a pool of individual learners are trained using different learning algorithms. Then, the most accurate models are selected for each learning algorithm in order to be aggregated and generate the final prediction. The only source of diversity in this approach is the use of heterogeneous ensemble (using different kinds of learning algorithms).

In this research the above procedure is used to identify the jet engine dynamics. First, in the previous section, several learners are trained using each of MLP-NARX, RBF-NARX, and SVM-NARX models to identify jet engine dynamics. Second, for each learning algorithm (e.g. MLP-NARX) the regressor with *best performance* is selected from the pool of individual trained regressors. Then the selected regressors are combined together by using the weighted averaging. Two different combination techniques are used to determine optimal averaging weights: gradient descent and generalized ensemble method.

According to equation (3.3) the α_is required for integrating the three individual learners are determined as follows:

$$\begin{aligned}
\alpha_1 &= \frac{\sum_{j=1}^{3} C_{1j}^{-1}}{\sum_{k=1}^{3}\sum_{j=1}^{3} C_{kj}^{-1}} \\
\alpha_2 &= \frac{\sum_{j=1}^{3} C_{2j}^{-1}}{\sum_{k=1}^{3}\sum_{j=1}^{3} C_{kj}^{-1}} \\
\alpha_3 &= \frac{\sum_{j=1}^{3} C_{3j}^{-1}}{\sum_{k=1}^{3}\sum_{j=1}^{3} C_{kj}^{-1}}
\end{aligned}$$

The averaging weights obtained by using GEM are presented in Table 3.39.

Table 3.39: GEM coefficients for integration of ensemble system (ensemble I)

	α_{MLP}	α_{RBF}	α_{SVM}
P_C	0.23	0.455	0.312
T_C	0.232	0.455	0.312
N	0.232	0.455	0.312
P_T	0.23	0.455	0.313
T_T	0.231	0.454	0.314

Alternatively, gradient descent is used to determine the optimal weights for combining the ensemble models. The optimization problem is formulated as follows:

$$\min_{\alpha_i} \quad RMSE_{ensemble}(\alpha) = \sum_{i=1}^{n} \alpha_i f_i(x_{training}) - f(x_{training})$$

$$\textit{Updating rule:} \quad \alpha_{k+1} = \alpha_k - 0.1 \qquad RMSE_{ensemble}(\alpha_k)$$

where $\alpha = [\alpha_1, \alpha_2, \alpha_3] = [\alpha_{RBF}, \alpha_{MLP}, \alpha_{SVM}]$. The α is initialized as $\alpha_0 = [1, 0, 0]$. After trying different step sizes, we picked a fixed step size $\gamma = 0.1$ with which the optimization problem converges within a reasonable time. Note that the fixed step size will assure convergence, if it is small enough (although it may take a long time to converge if the step size is too small). The optimized weights are presented in Table 3.40.

Table 3.40: Gradient descent coefficients for integration of ensemble system (ensemble I)

	α_{MLP}	α_{RBF}	α_{SVM}
P_C	$9.1313 \quad 10^{-5}$	1.001	$9.1266 \quad 10^{-5}$
T_C	0.0234	0.738	0.238
N	-0.238	1.854	-0.853
P_T	0.020	0.975	0.0049
T_T	0.272	0.785	0.053

The performance of *heterogeneous* ensemble with *ranked pruning* and *GEM* as integration method is summarized in Table 3.41.

Table 3.41: Heterogeneous ensemble with ranked pruning and GEM as integration method error analysis.

	$RMSE$	μ_{ae}	σ_{ae}
P_C	1.0043	0.8511	0.5332
T_C	0.0286	0.0219	0.0184
T_T	20.0723	16.7455	11.0693
PT_T	0.1093	0.0914	0.0599
N	39.7872	35.9183	17.1171

The performance of *heterogeneous* ensemble with *ranked pruning* and *gradient descent* as integration method is summarized in Table 3.42.

Table 3.42: Heterogeneous ensemble with ranked pruning and gradient descent as integration method error analysis.

	$RMSE$	μ_{ae}	σ_{ae}
P_C	0.536	0.4294	0.3212
T_C	0.0246	0.0208	0.0133
T_T	11.073	8.6915	6.862
P_T	0.0179	0.0145	0.0105
N	7.339	4.229	6.0014

According to the simulations the gradient descent has way better results in term of generalization error as compared with GEM. Further comparative studies of the obtained results with the other designed ensembles systems as well as individual learners are presented in the following sections. Table 3.43 summarizes heterogeneous ensemble training with ranked pruning.

Table 3.43: Summary of the heterogeneous ensemble training with ranked pruning.

1:	Several models are trained using MLP-NARX, RBF-NARX, and SVM-NARX modeling engine parameters.
2:	For each algorithm the best trained model is selected to be combined.
3:	GEM and gradient decent algorithms are used to determine averaging weights.
4:	The *initial point* of gradient decent algorithm is initialized such that the best learner in the pool has the maximum contribution.

3.7.5 Ensemble II: Heterogeneous Ensemble using Forward Sequential Selection

The use of heterogeneous ensemble with Forward Sequential Selection (FSS) as pruning algorithm is reported in several publications including but not limited to [201], [192], [134]. Forward selection starts with an empty set and iteratively adds models with the aim of decreasing the expected prediction error. Two different versions of FSS are presented in the literature, namely Forward Sequential Selection with Ranking (FSSwR) and Forward Sequential Selection (FSS). The FSSwR ranks all the candidates with respect to their performance on a training set. Then, it selects the candidate at the top until the performance of the ensemble decreases. In the FSS algorithm, each time a new candidate is added to the ensemble, all candidates are tested and it is selected the one that leads to the maximal improvement of the ensemble performance. Yates *et al.* [220] modify the

FSS by adding a diversity measure. In this version the criterion for the inclusion of a new model is a diversity measure, and that the new model is diverse from the previously selected models. The ensemble size is an input parameter of the algorithm.

In this research a heterogeneous ensemble with FSS as pruning algorithm is used to identify the jet engine dynamics. First, as presented at the beginning of this chapter, several models are trained using each of MLP-NARX, RBF-NARX, and SVM-NARX to model different parameters of the jet engine dynamics. To limit the complexity of the problem, a subset of the trained models is selected based on their performance (i.e. we select the 10 best RBF-NARX models, the 10 best MLP-NARX models and the 10 best SVM-NARX models). Then the members of the ensemble system are selected using the FSS algorithm. The FSS algorithm is initialized using the model with the best performance in the pool. Each time a new model is added to the ensemble, all candidates are tested and the model with maximal improvement would be added as the next model. In each iteration all the selected models are aggregated using gradient descent. Note that GEM is not employed in this section due to its poor performance in the previous section. Table 3.44 summarizes the procedure of construction of the ensemble system.

Table 3.44: Summary of the heterogeneous ensemble training with FSS pruning algorithm.

1:	Several models are trained using MLP-NARX, RBF-NARX, and SVM-NARX modeling engine parameters.
2:	Subsets of 10 best RBF-NARX models, the 10 best MLP-NARX models and the 10 best SVM-NARX models are selected from the pool of models trained in step 1 ***in order to reduce complexity***.
3:	For each engine parameter (i.e. P_C, T_C, N, P_T, T_T), FSS algorithm is initialized with the best trained model.
4:	Each time a new model is added to the ensemble, all candidates are tested and the model with maximal improvement would be added as next model.
5:	Each time a new model is added to the ensemble the optimal combining weights are recalculated using gradient descent algorithm as discussed in Section 3.7.3, page 148, equation 3.4.
6:	All the evaluations for FSS are performed on the training set.

In this section we train a heterogeneous ensemble MLP-NARX, RBF-NARX, and SVM-NARX models with FSS pruning. Therefore, for each engine parameter (e.g. rotational speed) we initialize the ensemble with the best individually trained model. Since in our application the RBF-NARX model shows a better performance (refer to the previous sections to see the performance of the individual models), we initialize the ensemble with the best RBF-NARX model of each engine parameter. Next we select the MLP-NARX model (or alternatively the SVM-NARX model) so that the maximal improvement to the performance of the ensemble is achieved. Note that to find the maximal improvement we have to test all the candidate models in the pool. Finally, we select the SVM-NARX (or alternatively the MLP-NARX model) which brings the maximal improvement to the ensemble previous two models.

Using the FSS algorithm we selected (i.e. pruning state) the following models as individual model of the ensemble (refer to Tables 3.45 to 3.49).

Table 3.45: Parameters of models inside ensemble model of compressor pressure (ensemble II).

	# training samples	# delays	# of neurons
MLP-NARX	1501	7	11
RBF-NARX	1801	7	10
SVM-NARX	1501	5	NA

Table 3.46: Parameters of models inside ensemble model of compressor temperature (ensemble II).

	# training samples	# delays	# of neurons
MLP-NARX	1801	5	11
RBF-NARX	1201	7	20
SVM-NARX	1201	6	NA

Table 3.47: Parameters of models inside ensemble model of rotational speed (ensemble II).

	# training samples	# delays	# of neurons
MLP-NARX	1201	6	10
RBF-NARX	1501	8	11
SVM-NARX	1201	6	NA

Table 3.48: Parameters of models inside ensemble model of turbine pressure (ensemble II).

	# training samples	# delays	# of neurons
MLP-NARX	1201	6	13
RBF-NARX	1201	7	10
SVM-NARX	1201	6	NA

Table 3.49: Parameters of models inside ensemble model of turbine temperature (ensemble II).

	# training samples	# delays	# of neurons
MLP-NARX	1801	5	10
RBF-NARX	1501	8	17
SVM-NARX	1501	4	NA

Gradient descent is used to minimize the RMSE by optimizing the averaging weights of models. The α_is required for integrating the three individual learners are determined as presented in Table 3.50.

Table 3.50: Gradient descent coefficients for integration of ensemble system (ensemble II).

	α_{MLP}	α_{RBF}	α_{SVM}
P_C	$9.5212 \quad 10^{-5}$	1.0001	$9.4425 \quad 10^{-5}$
T_C	0.0646	0.8164	0.1190
N	0.0685	1.7562	-0.8261
P_T	0.0048	0.8974	0.109
T_T	-0.0995	1.086	0.0131

The summary of the ensemble system performance for identification of each engine parameter is presented in Table 3.51.

Table 3.51: Heterogeneous ensemble with the FSS pruning, and gradient descent as integration method error analysis.

	$RMSE$	μ_{ae}	σ_{ae}
P_C	0.023135	0.023113	0.00101
T_C	0.5049	0.3883	0.1045
T_T	10.139	7.848	3.820
P_T	0.016865	0.016841	0.000896
N	6.8672	4.563	3.654

A comparative study of the obtained results with the other designed ensembles systems and each of the individual learners are presented in the following sections.

3.7.6 Ensemble III: Homogeneous with Bagging

Bootstrap sampling or *bagging* is one of the most extensively used techniques for manipulation of training data [223]. Empirical studies show that bagging is a simple and effective method in reducing prediction error in both classification and regression problems [129]. The main idea is to train a different model using different subsets of the training data which are generated by bootstrap sampling. In the bagging procedure, a training set with the size s, several bootstrap replicates of it would be constructed by taking s samples out of it *with replacement*. Thus, a new training set with the same size would be generated where each of the samples in the original training set may appear once, more than once or may not appear at all [129]. The learning algorithm then uses this new training set. This procedure would be repeated several times, and all the obtained models would be aggregated to generate the final ensemble. Figure 3.38 graphically summarizes the bagging procedure. Several studies have been conducted on the theoretical basis of bagging establishing its theoretical foundation. The theoretical background about bagging and effectiveness can be found in [128], [129], [119].

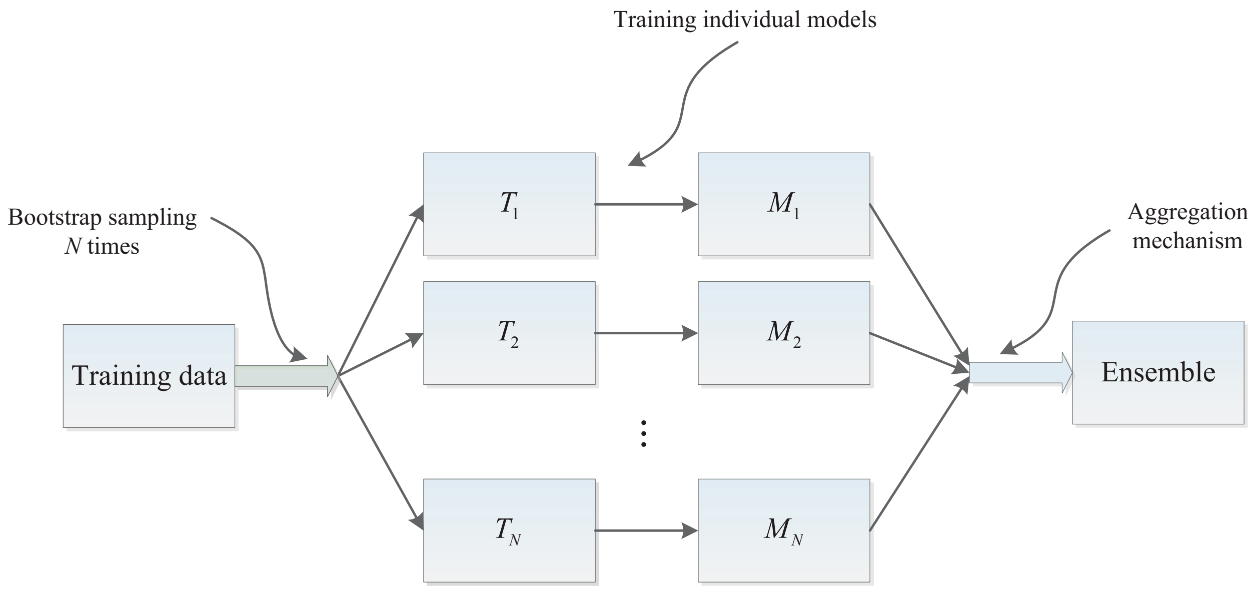


Figure 3.38: Ensemble learning with bagging.

In this section a homogeneous ensemble is trained using bagging for modeling each of the jet engine outputs. In order to select the training algorithm, and parameters of it we refer to the experiments conducted in the previous sections. According to Sections 3.4, 3.5, 3.6, the RBF-NARX model outperform MLP-NARX and SVM-NARX models for modeling engine outputs. Thus, in this section we use RBF-NARX to form our homogeneous ensemble. For the network parameters (i.e. number of neurons, number of delays, size of training data) we set them to the parameters of the models with best performance (in term of RMSE) in Section 3.5. *As an example*, in Section 3.5 we saw that the RBF-NARX model with the following parameters has the best performance for modeling compressor pressure of the jet engine:

Table 3.52: Best SVM-NARX for modeling the compressor temperature in terms of $RMSE_{test}$.

# delays	# training samples	RMSE$^{T_C}_{total}$	RMSE$^{T_C}_{training}$	RMSE$^{T_C}_{test}$	%RMSE$^{T_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
7	**1200**	**2.6112**	**2.7304**	**2.5287**	**0.39709**	**2.0356**	**1.6357**

Thus, in this section we train several RBF-NARX models with exactly the same parameters. The only thing which alters is the training data where for different models of the homogeneous models the training data is obtained by bootstrap sampling as mentioned above. As previously mentioned in Chapter 2 the number of models in an ensemble plays an important in its performance [187], [186]. Thus, in order to have a faire comparison between the heterogenous ensembles discussed before (ensemble I and ensemble II) and the homogeneous ensemble (ensemble III) discussed in this section we select the same number of models for all of them. We study the effects of the number of models in ensemble in the next section. As in the previous sections previous sections a weighted averaging is used to combine models of the ensemble. The weights are optimized using the gradient descent with the *objective of minimizing RMSE on training data*.

The obtained results are reflected in Table 3.53.

Table 3.53: Heterogeneous ensemble with FSS pruning, and gradient descent as integration method error analysis.

	$RMSE$	μ_{ae}	σ_{ae}
P_C	0.0316	0.0304	0.0315
T_C	0.6754	0.7737	0.6711
T_T	10.139	7.848	3.820
P_T	8.686	9.386	8.637
N	9.111	7.262	9.083

We study the effects of the number of models in the ensemble in the next section.

3.7.7 Effects of the Number of Models in an Ensemble

As previously mentioned the number of models in ensemble plays an important role in the accuracy of the ensemble. According to [187], [186] and several other references the error can be theoretically decreased arbitrarily by increasing the number of models in the ensemble. In this section we would like to validate this claim in our experimental setup.

For this purpose, we start increasing the number of models in the ensemble models of the jet engine outputs. Homogeneous ensemble with bagging is used for this purpose, as it is easy to generate arbitrary number of models in this approach just by resampling the training data. Figures 3.39 to 3.43 show the performance of bagged ensembles for identification of different outputs of a jet engine. The ensemble error generally decreases by increasing the number of models in an ensemble.

It should be noted that each iteration is totaly independent from the previous ones. Suppose that a bagged ensemble containing n models is trained in the i^{th} iteration, then for the step after (i.e. $i \quad 1^{th}$ step) $n \quad 1$ models are trained and aggregated independently from the models trained in the previous step. An alternative would be to keep all previously trained models and add a model to the ensemble in each iteration.

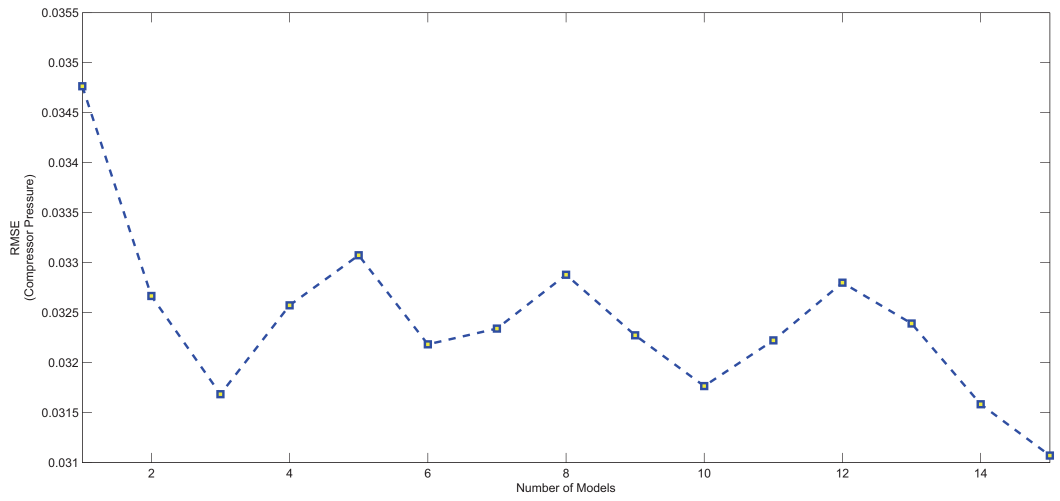


Figure 3.39: RMS error for bagged model of the compressor pressure with respect to the number of models in the ensemble.

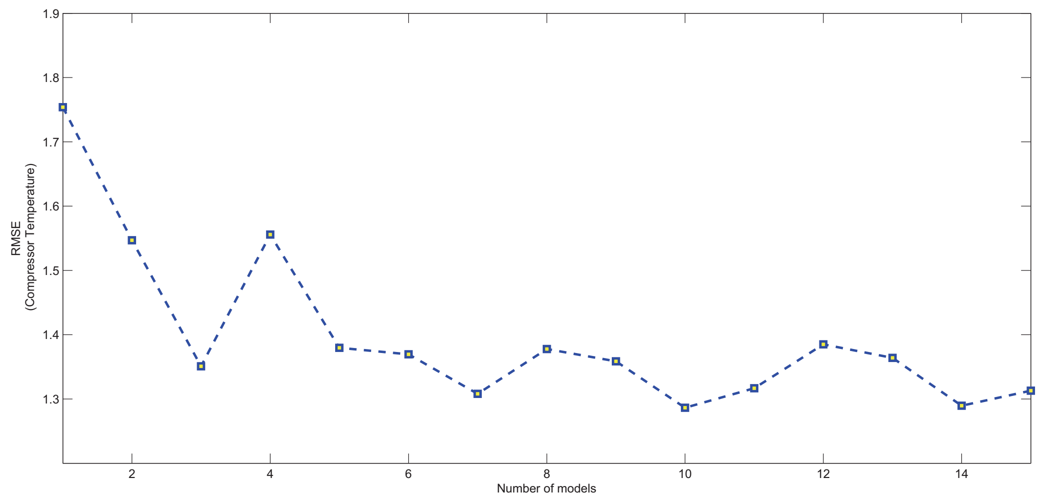


Figure 3.40: RMS error for bagged model of the compressor temperature with respect to the number of models in the ensemble.

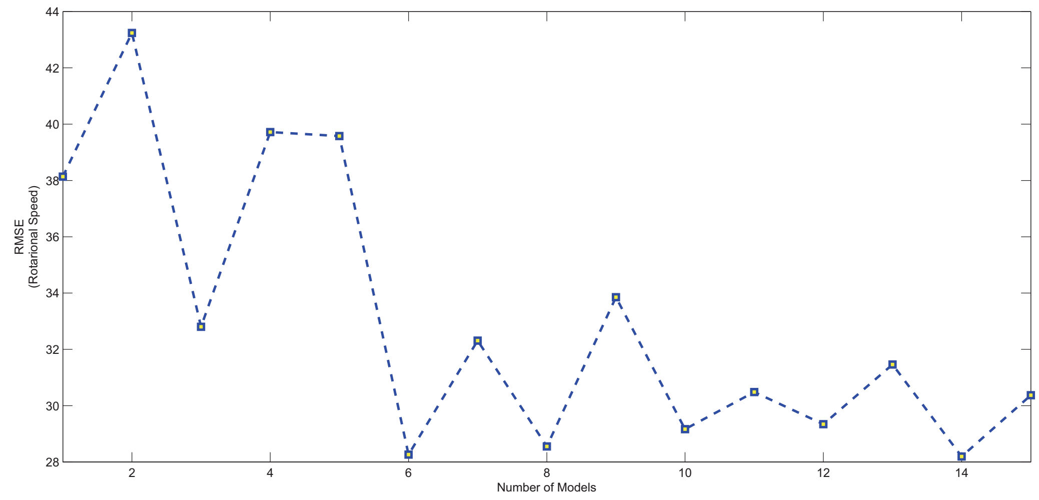


Figure 3.41: RMS error for bagged model of the rotational speed with respect to the number of models in the ensemble.

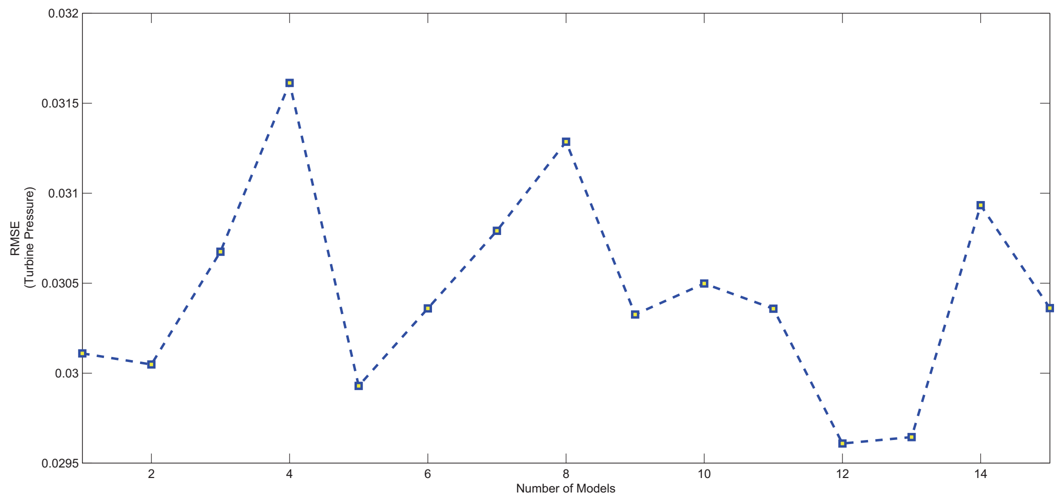


Figure 3.42: RMS error for bagged model of the turbine pressure with respect to the number of models in the ensemble.

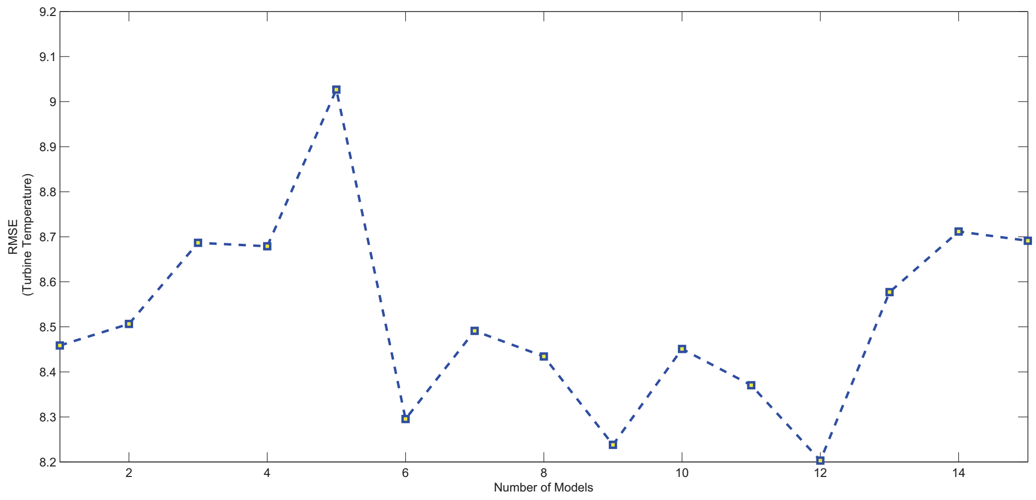


Figure 3.43: RMS error for bagged model of the turbine temperature with respect to the number of models in the ensemble.

3.7.8 Summary

In this chapter several models are developed to model dynamics of the jet engine. Three stand-alone (MLP-NARX, RBF-NARX, and SVM-NARX) learning algorithms are first used to model each engine output. Then the individual models are used to create ensemble systems. Different architectures of ensembles are trained and compared together and with the stand-alone models. We observe that by combining stand-alone models and building an ensemble system one can *always* build a regressor which has a better performance (or at least as well as the best model in the pool).

Table 3.54 shows comparison of different methods for modeling the compressor pressure. The error analysis is also shown in Figure 3.44. One can see that the heterogeneous ensemble with FSS pruning has a better performance in modeling of the compressor pressure.

Remark 3.7. $RMSE_{training}$, $RSME_{test}$ *and* $RSME_{total}$ *are the* root mean square error *calculated over the "training", "testing" and "all the available data" (training and testing) respectively. Note that the training data includes the samples which are exposed to the networks during the training stage. This covers the data which directly used for training as well as the cross-validation data which is indirectly used during the training stage.*

Table 3.54: Comparison of different methods for identification of compressor pressure.

	$RMSE_{total}$	$RMSE_{train}$	$RMSE_{test}$	μ_{ae}	σ_{ae}
MLP-NARX	0.0531	0.060215	0.040119	0.033786	0.040972
RBF-NARX	0.026612	0.027382	0.026087	0.021582	0.015573
SVM-NARX	0.041185	0.037001	0.046766	0.028081	0.03013
Ensemble I	0.024672	0.024926	0.024285	0.02894	0.02103
Ensemble II	0.023135	0.023115	0.023166	0.023113	0.0010121
Ensemble III	0.031684	0.027609	0.042234	0.030476	0.031543

	RMSE_total	RMSE_train	RMSE_test	%RMSE	mean	sigma
MLP-NARX	0.05695994	0.06459238	0.0430354	0.481906	0.036242	0.04395
RBF-NARX	0.03948843	0.04063023	0.038709	0.328007	0.032025	0.023107
SVM-NARX	0.06804089	0.06112859	0.0772607	0.564795	0.046393	0.049776
Ensemble I	0.03099947	0.03420156	0.0274235	0.265349	0.023378	0.020361
Ensemble II	0.032391					
Ensemble III	0.03107098					

improveme

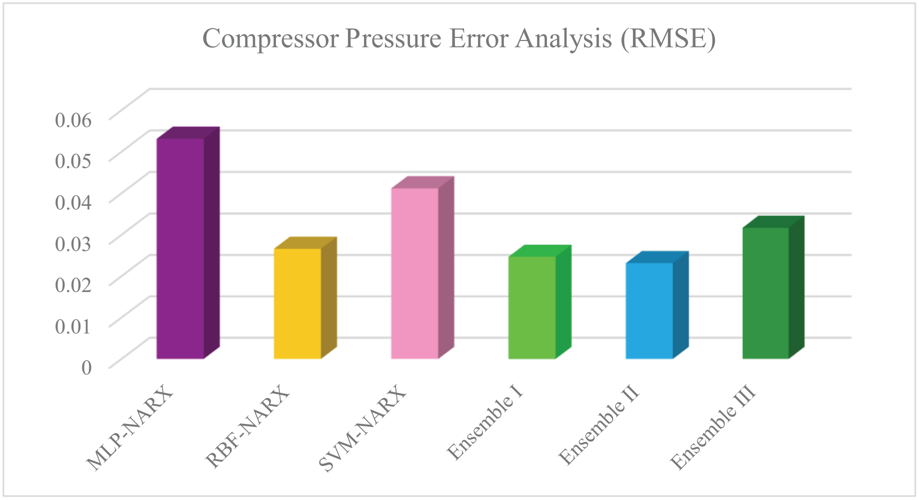


Figure 3.44: Error analysis: compressor pressure identification.

Table 3.55 shows comparison of different methods for modeling the compressor pressure. The error analysis is also shown in Figure 3.45. One can see that the heterogeneous ensemble with FSS pruning has a better performance in modeling compressor pressure.

Table 3.55: Comparison of different methods for identification of compressor temperature.

	$RMSE_{total}$	$RMSE_{train}$	$RMSE_{test}$	μ_{ae}	σ_{ae}
MLP-NARX	1.0016	1.0817	0.91441	0.73773	0.67756
RBF-NARX	0.56069	0.60188	0.49245	0.44584	0.34007
SVM-NARX	2.6112	2.7304	2.5287	2.0356	1.6357
Ensemble I	0.53621	0.57179	0.47787	0.4294	0.1032
Ensemble II	0.50493	0.47823	0.5303	0.38835	0.10459
Ensemble III	0.67545	0.75346	0.64762	0.77372	0.67111

	RMSE_total	RMSE_train	RMSE_test	%RMSE	mean	sigma
MLP-NARX	1.5339092	1.65661695	1.4003979	0.233214	1.129811	1.037672
RBF-NARX	0.82816977	0.88900992	0.7273705	0.125567	0.658522	0.502293
SVM-NARX	3.26100159	3.40977775	3.157955	0.495902	2.542115	2.042665
Ensemble I	1.3152219	1.29057196	1.3394343	0.199706	1.141432	0.653516
Ensemble II						
Ensemble III	1.2864					

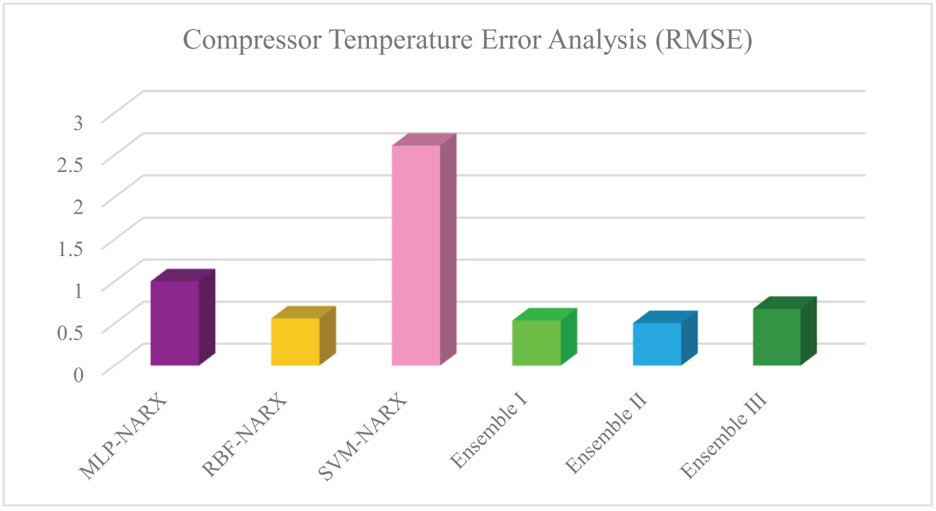


Figure 3.45: Error analysis: compressor temperature identification.

Table 3.56 shows comparison of different methods for modeling the compressor pressure. The error analysis is also shown in Figure 3.46. One can see that the heterogeneous ensemble with FSS pruning has a better performance in modeling the compressor pressure.

Table 3.56: Comparison of different methods for identification of rotational speed.

	$RMSE_{total}$	$RMSE_{train}$	$RMSE_{test}$	μ_{ae}	σ_{ae}
MLP-NARX	22.7076	24.7451	21.2417	19.597	11.4732
RBF-NARX	17.8349	19.7104	15.7359	14.2826	10.6831
SVM-NARX	24.6728	26.5164	22.6791	21.236	12.562
Ensemble I	7.3392	8.575	4.9343	4.2259	3.5032
Ensemble II	6.8672	6.7947	6.9391	4.5636	3.6543
Ensemble III	9.1115	8.0348	12.0596	7.2627	9.084

	RMSE_total	RMSE_train	RMSE_test	%RMSE	mean	sigma
MLP-NARX	56.8288983	61.9281767	53.160426	0.478681	49.04432	28.71329
RBF-NARX	44.8280081	49.5421607	39.552223	0.378495	35.89936	26.85193
SVM-NARX	93.0503374	100.003348	85.53136	0.785621	80.08902	47.37603
Ensemble I	51.1684386	53.5649498	38.461508	0.178457	37.69918	31.61408
Ensemble II	36.2141536					
Ensemble III	28.1931					

improveme

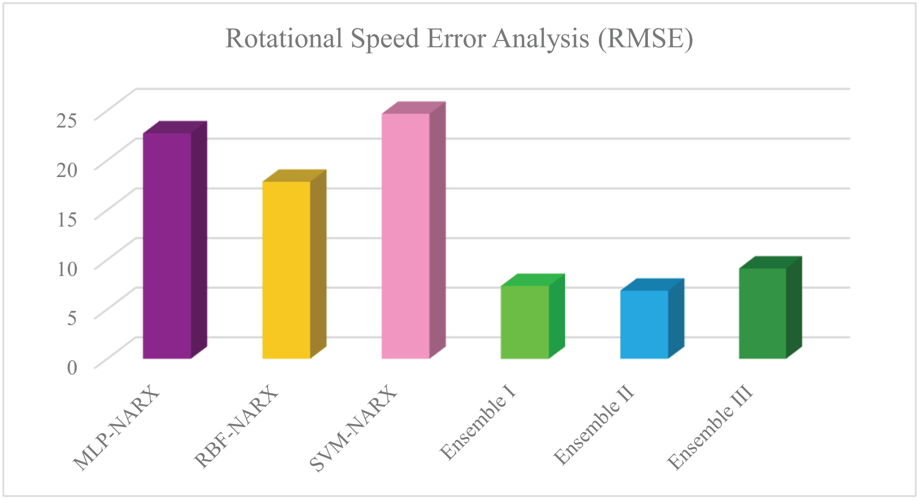


Figure 3.46: Error analysis: rotational speed identification.

Table 3.57 shows comparison of different methods for modeling the compressor pressure. The error analysis is also shown in Figure 3.47. One can see that the heterogeneous ensemble with FSS pruning has a better performance in modeling compressor pressure.

Table 3.57: Comparison of different methods for identification of turbine pressure.

	$RMSE_{total}$	$RMSE_{train}$	$RMSE_{test}$	μ_{ae}	σ_{ae}
MLP-NARX	0.28118	0.33468	0.21468	0.1725	0.22208
RBF-NARX	0.018406	0.018638	0.018172	0.015086	0.010548
SVM-NARX	0.088902	0.084001	0.095787	0.075015	0.047715
Ensemble I	0.01791	0.018221	0.017432	0.014507	0.00011
Ensemble II	0.016865	0.016835	0.016894	0.016841	0.0008965
Ensemble III	0.02045	0.023039	0.019989	0.020046	0.020861

	RMSE_total	RMSE_train	RMSE_test	%RMSE	mean	sigma
MLP-NARX	0.07101077	0.08452287	0.0542175	1.798008	0.043566	0.056086
RBF-NARX	0.03734914	0.03781917	0.0368728	0.777323	0.030611	0.021403
SVM-NARX	0.08327656	0.07868526	0.0897258	1.756486	0.070268	0.044695
Ensemble I	0.06410522	0.05560946	0.066296	1.319943	0.117578	0.077931
Ensemble II	0.03098247					
Ensemble III	0.03261					

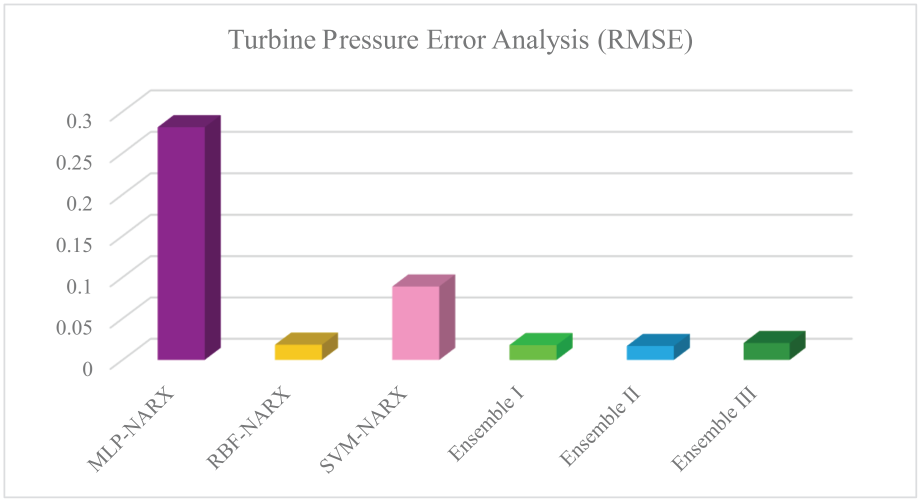


Figure 3.47: Error analysis: turbine pressure identification.

Table 3.58 shows comparison of different methods for modeling the compressor pressure. The error analysis is also shown in Figure 3.48. One can see that the heterogeneous ensemble with FSS pruning has a better performance in modeling of the compressor pressure.

Table 3.58: Comparison of different methods for identification of turbine temperature.

	$RMSE_{total}$	$RMSE_{train}$	$RMSE_{test}$	μ_{ae}	σ_{ae}
MLP-NARX	41.9615	40.4995	43.3752	37.1441	19.5246
RBF-NARX	13.4734	15.2044	12.1843	9.8925	9.1487
SVM-NARX	104.3983	107.0395	102.6002	90.6818	51.7326
Ensemble I	11.0734	11.1823	10.9079	8.6915	6.8625
Ensemble II	10.1397	9.8846	10.3887	7.8483	3.821
Ensemble III	8.6865	8.2657	9.7165	9.3865	8.6371

	RMSE_total	RMSE_train	RMSE_test	%RMSE	mean	sigma
MLP-NARX	20.9592222	20.2289664	21.66534002	1.15729486	18.5529861	9.752301928
RBF-NARX	1.33378387	1.50513759	1.206172871	0.07450583	0.97930077	0.905661486
SVM-NARX	52.1645898	53.48433	51.26614367	2.95170614	45.3108776	25.84918062
Ensemble I	34.6805544	31.6187933	37.5816222	2.06100145	29.1721054	13.92190986
Ensemble II	1.33378387	1.50513759	1.206172871	0.07450583	0.97930077	0.905661486
Ensemble III	8.203					

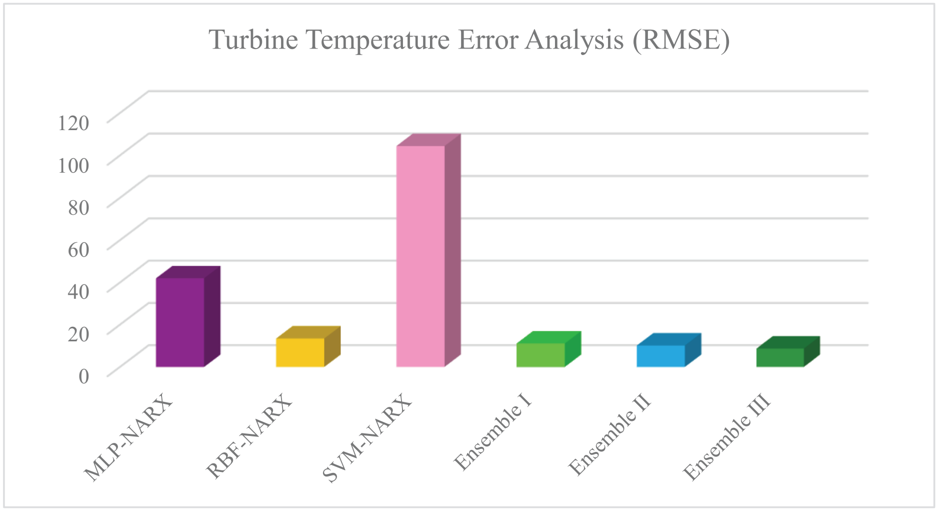


Figure 3.48: Error analysis: turbine temperature identification.

In summary one can see that the heterogeneous ensemble with FSS pruning and gradient descent combination (ensemble II) has better performance in modeling and identification of the jet engine dynamics. In the rest of this thesis we use this model (heterogeneous ensemble with FSS) for residual generation. Then generated residuals would be evaluated for the FDI purpose. The results are compared with RBF-NARX model as it has the best performance among the stand-alone models which are trained in this section. In this section we showed how utilization of ensemble methods can improve the accuracy of modeling in comparison with stand-alone algorithms. Below we show the effects of this improvement in the FDI application.

3.8 Fault Detection Process

In the previous sections we modeled the jet engine dynamics using both ensemble-based and stand-alone models. In this section we would like to use the trained models to generate the residual signals while the engine is operating in different conditions (healthy as well as different faulty conditions). The faulty conditions were previously explained in Section 2.3.3. Table 3.59 summarizes the jet engine degradations that are studied in the remainder of this thesis.

Table 3.59: Jet engine component fault indications.

Component Fault	Indication	Symbol
Compressor fouling	Decrease in the compressor flow capacity ($\dot{m}_C$)	F_{mc}
Compressor erosion	Decrease in the compressor efficiency (η_C)	F_{ec}
Turbine fouling	Decrease in the turbine flow capacity ($\dot{m}_T$)	F_{mt}
Turbine erosion	Decrease in the turbine efficiency (η_C)	F_{et}

3.8.1 Fault Detection Logic

This section explains the fault detection logic that is used in this thesis, noting that once the residual signals are generated then the fault detection would be a relatively easy task. The first required step in fault detection process is to gather the residual signals while the engine is working under healthy condition. These signals (residual signals under healthy condition) would be used to determine the fault detection thresholds, so that one determines an interval such that the residuals of a healthy jet engine stay within it with a high confidence (probability). A fault would be detected if any of the engine residuals goes beyond its threshold. The fault detection logic is summarized in Figure 3.49.

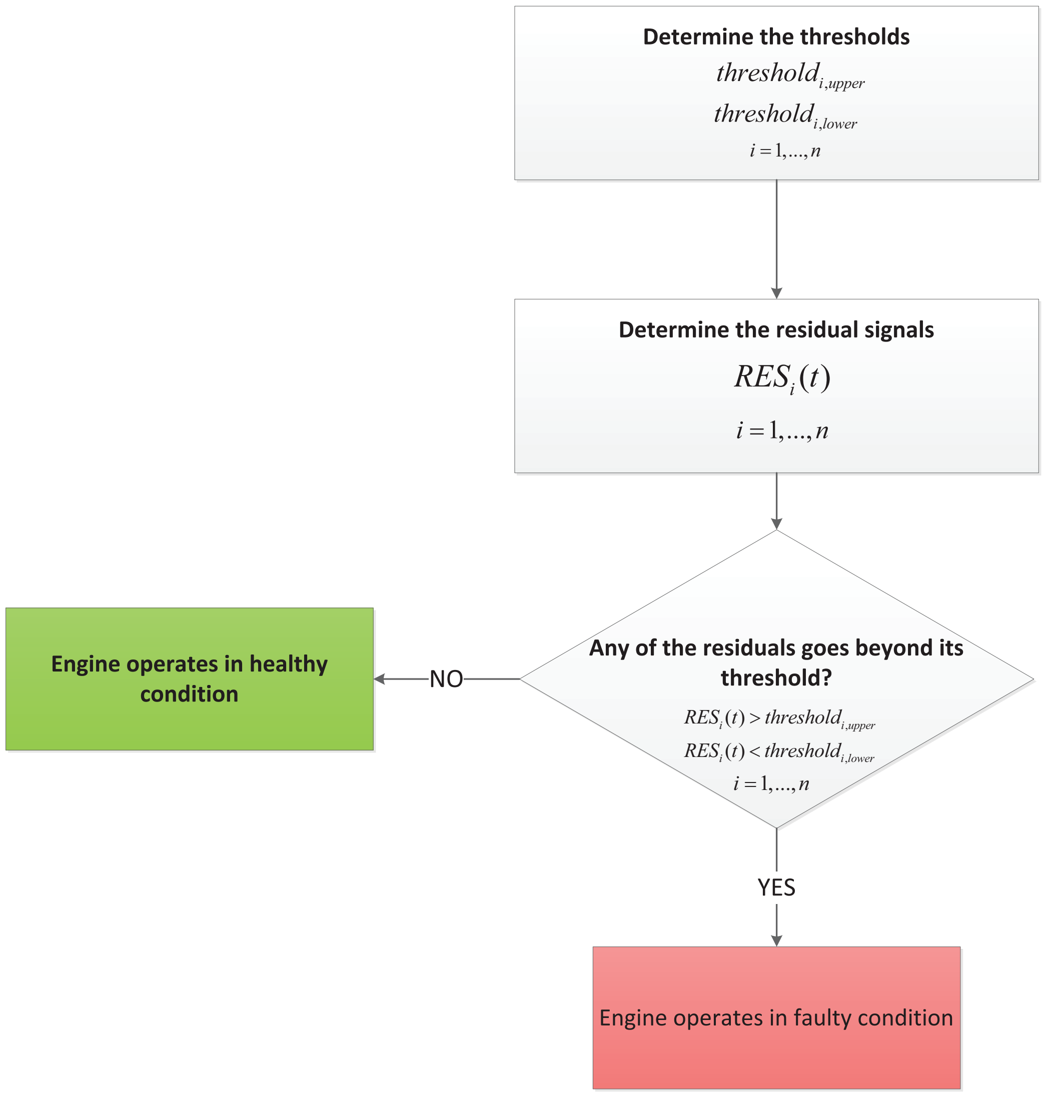

Determine the thresholds
$threshold_{i,upper}$
$threshold_{i,lower}$
$i = 1,\ldots,n$
Determine the residual signals
$RES_i(t)$
$i = 1,\ldots,n$
Any of the residuals goes beyond its threshold?
$RES_i(t) > threshold_{i,upper}$
$RES_i(t) < threshold_{i,lower}$
$i = 1,\ldots,n$
NO
Engine operates in healthy condition
YES
Engine operates in faulty condition

Figure 3.49: Flow chart of the proposed fault detection algorithm.

3.8.2 Fault Detection Threshold Generation

In the previous section we trained individual learning methods and the ensemble system using healthy engine data. From now on, the goal is to use the trained models to detect engine faults by comparing the residual signals with predefined thresholds. As explained before, residual signals are the difference between actual engine outputs with predicted values obtained from trained models. In this comparison, a residual signal should remain below a predefined threshold in the healthy condition of jet engine, while it exceeds the defined threshold under the faulty condition. The thresholds can be determined using the residual signals which are obtained from a healthy engine. In this thesis, we define the thresholds as follows:

$$\begin{aligned} t.h._{upper} &= \mu \quad z\sigma \\ t.h._{lower} &= \mu - z\sigma \end{aligned}$$

where μ and σ are mean and standard deviation of residual signals obtained from healthy engine in previous experiments. Assuming that the residual signals have normal distribution, we can define a confidence interval for each of them. In statistics, confidence interval is a measure of the reliability of an estimate. For future observations of a random variable X with mean and standard deviation of μ and σ, respectively, the probability of $l \quad X \quad u$ is determined from:

$$\begin{aligned} P(t.h._{lower} \quad X \quad t.h._{upper}) &= P(\frac{t.h._{lower} - \mu}{\sigma} \quad Z \quad \frac{t.h._{upper} \quad \mu}{\sigma}) \\ &= P(-z \quad Z \quad z) = \phi(z) - \phi(-z) = 2\phi(z) \end{aligned}$$

where Z is the standard normal variable. Thus, according to the above equation $t.h._{upper} = \mu \quad z\sigma$, $t.h._{lower} = \mu - z\sigma$. Hence, the probability of having the residual signals under normal condition can be adjusted by choosing different threshold values. In this thesis we consider a 99% confidence interval which corresponds to $z = 2.6$

In the following section, we study different single fault scenarios for the best individual learning methods (i.e. RBF-NARX) as well as the ensemble system in terms of the modeling accuracy. Thus, for each residual signal the threshold is determined by using the following relation:

$$\begin{aligned} t.h._{upper} &= \mu \quad 2.6\sigma \\ t.h._{lower} &= \mu - 2.6\sigma \end{aligned}$$

where μ and σ are mean and standard deviation of residual signals obtained from the healthy engine in previous experiments.

Remark 3.8. *In this research all ˇve residuals of the engine parameters are used for fault detection purpose. The fault would then be detected by the residual which acts faster than the others. If multiple residuals detect a fault at different time instances, then the fault detection time would be the minimum of different detection times given by different residuals. In other words:*

$$t_{FD} = \min(t_{FD_{res_1}}, ..., t_{FD_{res_n}})$$

where t_{FD} is the fault detection time and $t_{FD_{res_i}}$ is the time instance when the fault is detected by the i^{th} residual.

3.8.3 Fault Simulation Results

In this section different fault scenarios are studied to determine the effectiveness of the proposed ensemble method in fault detection. The simulation results of the selected scenarios are described in this section. These scenarios vary in the (a) fault type, (b) fault magnitude, and (c) fuel rate. In order to make a comparative study, the simulation results are individual learning models are also provided in this section. The simulations are performed in the cruise mode where the fuel rate varies between $\dot{m}_f = 0.7, 0.75, 0.8, 0.85$ of the maximum fuel rate. The ambient condition is set to normal condition, which is 0.7 Mach as a typical number in the cruise mode. The temperature is set to 0 Celsius degree.

The residuals are generated using the ensemble model that is trained in the previous section. We also generated the residuals using RBF-NARX model in order to make comparison between the ensemble and single-model solutions. The residuals are evaluated using the generated thresholds. We detect a fault if at least one of the residuals is greater than its corresponding determined thresholds.

In the following we study different fault scenarios in terms of fault type, fault severity, and fuel flow rate when the fault occurs.

3.8.4 Scenario I: Fault in the Compressor Efficiency

In this case we assume that there is a decrease in efficiency of the compressor. A failure happens with different magnitudes that are 1%, 2%, 4%, 6%, and 8% at t = 20 sec. Also the fuel rate varies within the range of cruise mode between 70%, 75%, 80%, 85% of the maximum fuel flow rate. The residual signals are generated for both the ensemble model (heterogenous ensemble with FSS pruning), and single-model (RBF-NARX) solutions. For breviary only the cases with $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$ are plotted here. Figures 3.50, 3.52, 3.54, 3.56, 3.58 show the fault in compressor efficiency detected using the ensemble model. Figures 3.51, 3.53, 3.55, 3.57, 3.59 show the fault in compressor efficiency detected using the single-model based solution (i.e. RBF-NARX model).

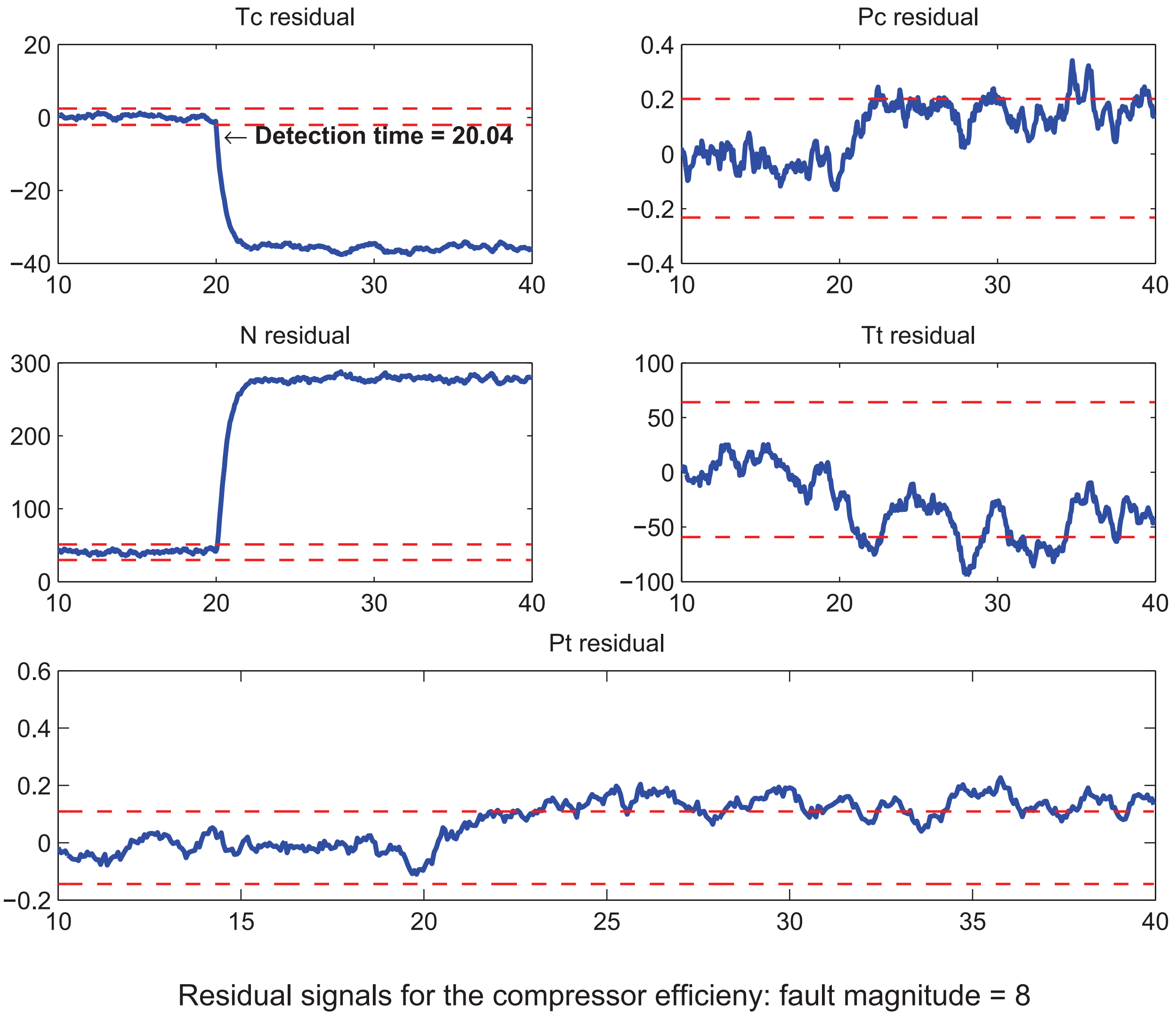
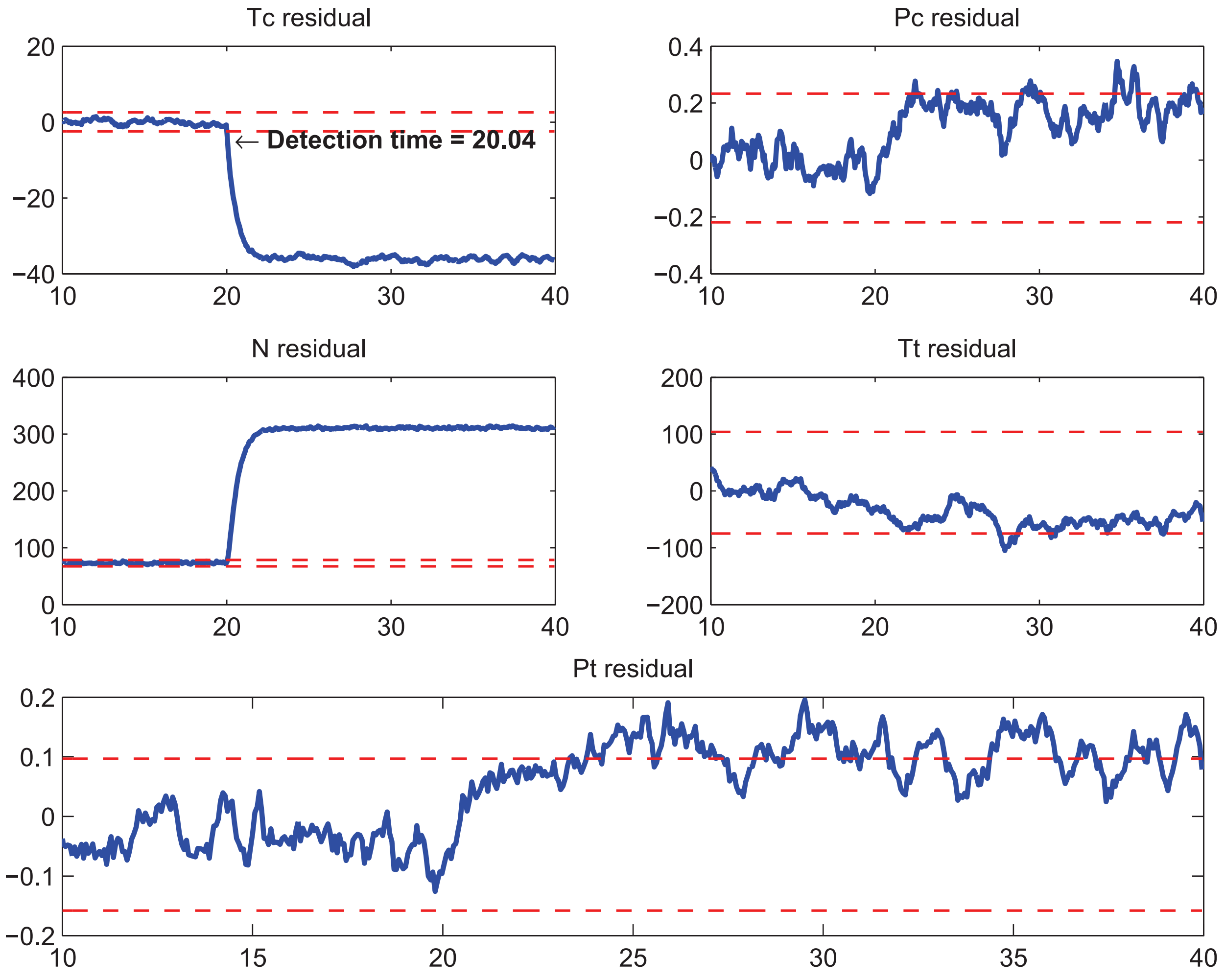


Figure 3.50: Residual generated using ensemble model with FSS pruning, 8% decrease in compressor efficiency at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.



Residual signals for the compressor efficiency: fault magnitude = 8

Figure 3.51: Residual generated using RBF-NARX model, 8% decrease in compressor efficiency at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.

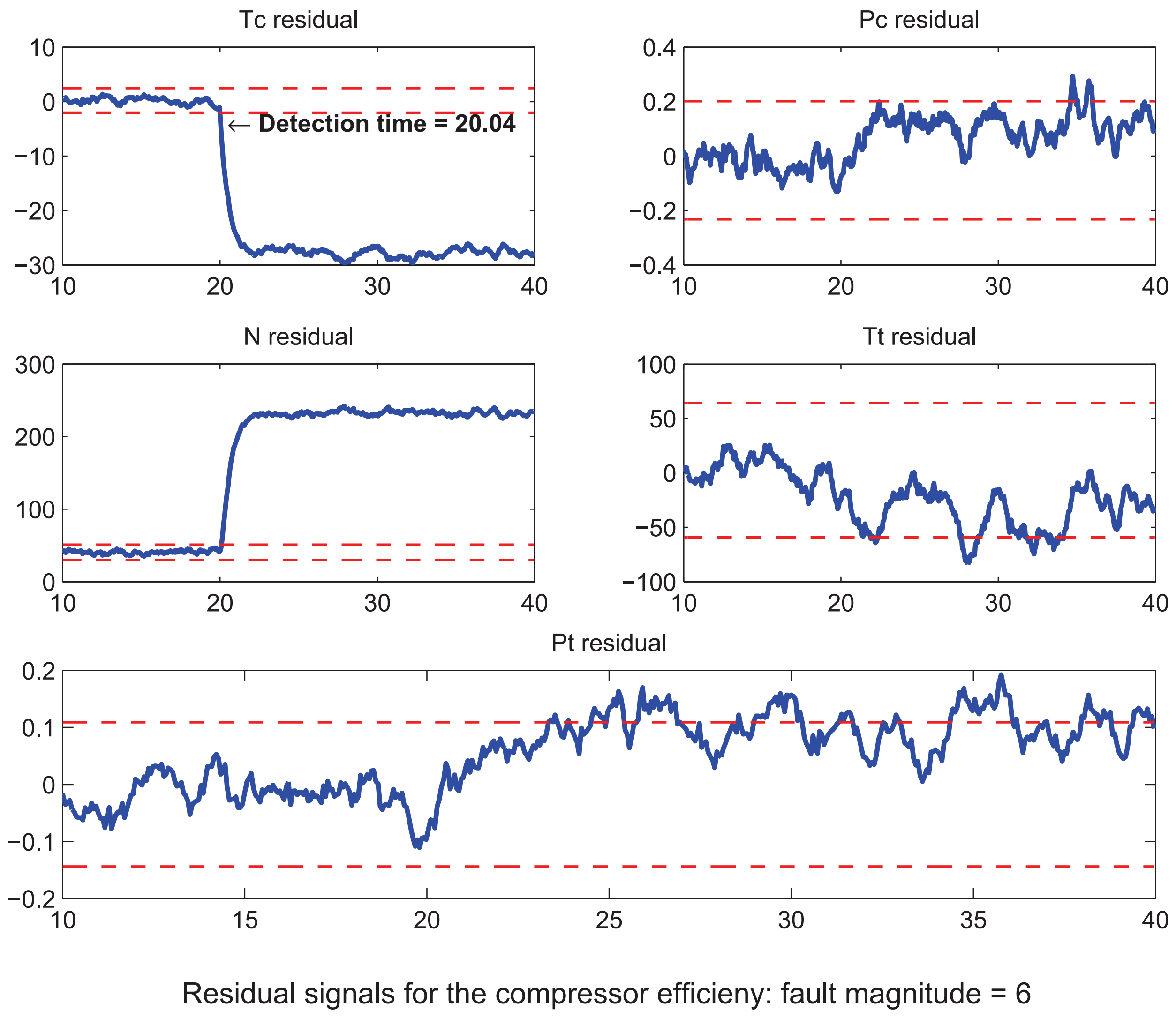
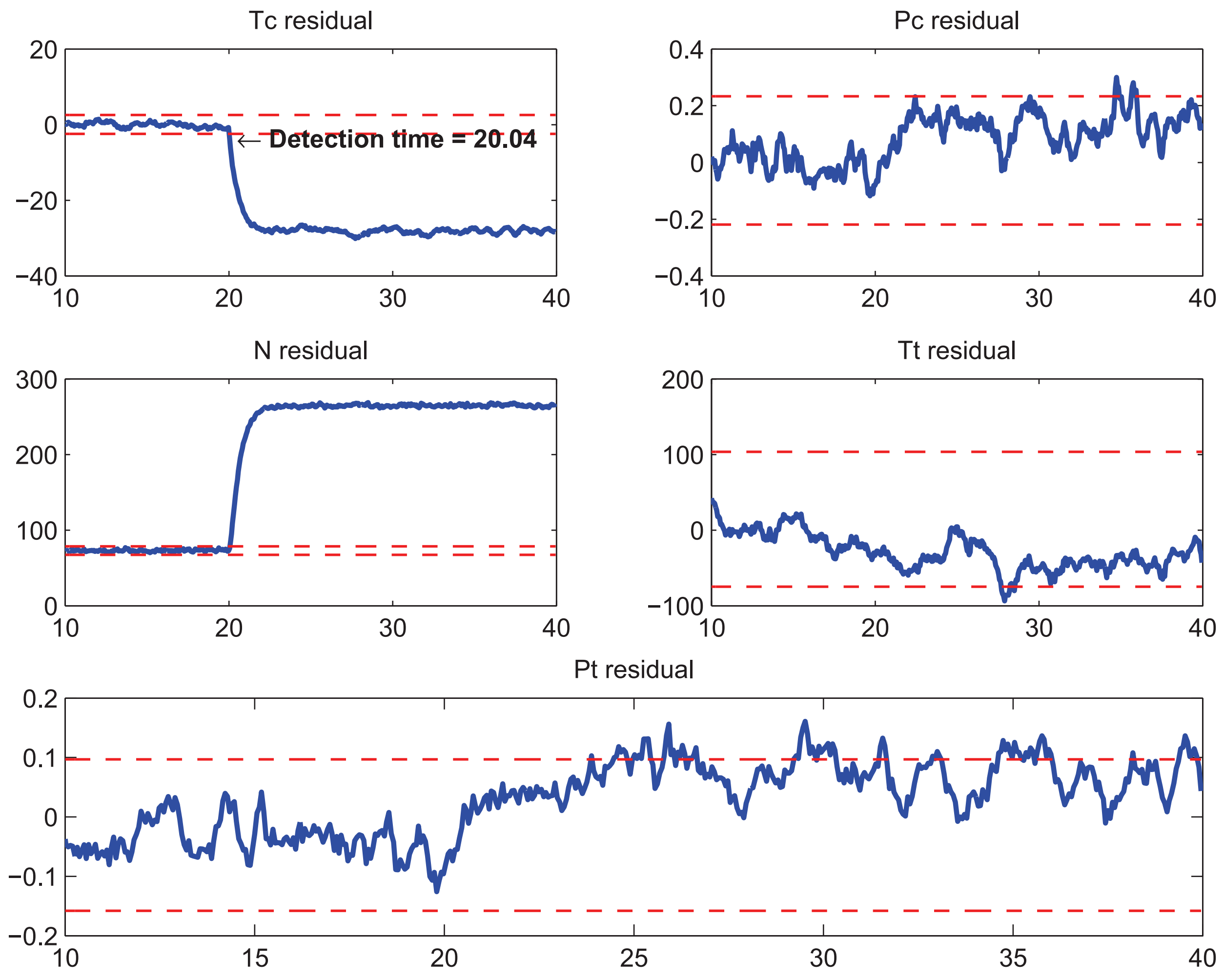


Figure 3.52: Residual generated using ensemble model with FSS pruning, 6% decrease in compressor efficiency at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.



Residual signals for the compressor efficiency: fault magnitude = 6

Figure 3.53: Residual generated using RBF-NARX model, 6% decrease in compressor efficiency at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.

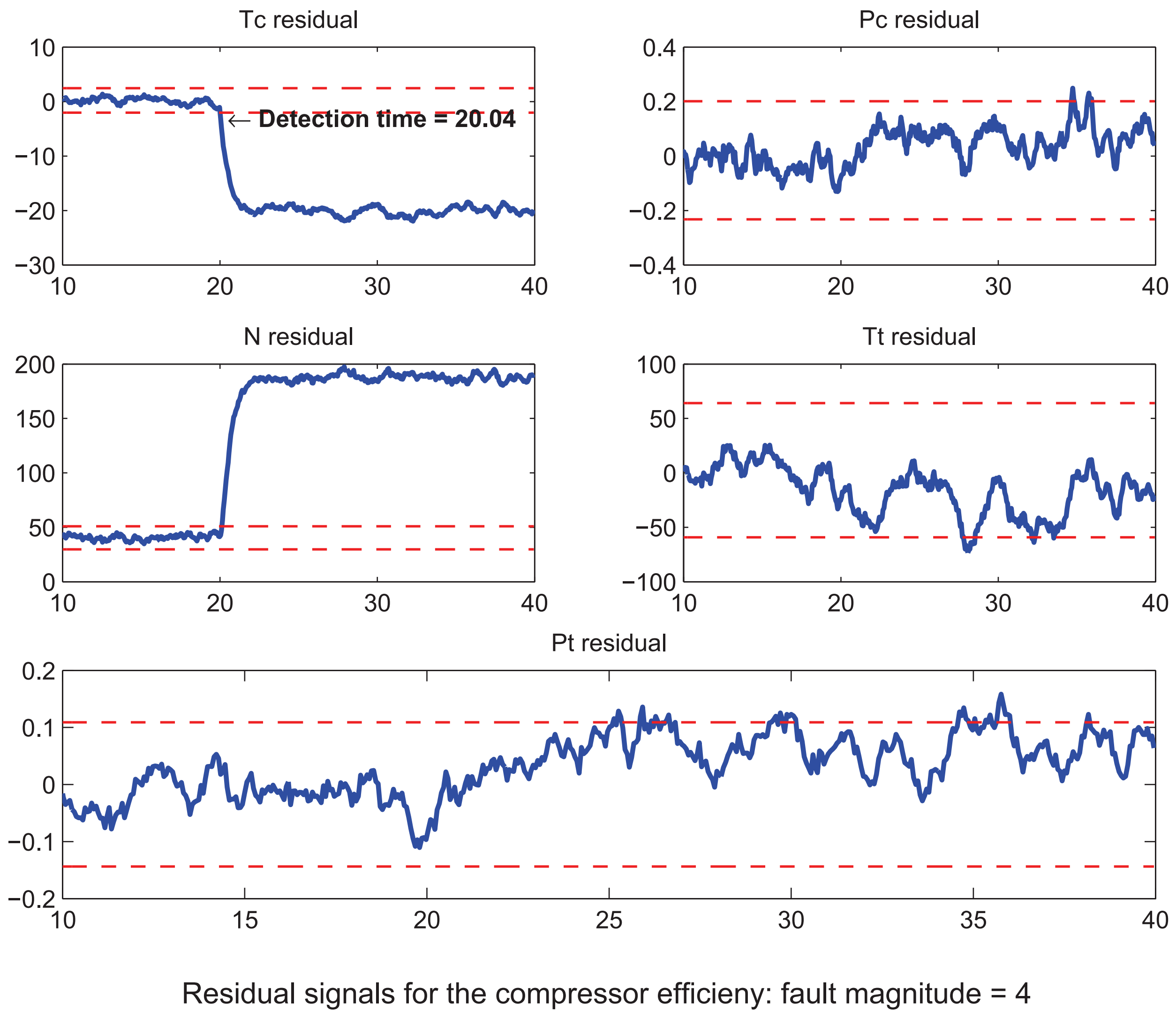


Figure 3.54: Residual generated using ensemble model with FSS pruning, 4% decrease in compressor efficiency at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.

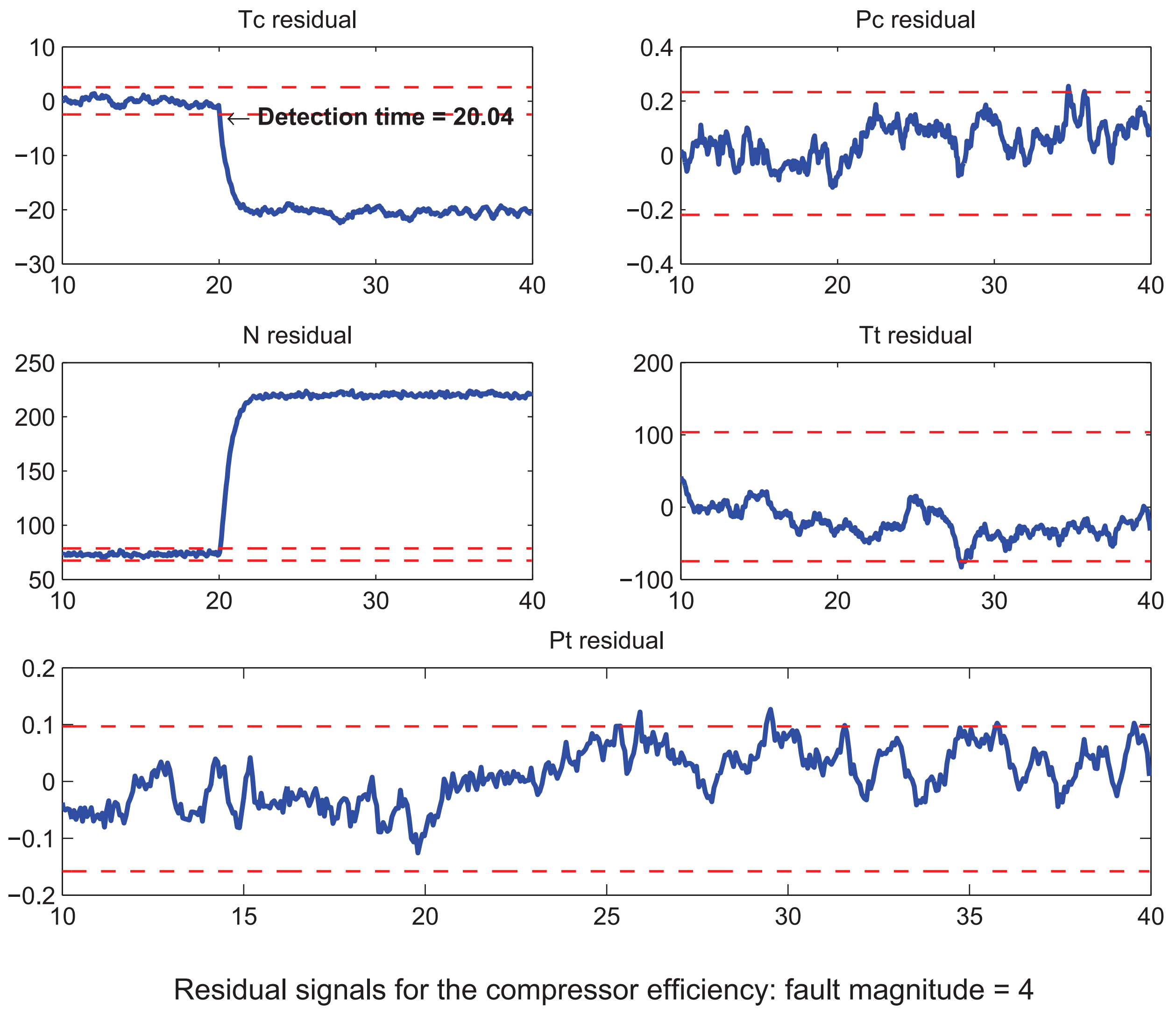


Figure 3.55: Residual generated using RBF-NARX model, 4% decrease in compressor efficiency at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.

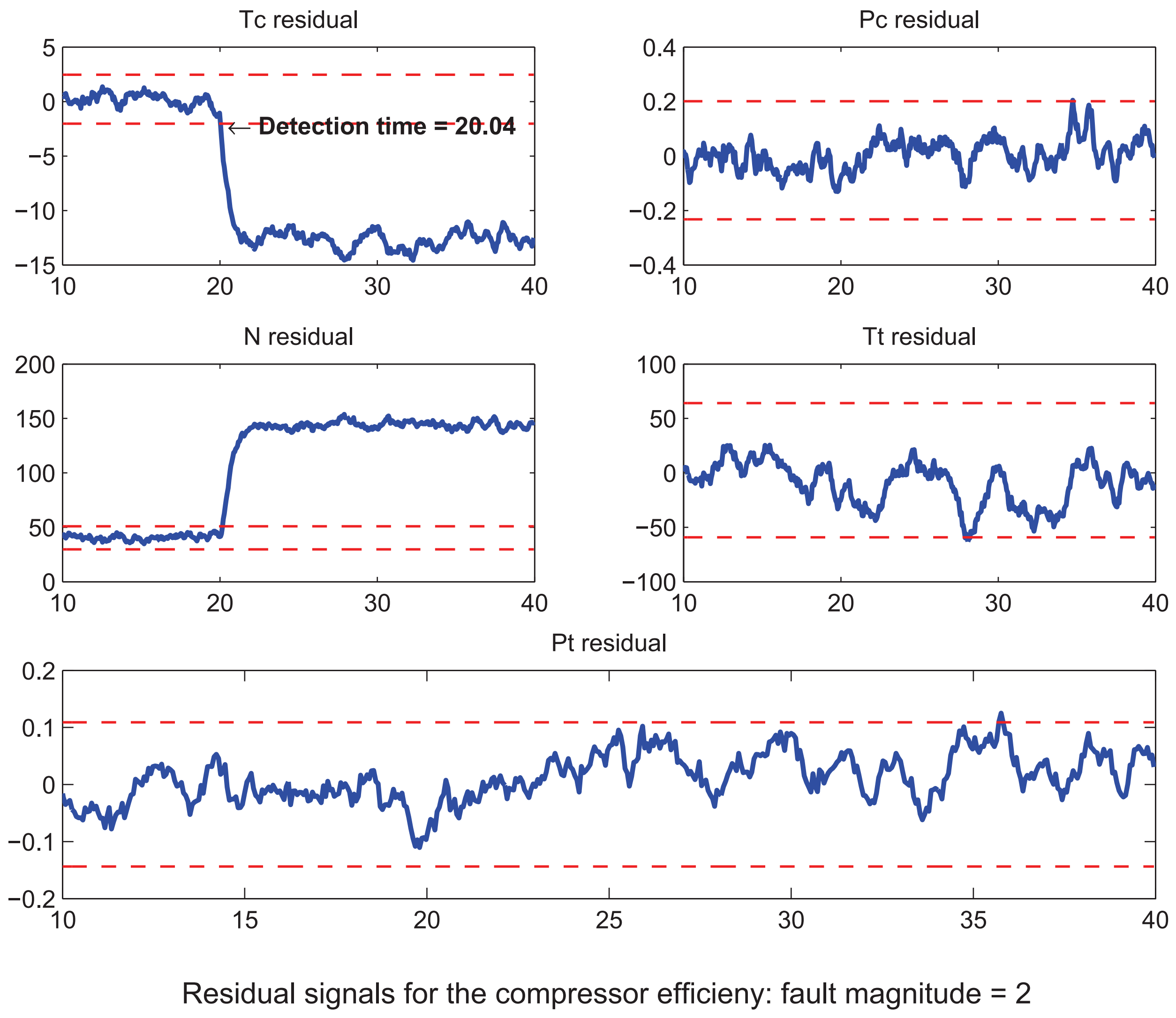


Figure 3.56: Residual generated using ensemble model with FSS pruning, 2% decrease in compressor efficiency at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.

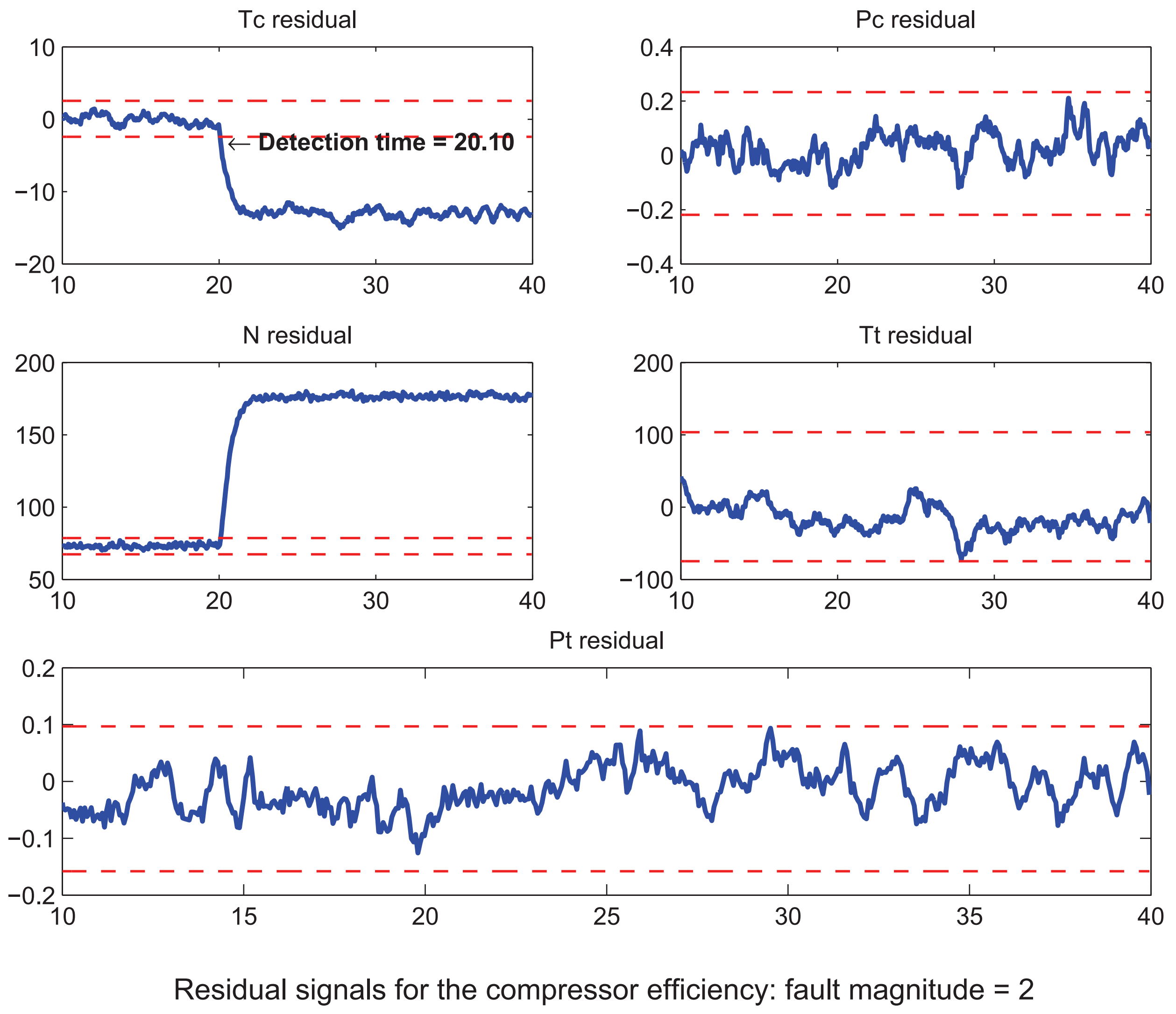


Figure 3.57: Residual generated using RBF-NARX model, 2% decrease in compressor efficiency at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.

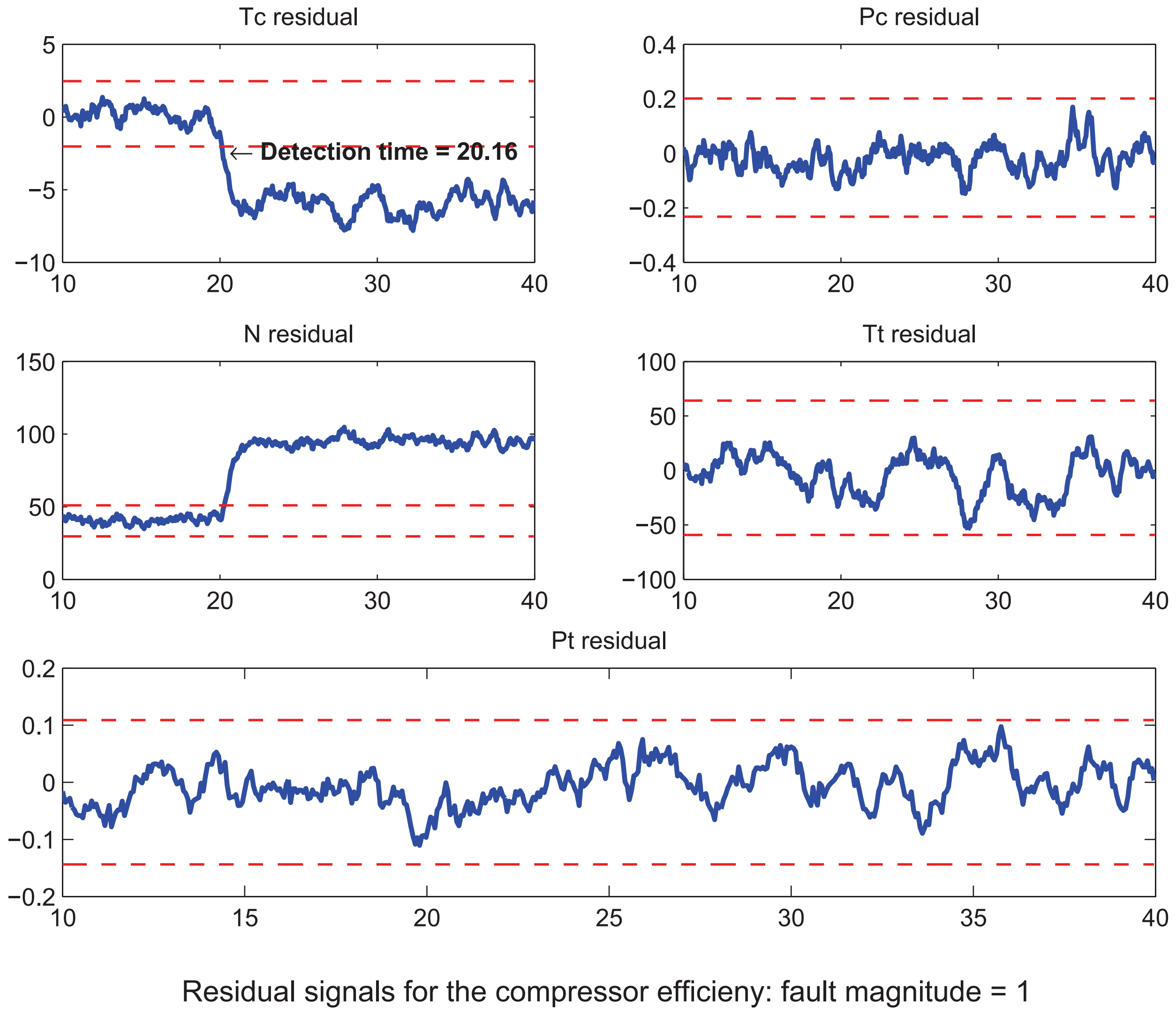


Figure 3.58: Residual generated using ensemble model with FSS pruning, 1% decrease in compressor efficiency at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.

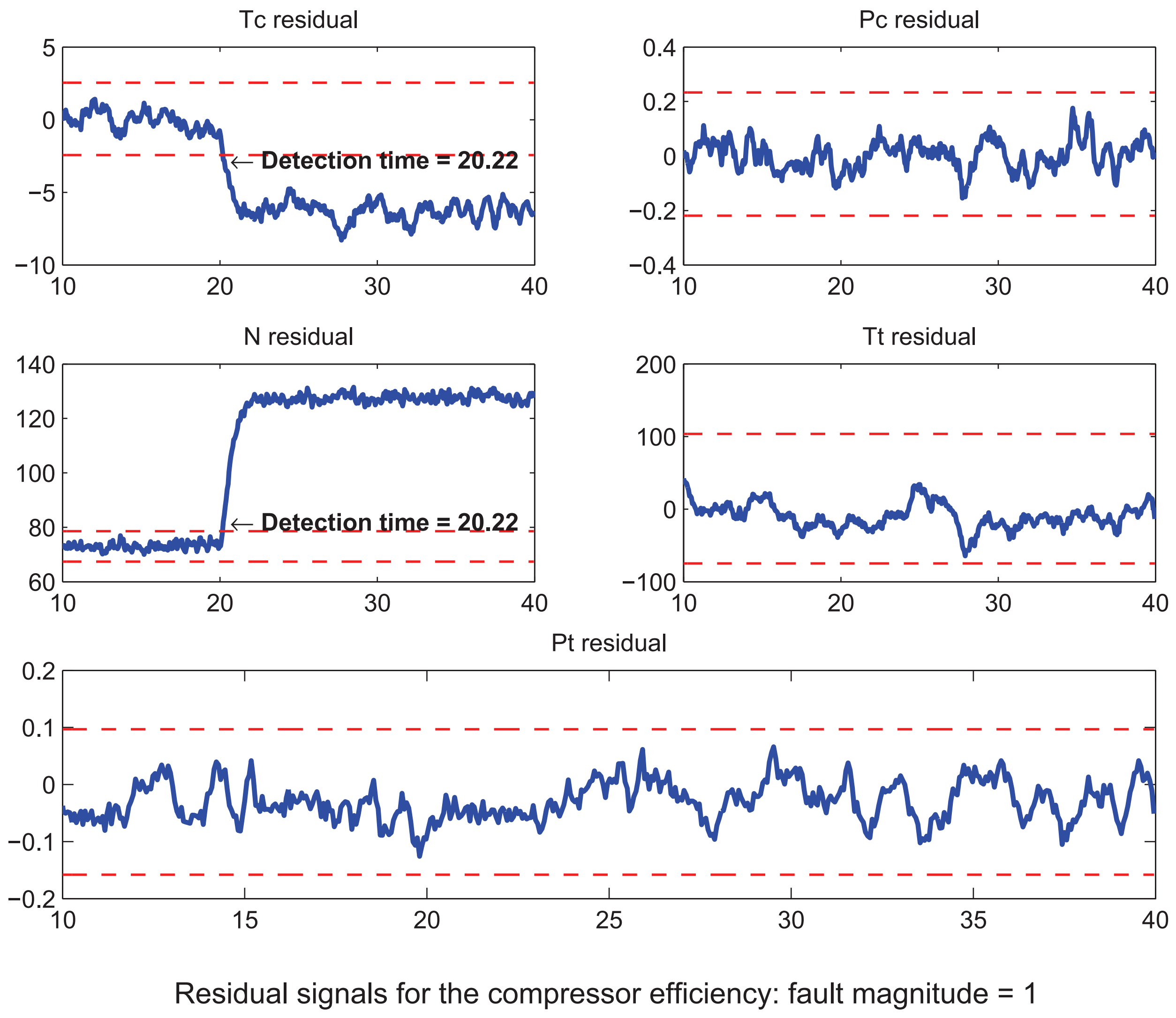


Figure 3.59: Residual generated using RBF-NARX model, 1% decrease in compressor efficiency at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.

Section 3.8.8 shows a comparative study between the fault detection results of the ensemble solution and the single-model solution which indicates an improvement in fault detection accuracy using the ensemble model. The failure has happened at $t = 20$ sec. Tables 3.60 and 3.61 summarize the fault detection time using ensemble model and RBF-NARX models respectively.

Table 3.60: Fault detection time summary using ensemble model: compressor efficiency fault injected at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.

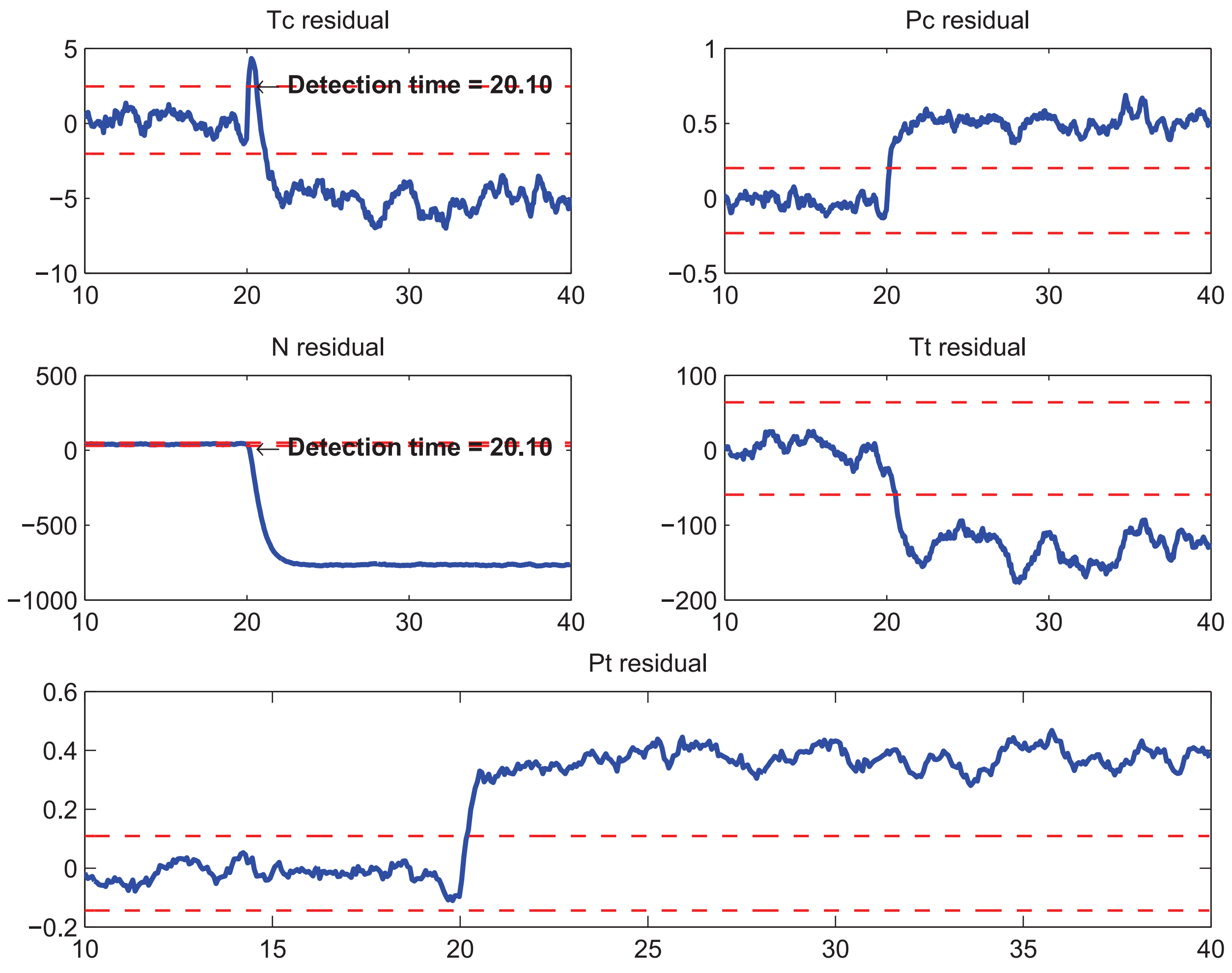
Fault Type	Fault Severity	Fuel Rate	Injection Time	Detection Time
F_{ec}	8%	$\dot{m}_f$ = 0.85 $\quad \dot{m}_{f,maximum}$	20	20.04
F_{ec}	6%	$\dot{m}_f$ = 0.85 $\quad \dot{m}_{f,maximum}$	20	20.04
F_{ec}	4%	$\dot{m}_f$ = 0.85 $\quad \dot{m}_{f,maximum}$	20	20.04
F_{ec}	2%	$\dot{m}_f$ = 0.85 $\quad \dot{m}_{f,maximum}$	20	20.04
F_{ec}	1%	$\dot{m}_f$ = 0.85 $\quad \dot{m}_{f,maximum}$	20	20.16

Table 3.61: Fault detection time summary using RBF-NARX model: compressor efficiency fault injected at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.

Fault Type	Fault Severity	Fuel Rate	Injection Time	Detection Time
F_{ec}	8%	$\dot{m}_f$ = 0.85 $\quad \dot{m}_{f,maximum}$	20	20.04
F_{ec}	6%	$\dot{m}_f$ = 0.85 $\quad \dot{m}_{f,maximum}$	20	20.04
F_{ec}	4%	$\dot{m}_f$ = 0.85 $\quad \dot{m}_{f,maximum}$	20	20.04
F_{ec}	2%	$\dot{m}_f$ = 0.85 $\quad \dot{m}_{f,maximum}$	20	20.10
F_{ec}	1%	$\dot{m}_f$ = 0.85 $\quad \dot{m}_{f,maximum}$	20	20.22

3.8.5 Scenario II: Fault in the Compressor Mass Flow Rate

In this case we assume that there is a decrease in effectiveness of the compressor mass flow rate. The failure happens with different magnitudes that are 1%, 2%, 4%, 6%, and 8%. Also the fuel rate varies between 70%, 75%, 80%, and 85% of the maximum in each case. The residual signals are generated for the individual learning algorithms as well as the ensemble system. Figures 3.60, 3.62, 3.64, 3.66, 3.68 show the fault in compressor efficiency detected using the ensemble model. Figures 3.61, 3.63, 3.65, 3.67, 3.69 show the fault in compressor efficiency detected using the single-model based solution (i.e. RBF-NARX model).



Residual signals for the compressor mass flow rate: fault magnitude = 8

Figure 3.60: Residual generated using ensemble model with FSS pruning, 8% decrease in compressor mass flow rate at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.

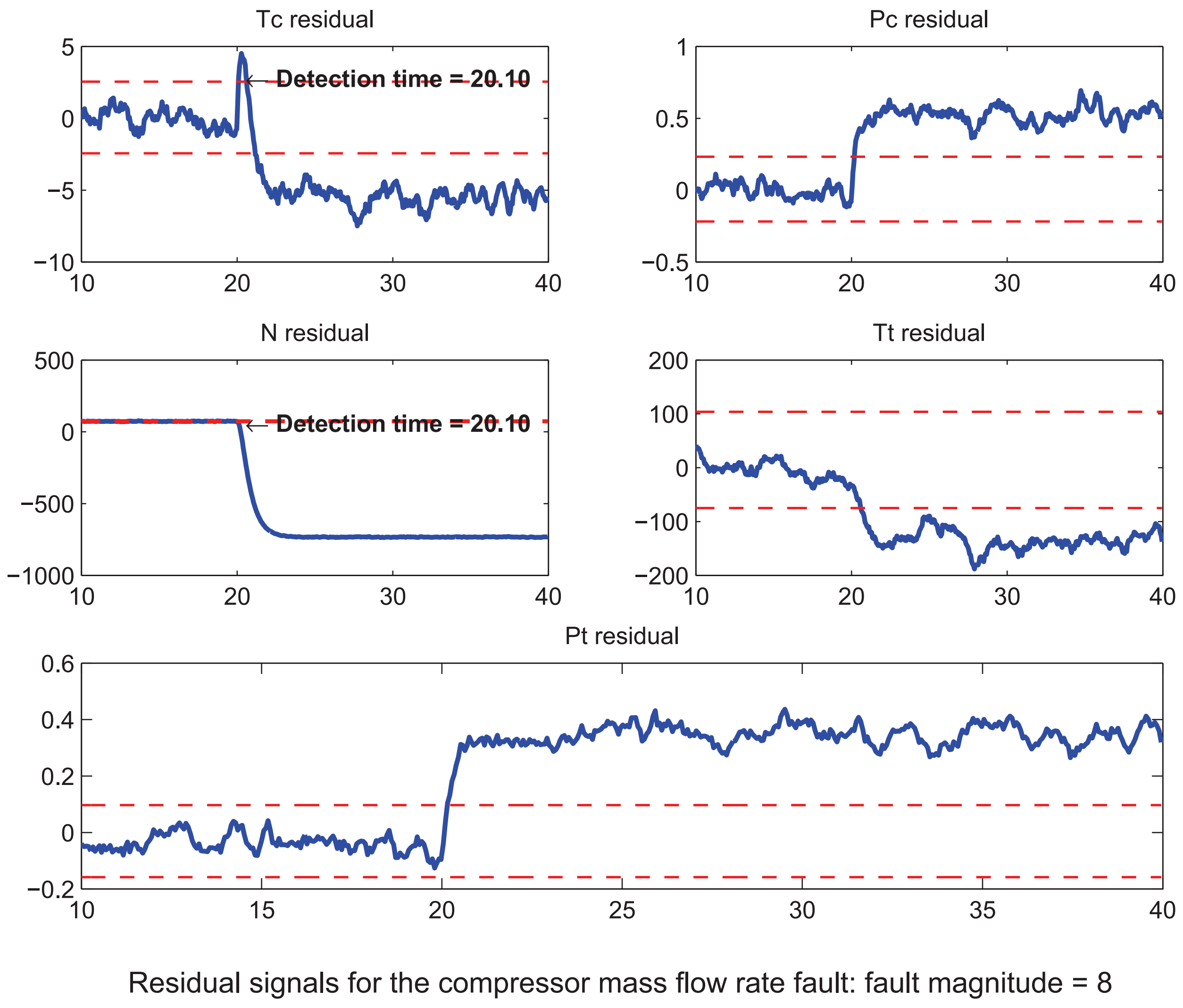
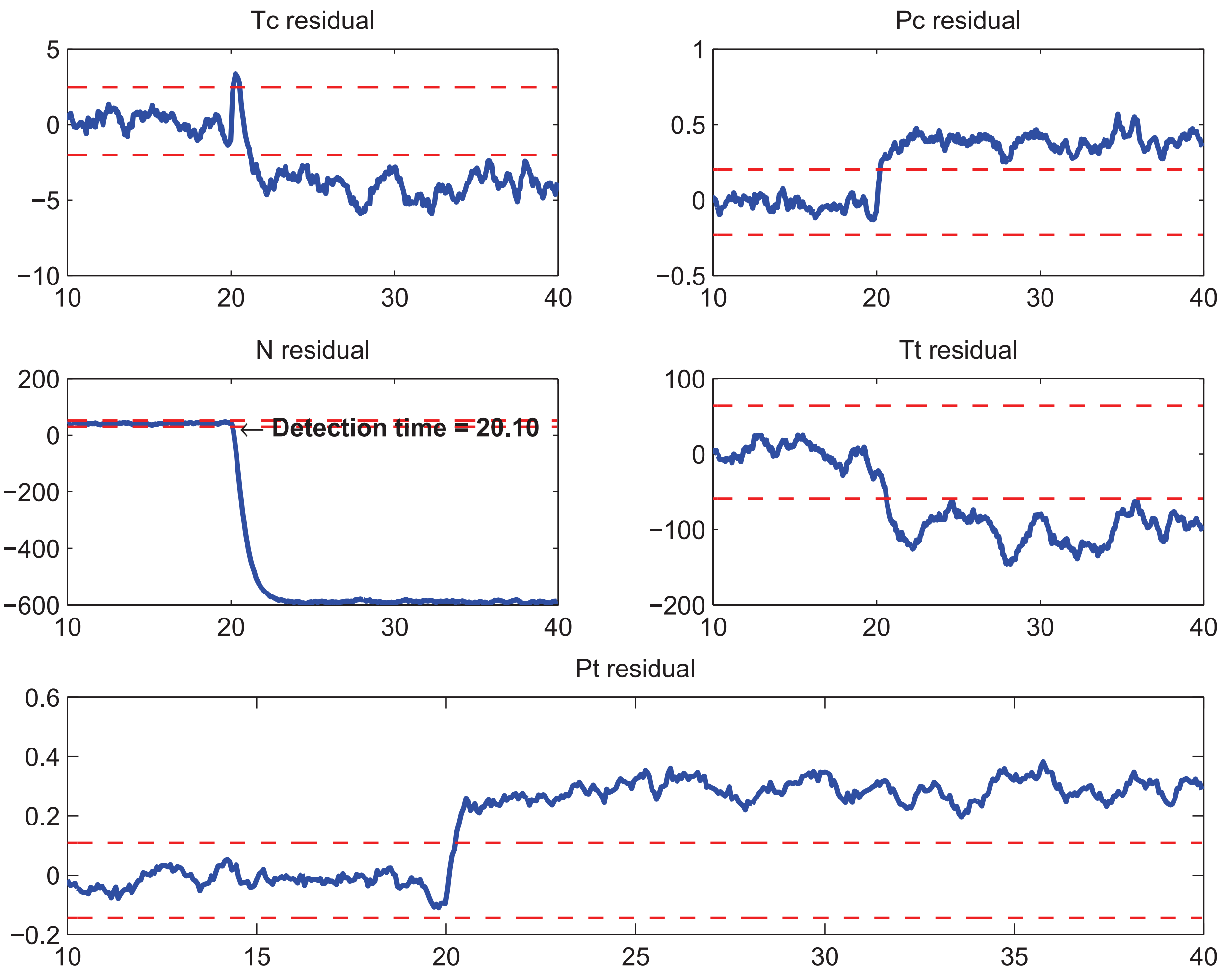
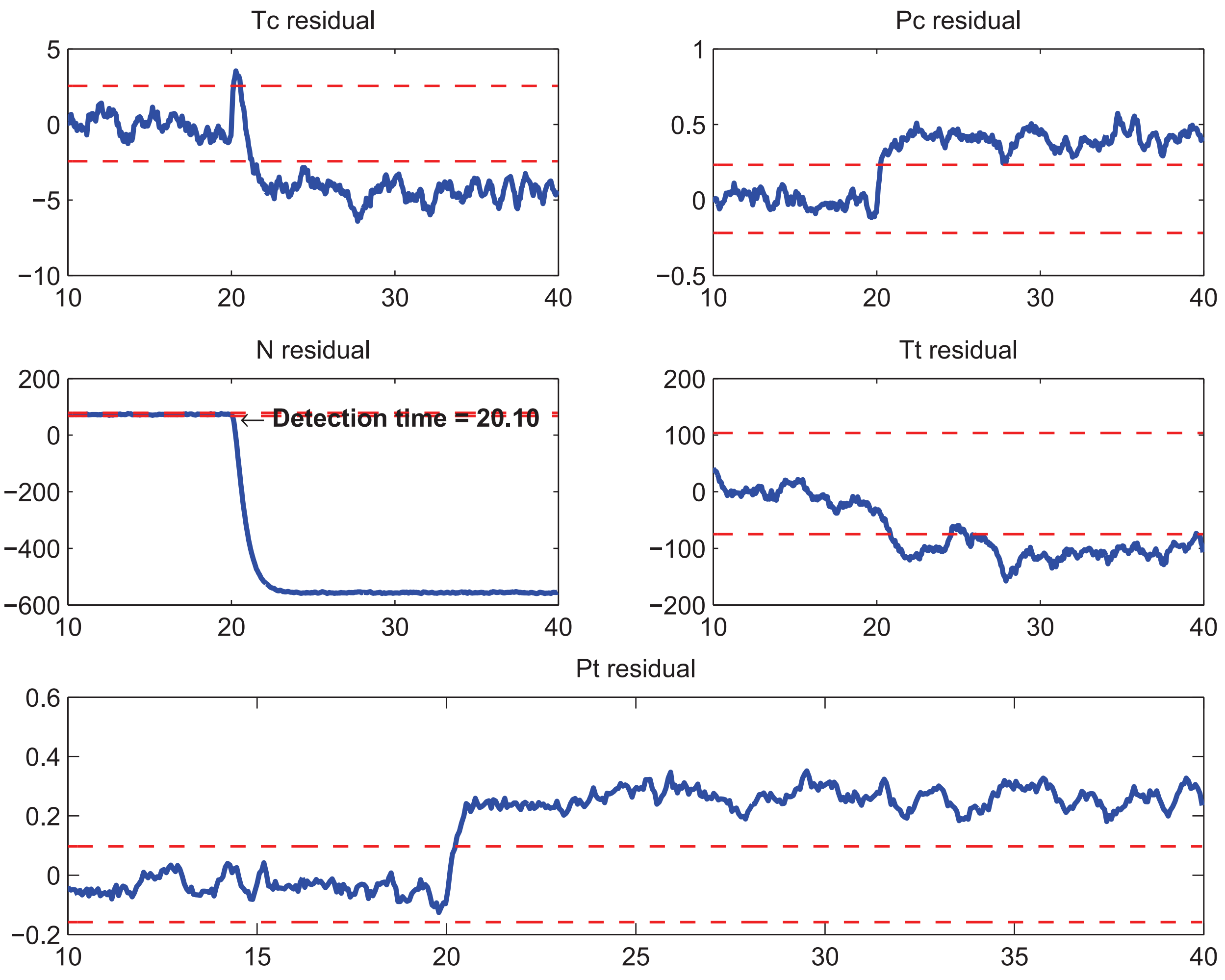


Figure 3.61: Residual generated using RBF-NARX model, 8% decrease in compressor mass flow rate at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.



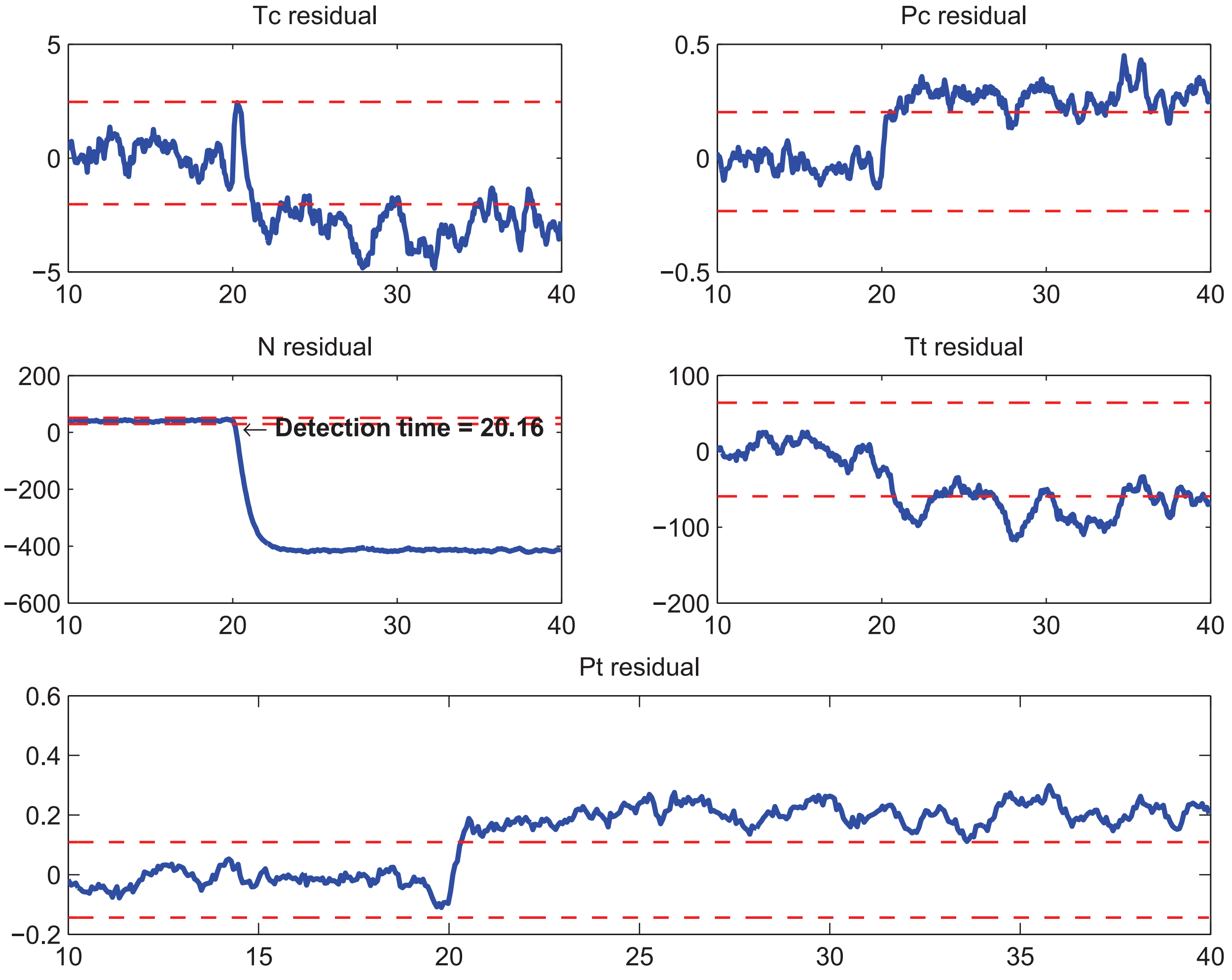
Residual signals for the compressor mass flow rate: fault magnitude = 6

Figure 3.62: Residual generated using ensemble model with FSS pruning, 6% decrease in compressor mass flow rate at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.



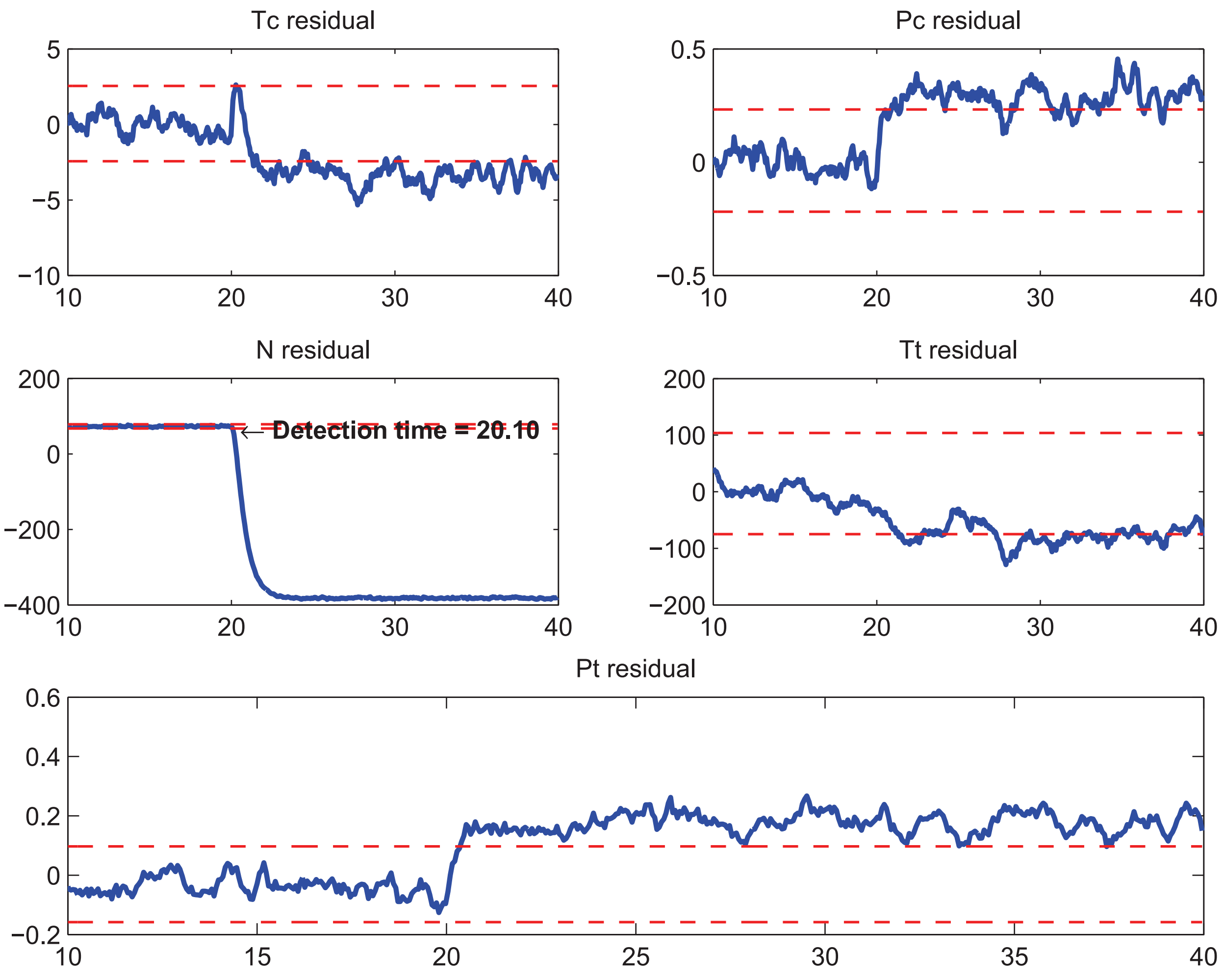
Residual signals for the compressor mass flow rate fault: fault magnitude = 6

Figure 3.63: Residual generated using RBF-NARX model, 6% decrease in compressor mass flow rate at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.



Residual signals for the compressor mass flow rate: fault magnitude = 4

Figure 3.64: Residual generated using ensemble model with FSS pruning, 4% decrease in compressor mass flow rate at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.



Residual signals for the compressor mass flow rate fault: fault magnitude = 4

Figure 3.65: Residual generated using RBF-NARX model, 4% decrease in compressor mass flow rate at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.

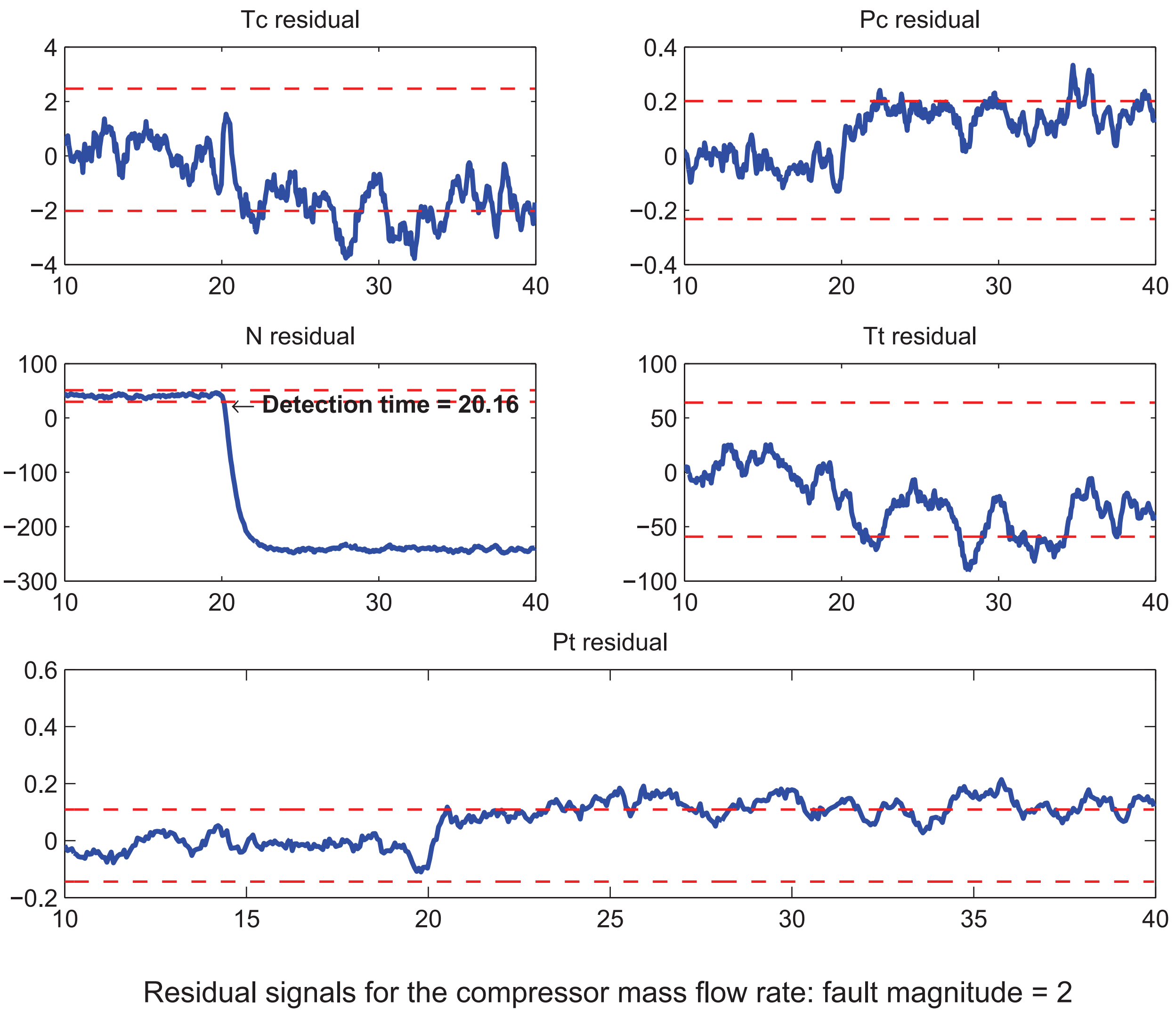
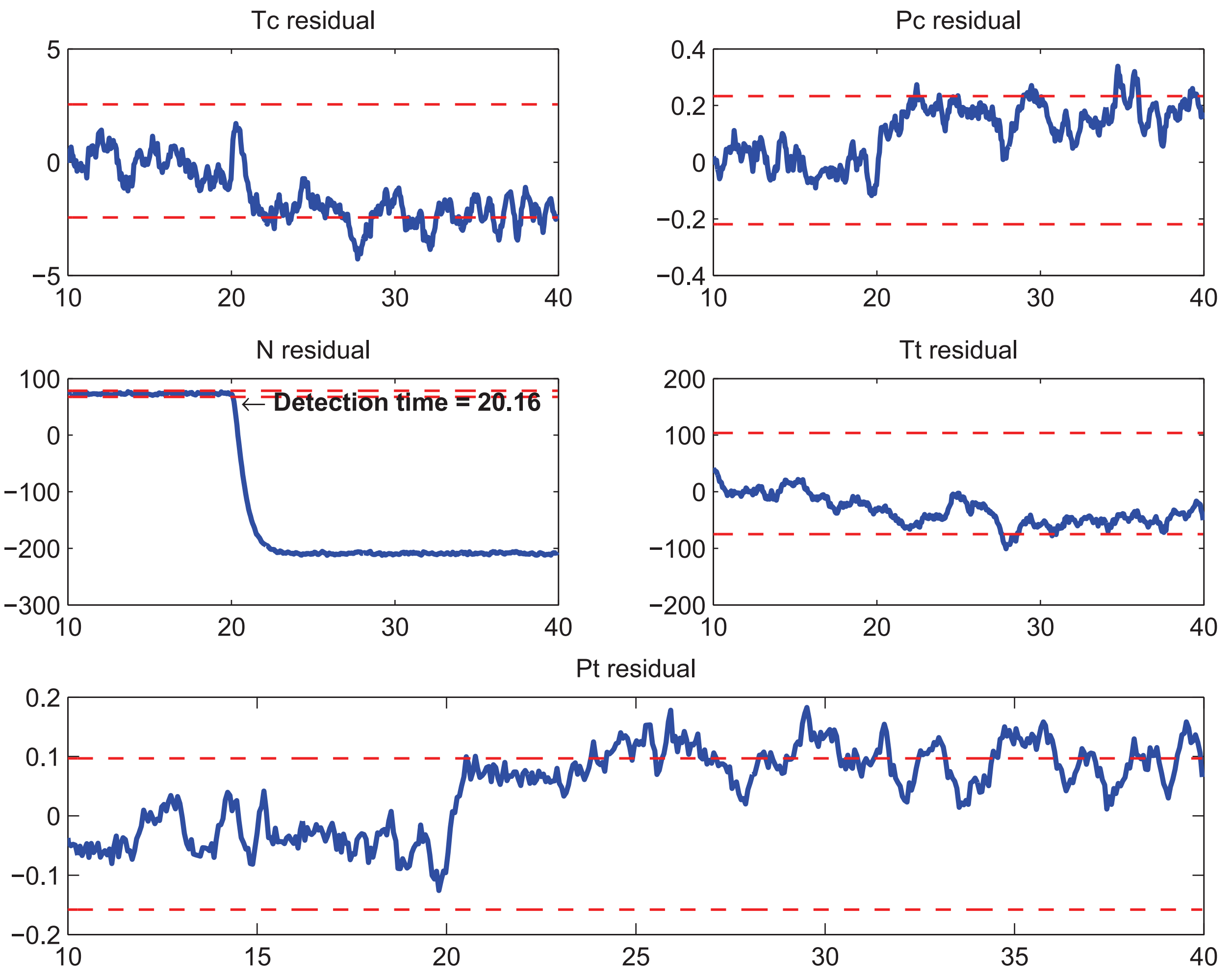
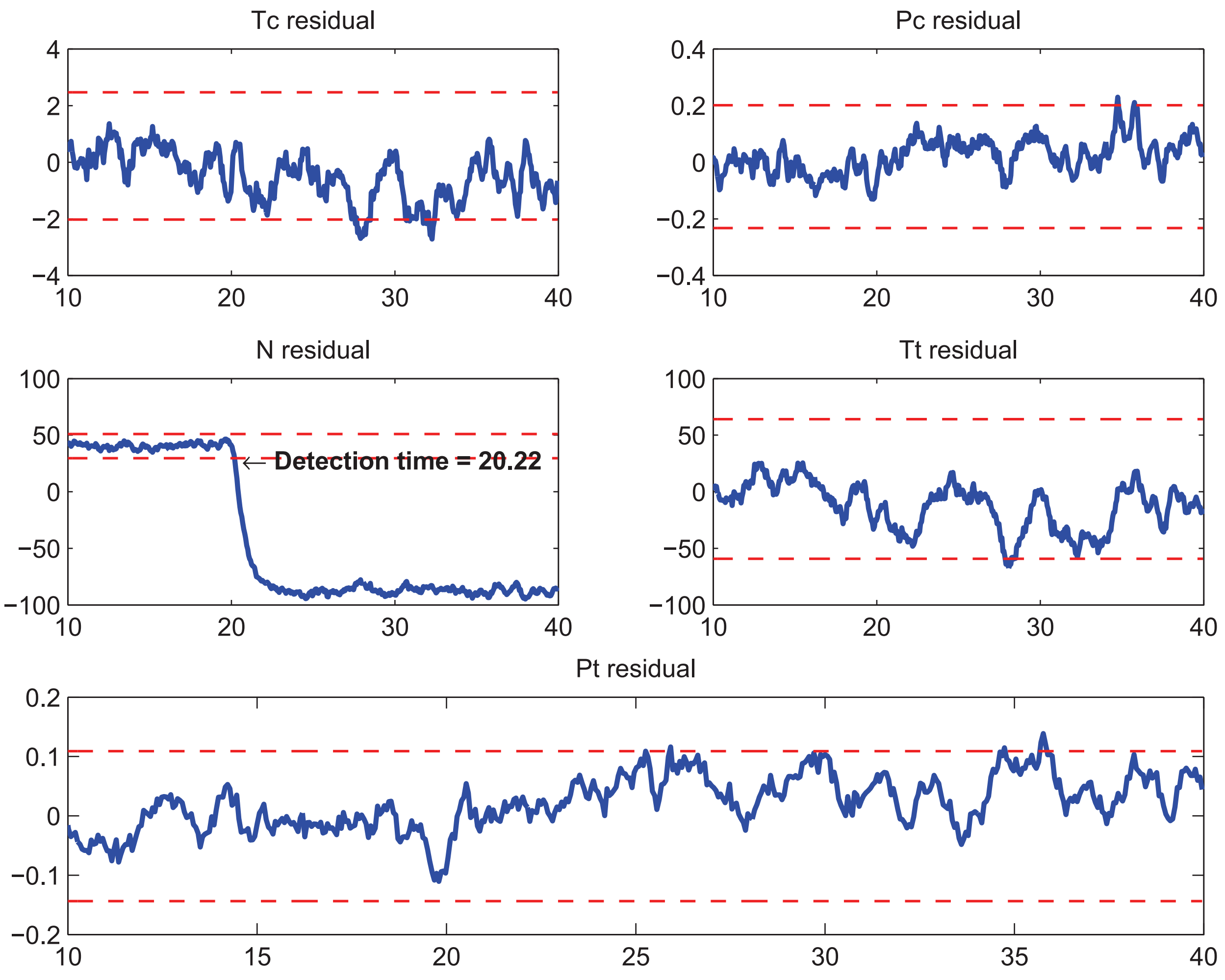


Figure 3.66: Residual generated using ensemble model with FSS pruning, 2% decrease in compressor mass flow rate at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.



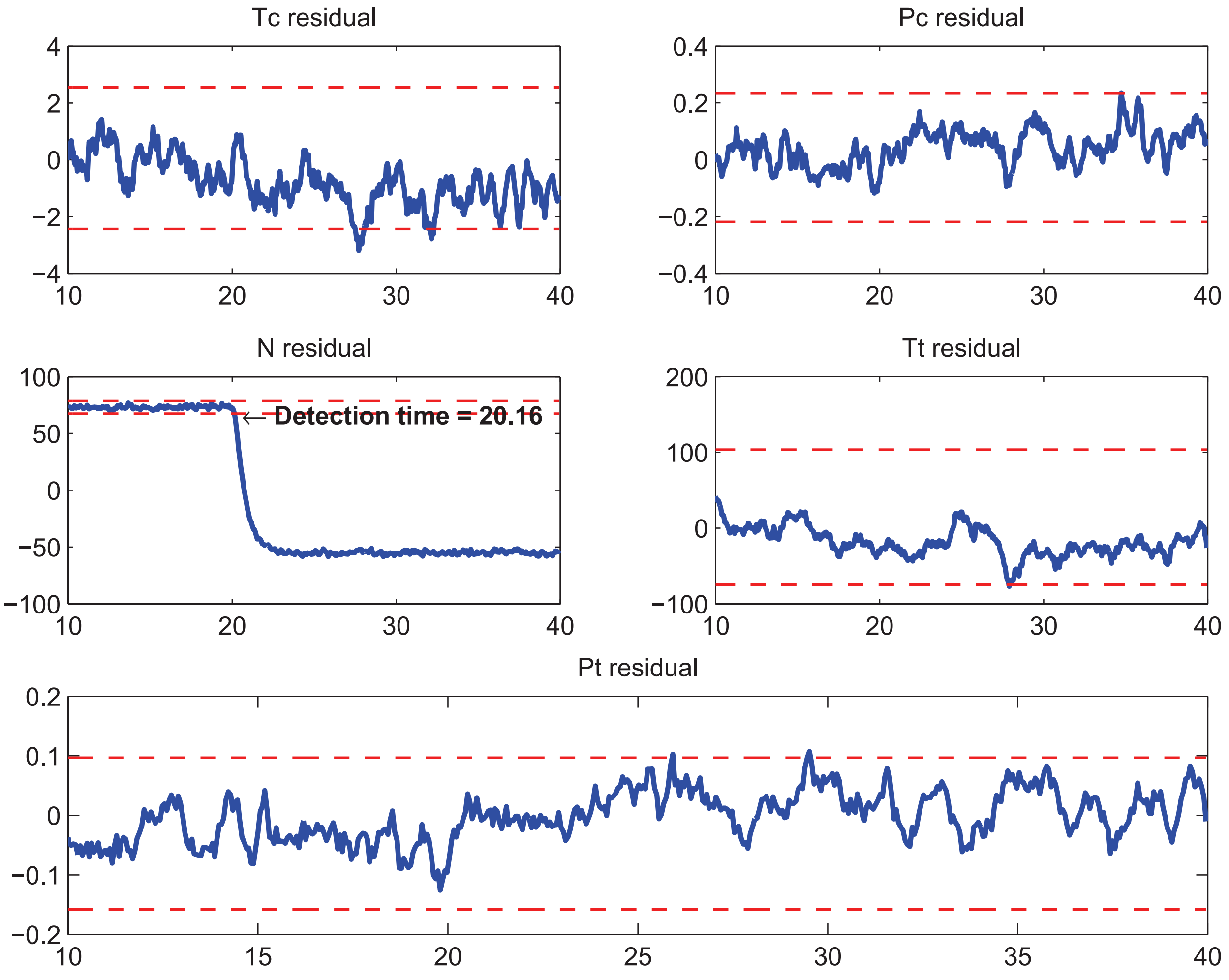
Residual signals for the compressor mass flow rate fault: fault magnitude = 2

Figure 3.67: Residual generated using RBF-NARX model, 2% decrease in compressor mass flow rate at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.



Residual signals for the compressor mass flow rate: fault magnitude = 1

Figure 3.68: Residual generated using ensemble model with FSS pruning, 1% decrease in compressor mass flow rate at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.



Residual signals for the compressor mass flow rate fault: fault magnitude = 1

Figure 3.69: Residual generated using RBF-NARX model, 1% decrease in compressor mass flow rate at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.

Section 3.8.8 shows a comparative study between the fault detection results of the ensemble solution and the single-model solution which indicates an improvement in fault detection accuracy using the ensemble model. The failure has happened at $t = 20$ sec. Tables 3.62 and 3.63 summarize the fault detection time using ensemble model and RBF-NARX models respectively.

Table 3.62: Fault detection time summary using ensemble model: compressor efficiency fault injected at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.

Fault Type	Fault Severity	Fuel Rate	Injection Time	Detection Time
F_{mc}	8%	$\dot{m}_f$ = 0.85 $\dot{m}_{f,maximum}$	20	20.10
F_{mc}	6%	$\dot{m}_f$ = 0.85 $\dot{m}_{f,maximum}$	20	20.10
F_{mc}	4%	$\dot{m}_f$ = 0.85 $\dot{m}_{f,maximum}$	20	20.16
F_{mc}	2%	$\dot{m}_f$ = 0.85 $\dot{m}_{f,maximum}$	20	20.16
F_{mc}	1%	$\dot{m}_f$ = 0.85 $\dot{m}_{f,maximum}$	20	20.22

Table 3.63: Fault detection time summary using RBF-NARX model: compressor efficiency fault injected at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.

Fault Type	Fault Severity	Fuel Rate	Injection Time	Detection Time
F_{mc}	8%	$\dot{m}_f$ = 0.85 $\dot{m}_{f,maximum}$	20	20.10
F_{mc}	6%	$\dot{m}_f$ = 0.85 $\dot{m}_{f,maximum}$	20	20.10
F_{mc}	4%	$\dot{m}_f$ = 0.85 $\dot{m}_{f,maximum}$	20	20.10
F_{mc}	2%	$\dot{m}_f$ = 0.85 $\dot{m}_{f,maximum}$	20	20.16
F_{mc}	1%	$\dot{m}_f$ = 0.85 $\dot{m}_{f,maximum}$	20	20.16

3.8.6 Scenario III: Fault in the Turbine Efficiency

In this scenario we assume that there is a decrease in the turbine efficiency. The failure happens with different magnitudes that are 1%, 2%, 4%, 6%, and 8%. Also the fuel rate varies between 70%, 75%, 80%, 90%, and 95% of the maximum in each case. The residual signals are generated for the individual learning algorithms as well as the ensemble system. Figures 3.70, 3.72, 3.74, 3.76, 3.78 show the fault in compressor efficiency detected using the ensemble model. Figures 3.71, 3.73, 3.75, 3.77, 3.79 show the fault in compressor efficiency detected using the single-model based solution (i.e. RBF-NARX model).

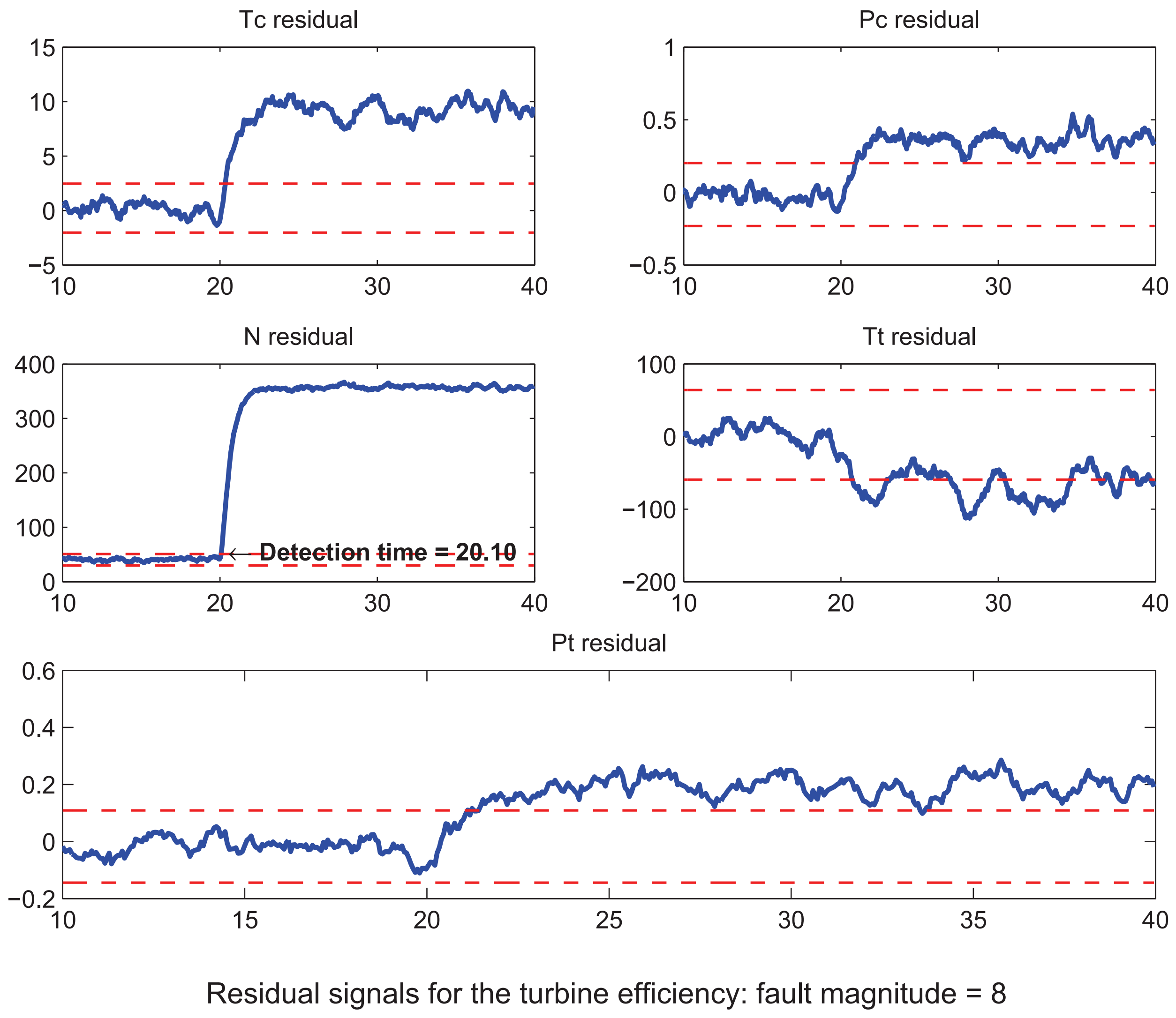


Figure 3.70: Residual generated using ensemble model with FSS pruning, 8% decrease in turbine efficiency at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.

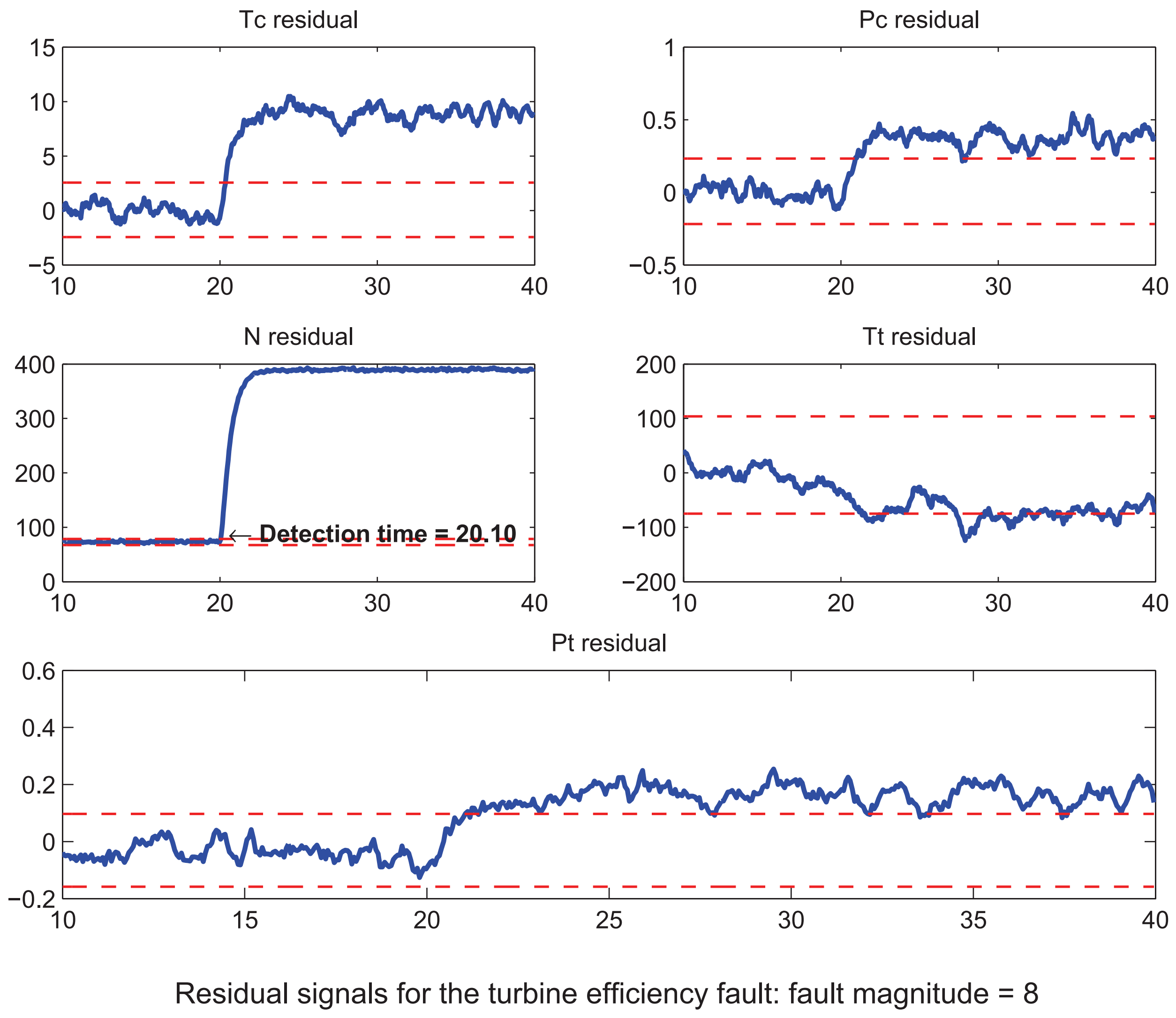


Figure 3.71: Residual generated using RBF-NARX model, 8% decrease in turbine efficiency at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.

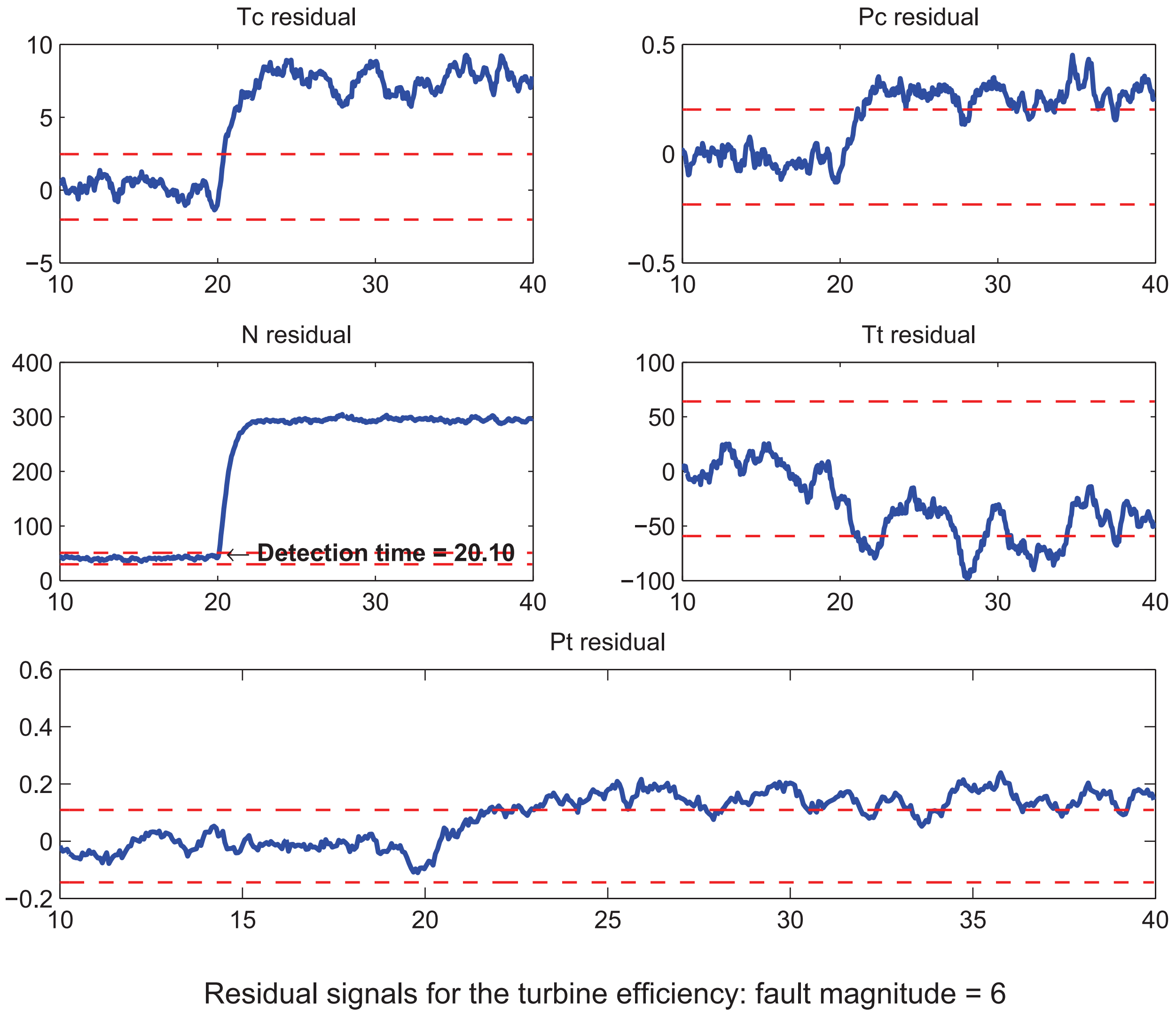
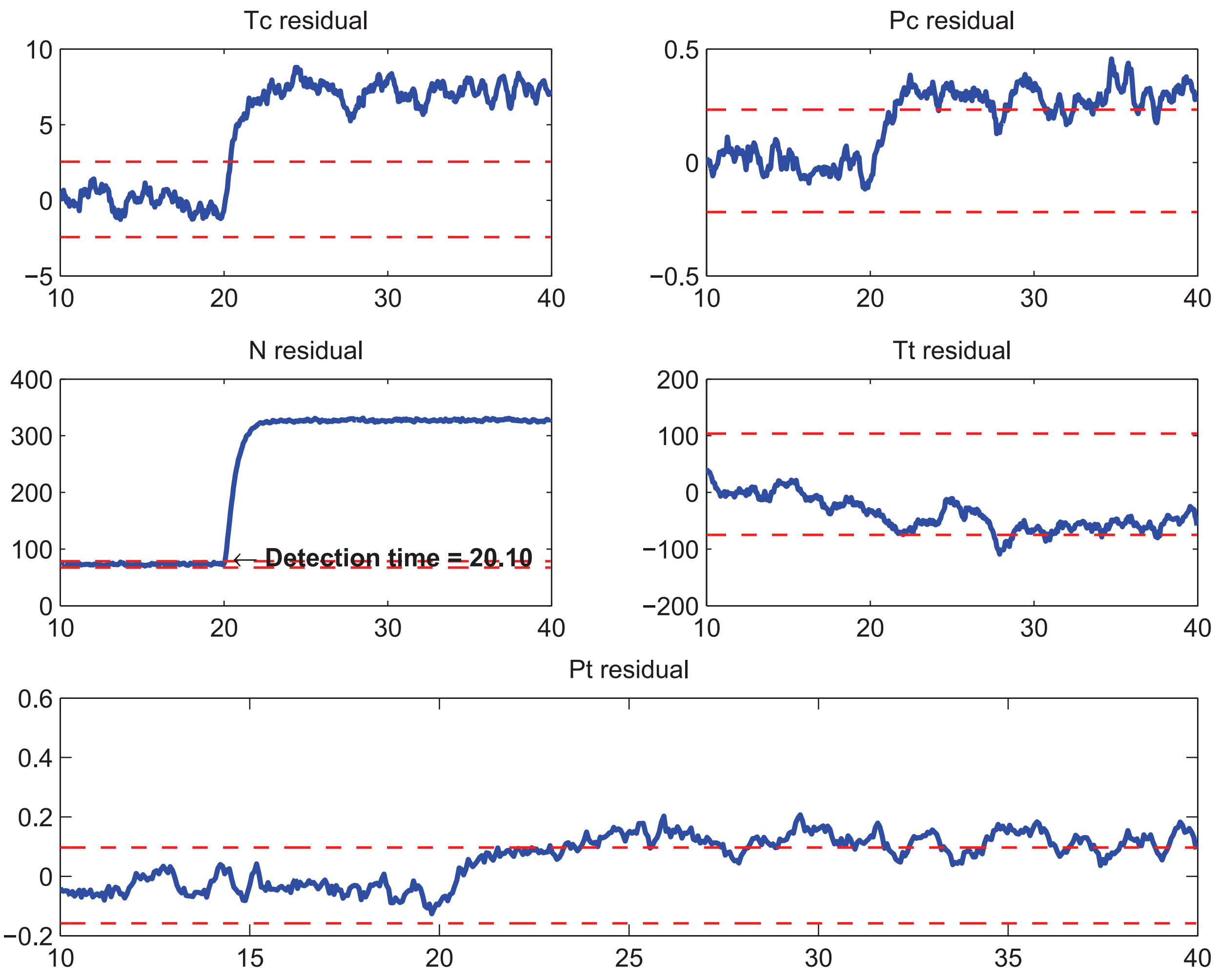
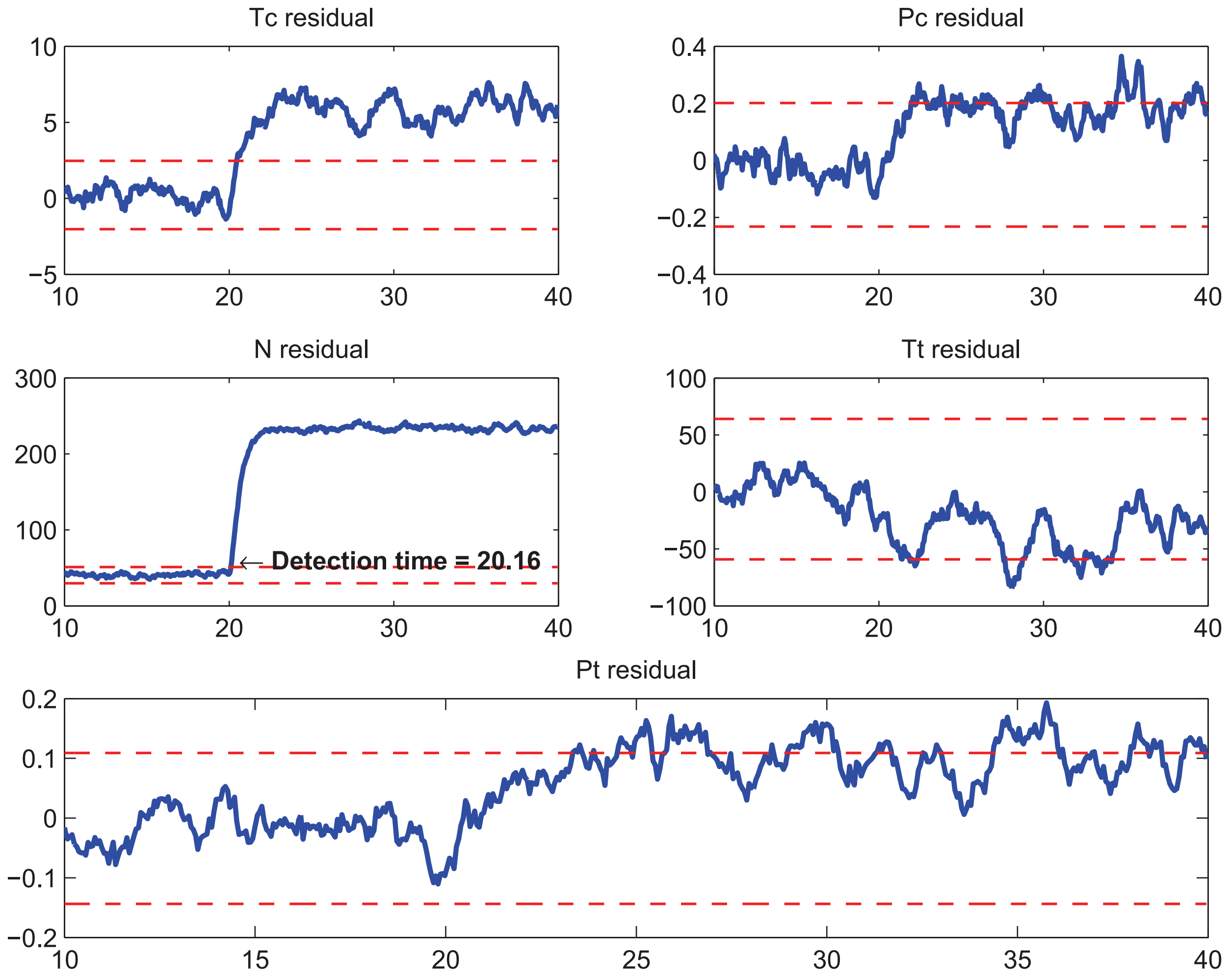


Figure 3.72: Residual generated using ensemble model with FSS pruning, 6% decrease in turbine efficiency at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.



Residual signals for the turbine efficiency fault: fault magnitude = 6

Figure 3.73: Residual generated using RBF-NARX model, 6% decrease in turbine efficiency at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.



Residual signals for the turbine efficiency: fault magnitude = 4

Figure 3.74: Residual generated using ensemble model with FSS pruning, 4% decrease in turbine efficiency at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.

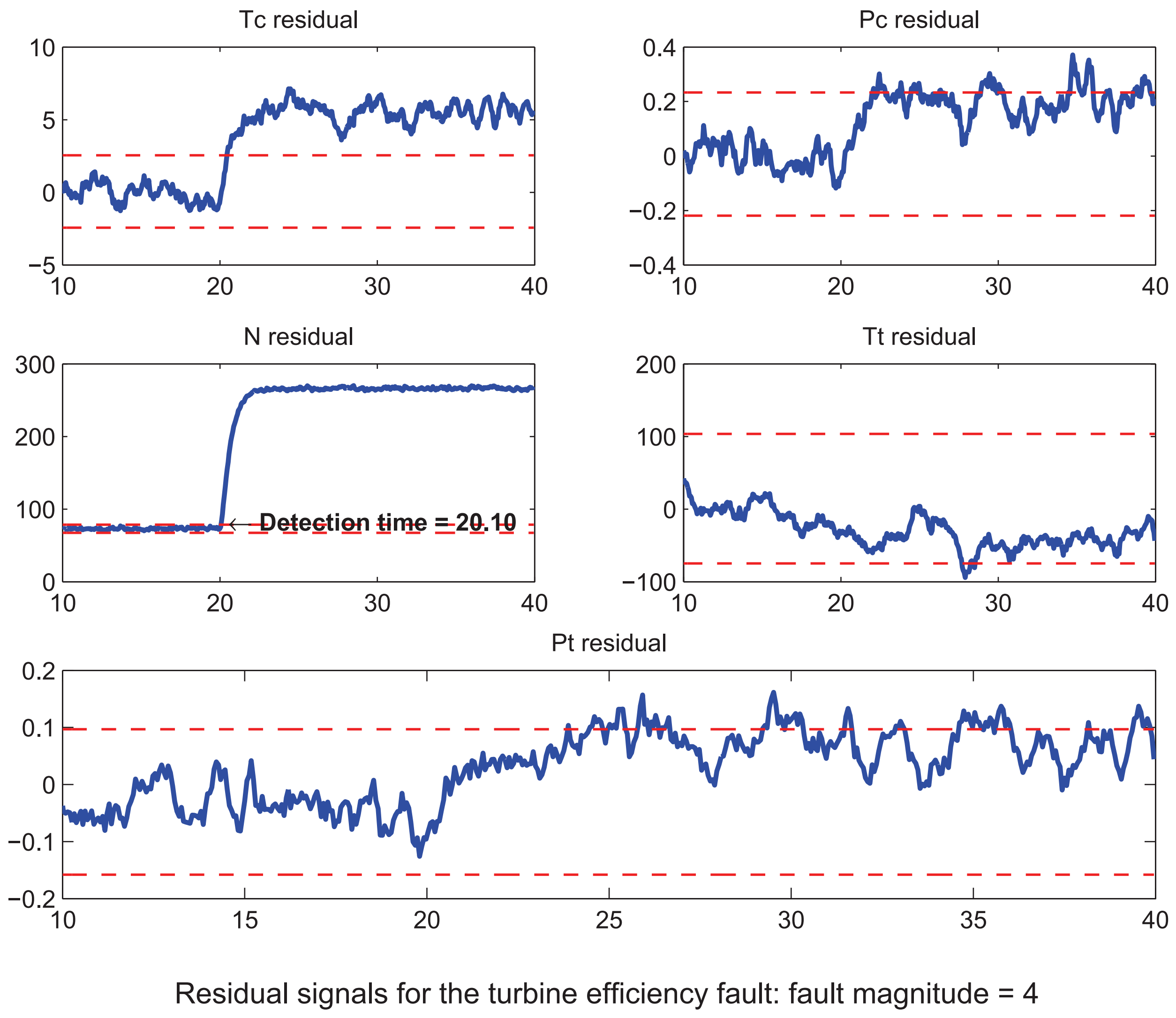
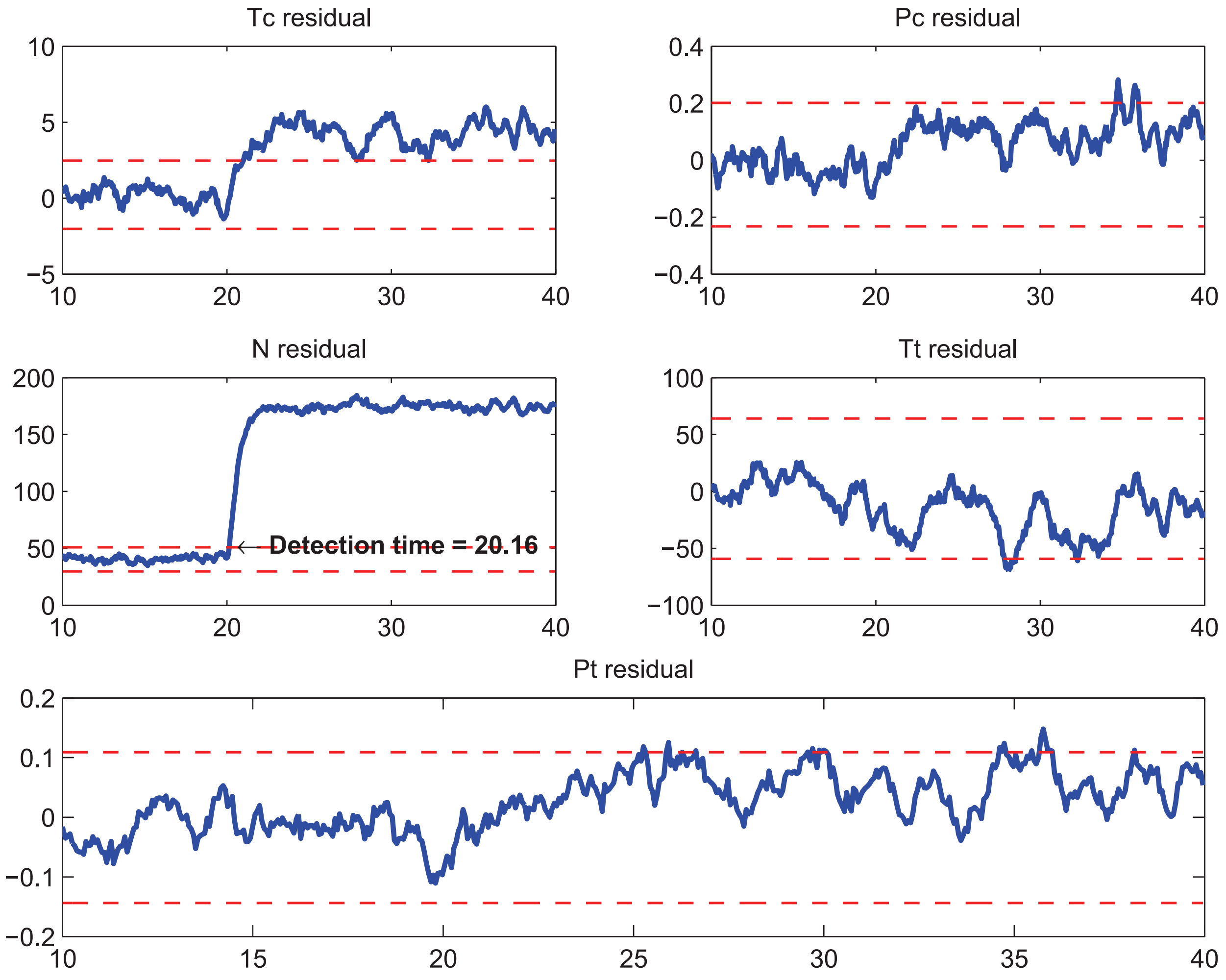


Figure 3.75: Residual generated using RBF-NARX model, 4% decrease in turbine efficiency at t = 20 sec, $\dot{m}_f$ = $0.85 \quad \dot{m}_{f,maximum}$.



Residual signals for the turbine efficiency: fault magnitude = 2

Figure 3.76: Residual generated using ensemble model with FSS pruning, 2% decrease in turbine efficiency at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.

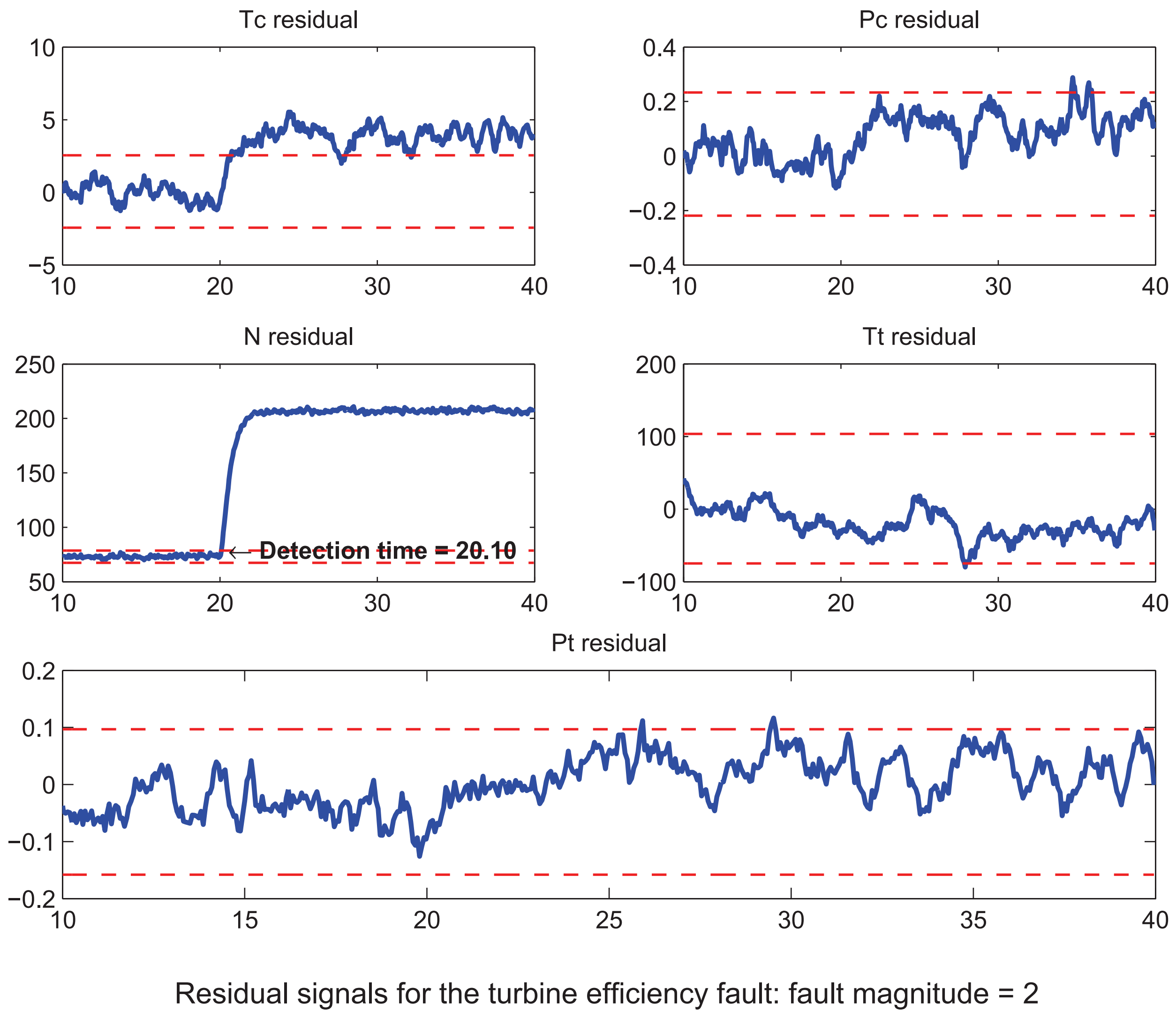


Figure 3.77: Residual generated using RBF-NARX model, 2% decrease in turbine efficiency at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.

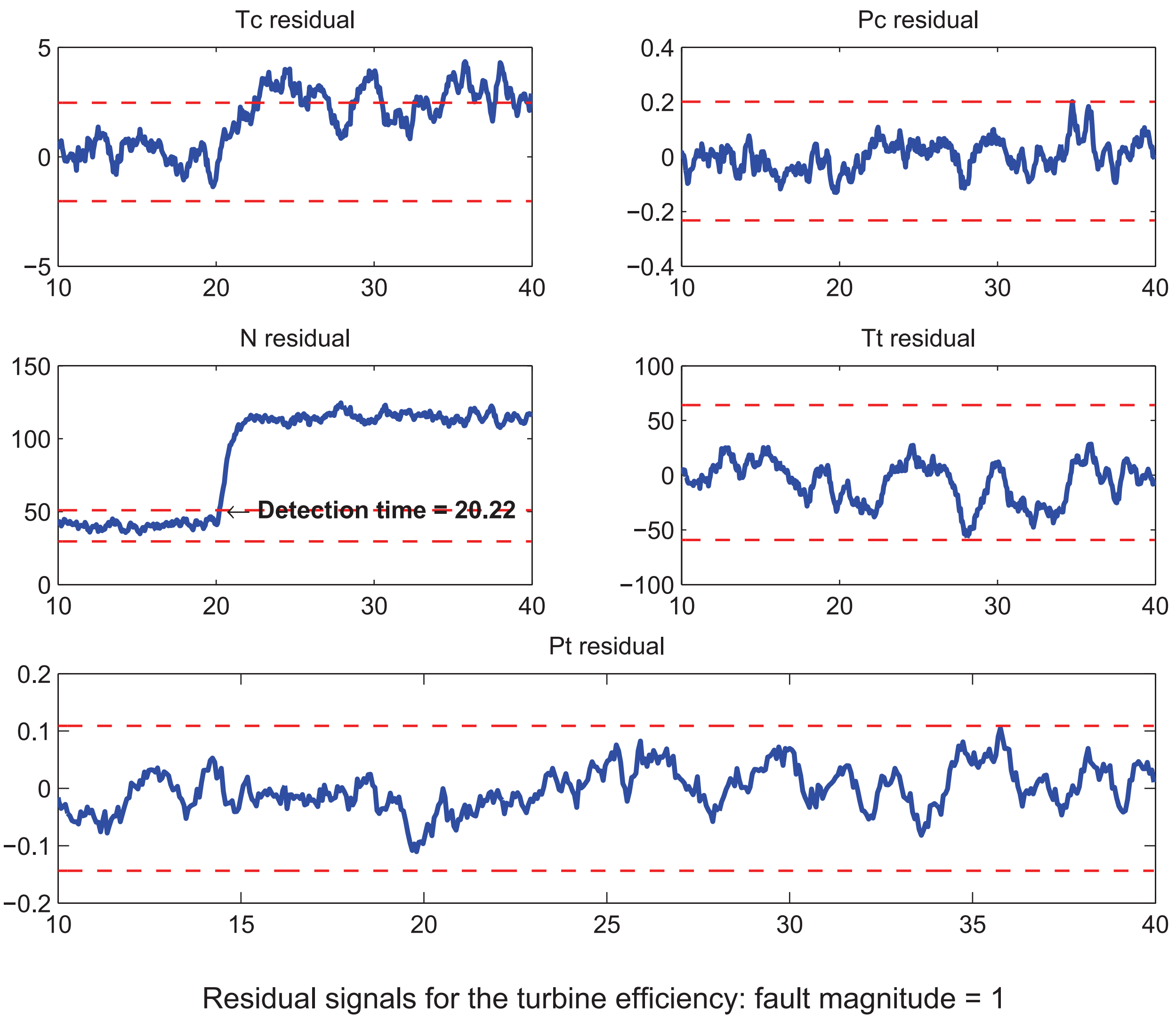


Figure 3.78: Residual generated using ensemble model with FSS pruning, 1% decrease in turbine efficiency at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.

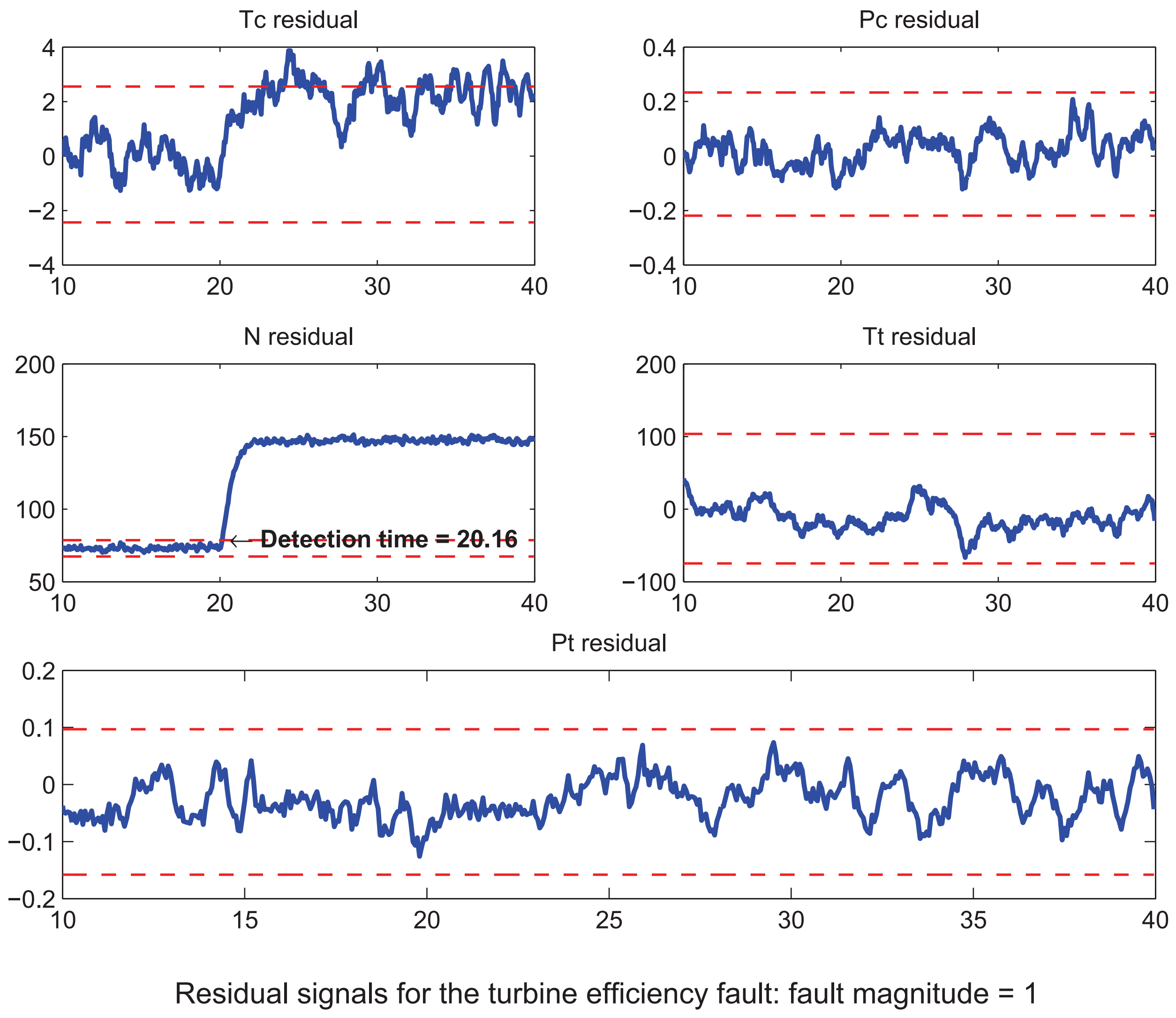


Figure 3.79: Residual generated using RBF-NARX model, 1% decrease in turbine efficiency at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.

Section 3.8.8 shows a comparative study between the fault detection results of the ensemble solution and the single-model solution which indicates an improvement in fault detection accuracy using the ensemble model. The failure has happened at $t = 20$ sec. Tables 3.64 and 3.65 summarize the fault detection time using ensemble model and RBF-NARX models respectively. Figures 3.60, 3.62, 3.64, 3.66, 3.68 show the fault in compressor efficiency detected using the ensemble model. Figures 3.61, 3.63, 3.65, 3.67, 3.69 show the fault in compressor efficiency detected using the single-model based solution (i.e. RBF-NARX model).

Table 3.64: Fault detection time summary using ensemble model: turbine efficiency fault injected at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.

Fault Type	Fault Severity	Fuel Rate	Injection Time	Detection Time
F_{et}	8%	$\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$	20	20.10
F_{et}	6%	$\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$	20	20.10
F_{et}	4%	$\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$	20	20.16
F_{et}	2%	$\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$	20	20.16
F_{et}	1%	$\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$	20	20.22

Table 3.65: Fault detection time summary using RBF-NARX model: turbine efficiency fault injected at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.

Fault Type	Fault Severity	Fuel Rate	Injection Time	Detection Time
F_{et}	8%	$\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$	20	20.10
F_{et}	6%	$\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$	20	20.10
F_{et}	4%	$\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$	20	20.10
F_{et}	2%	$\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$	20	20.16
F_{et}	1%	$\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$	20	20.16

3.8.7 Scenario IV: Fault in the Turbine Mass Flow

In this scenario we assume that there is a decrease in the effectiveness of turbine mass flow rate. The failure happens with different magnitudes that are 1%, 2%, 4%, 6%, and 8%. Also the fuel rate varies between 70%, 75%, 80%, 90%, and 95% of the maximum in each case. The residual signals are generated for the individual learning algorithms as well as the ensemble system. Figures 3.80, 3.82, 3.84, 3.86, 3.88 show the fault in compressor efficiency detected using the ensemble model.

Figures 3.81, 3.83, 3.85, 3.87, 3.89 show the fault in compressor efficiency detected using the single-model based solution (i.e. RBF-NARX model).

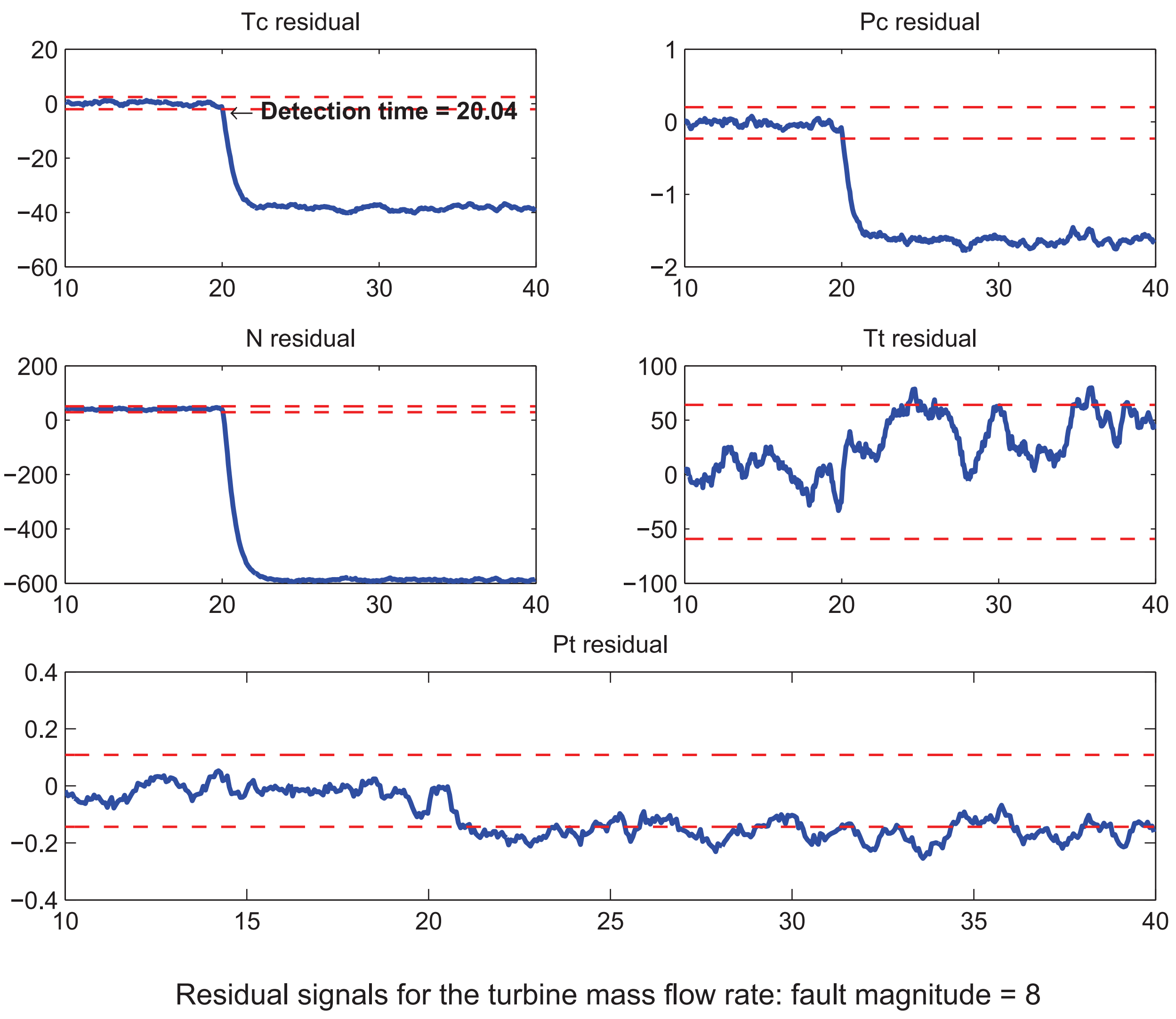


Figure 3.80: Residual generated using ensemble model with FSS pruning, 8% decrease in turbine efficiency at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.

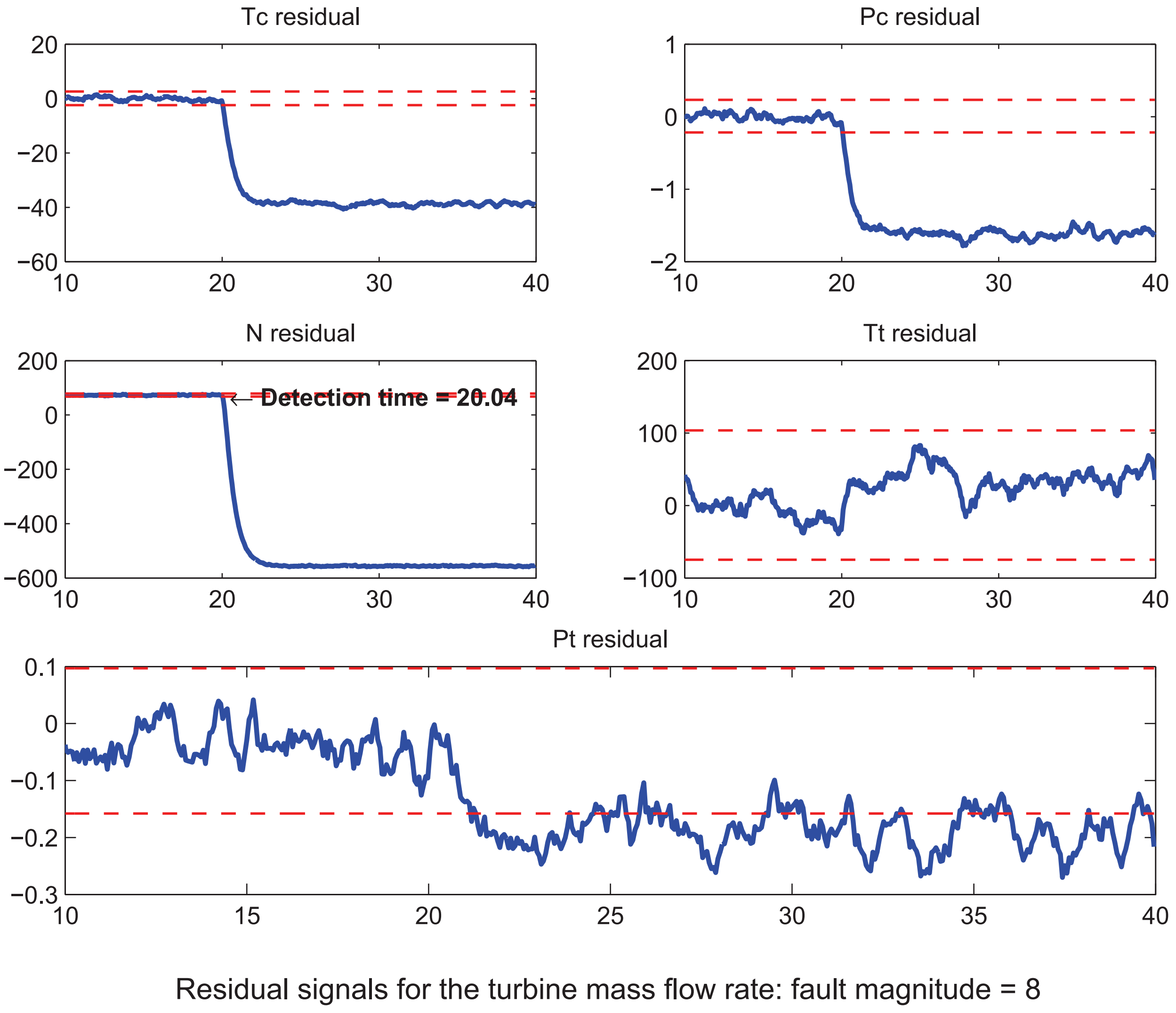


Figure 3.81: Residual generated using RBF-NARX model, 8% decrease in turbine efficiency at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.

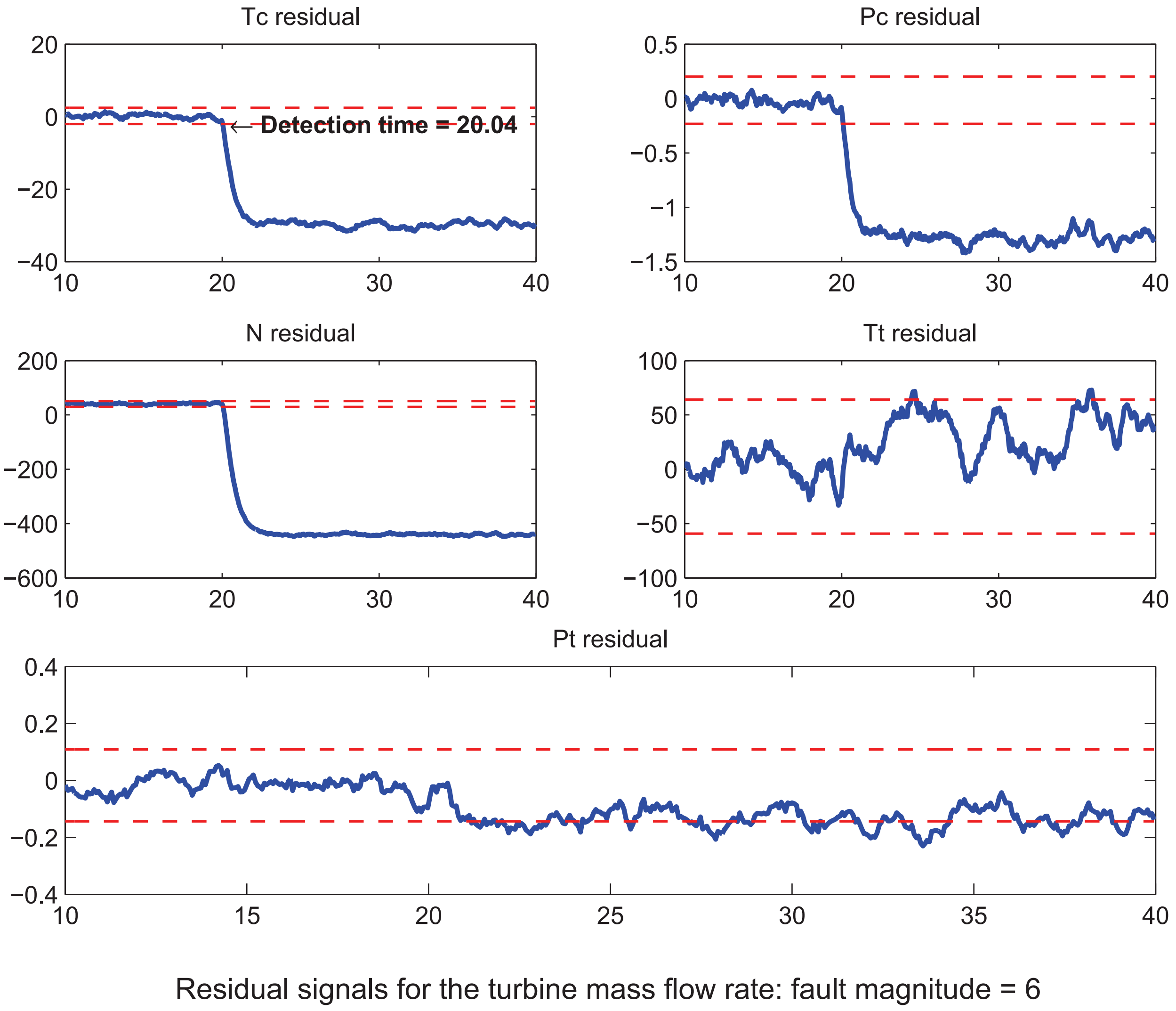


Figure 3.82: Residual generated using ensemble model with FSS pruning, 6% decrease in turbine efficiency at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.

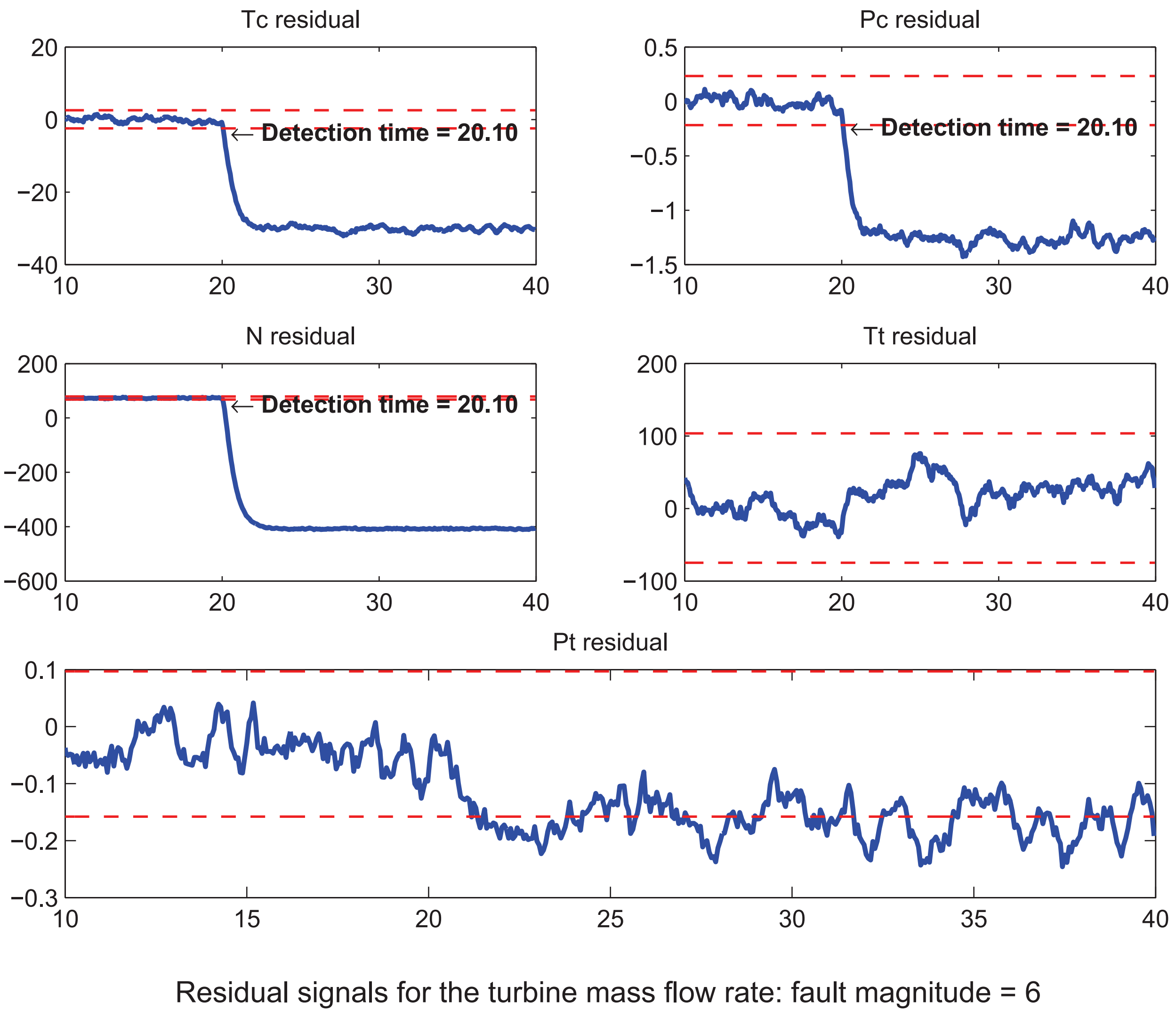
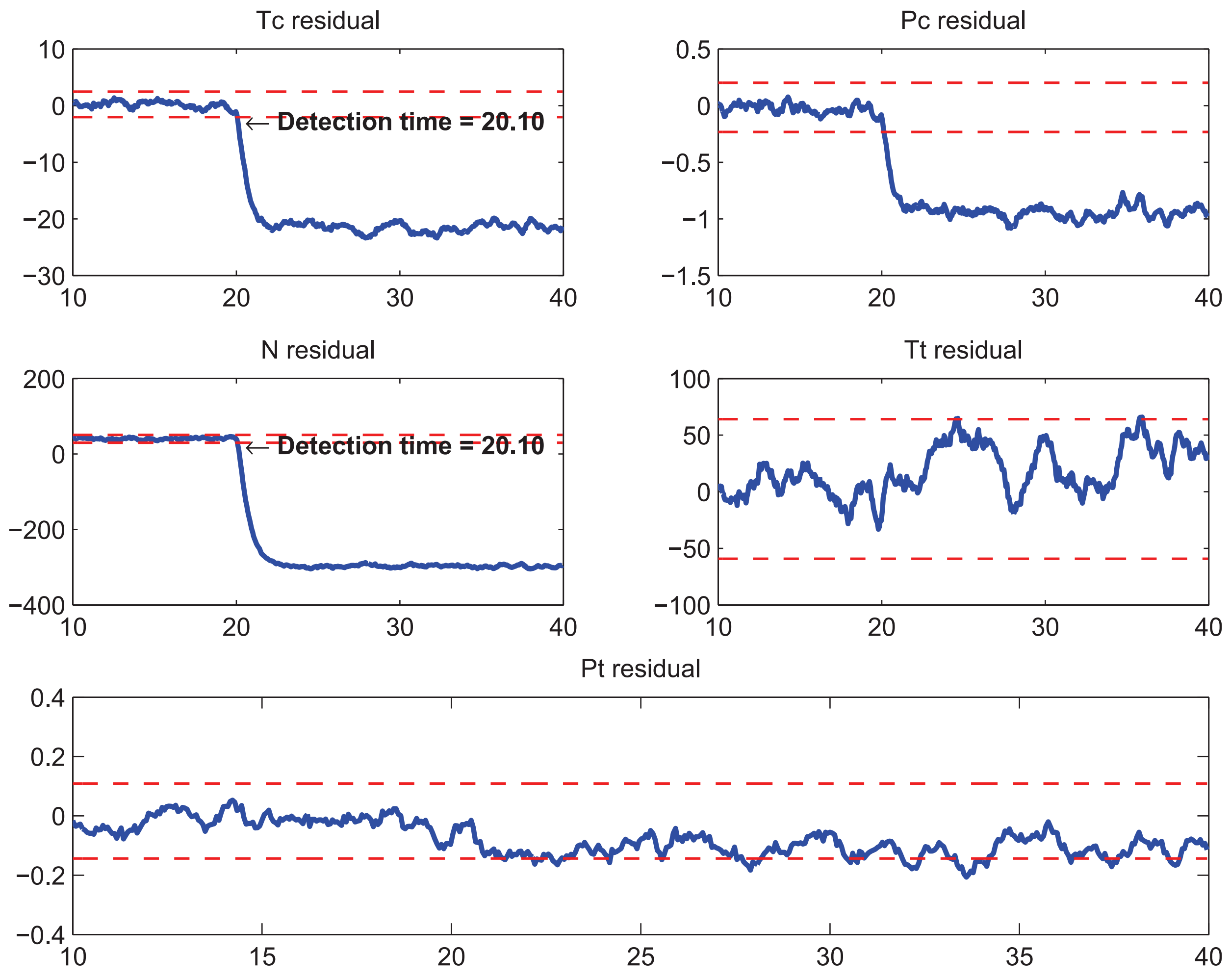


Figure 3.83: Residual generated using RBF-NARX model, 6% decrease in turbine efficiency at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.



Residual signals for the turbine mass flow rate: fault magnitude = 4

Figure 3.84: Residual generated using ensemble model with FSS pruning, 4% decrease in turbine mass flow rate at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.

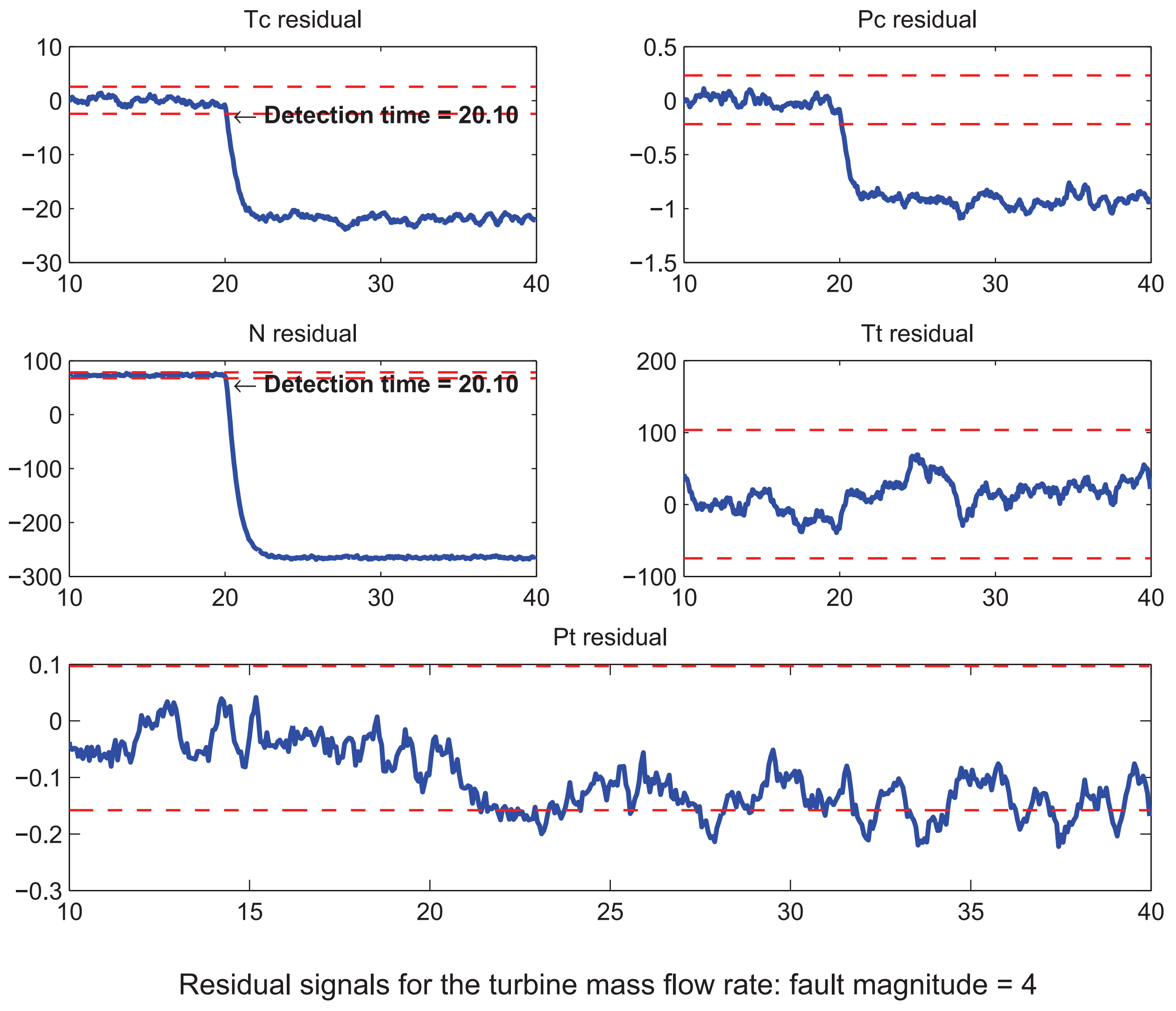


Figure 3.85: Residual generated using RBF-NARX model, 4% decrease in turbine mass flow rate at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.

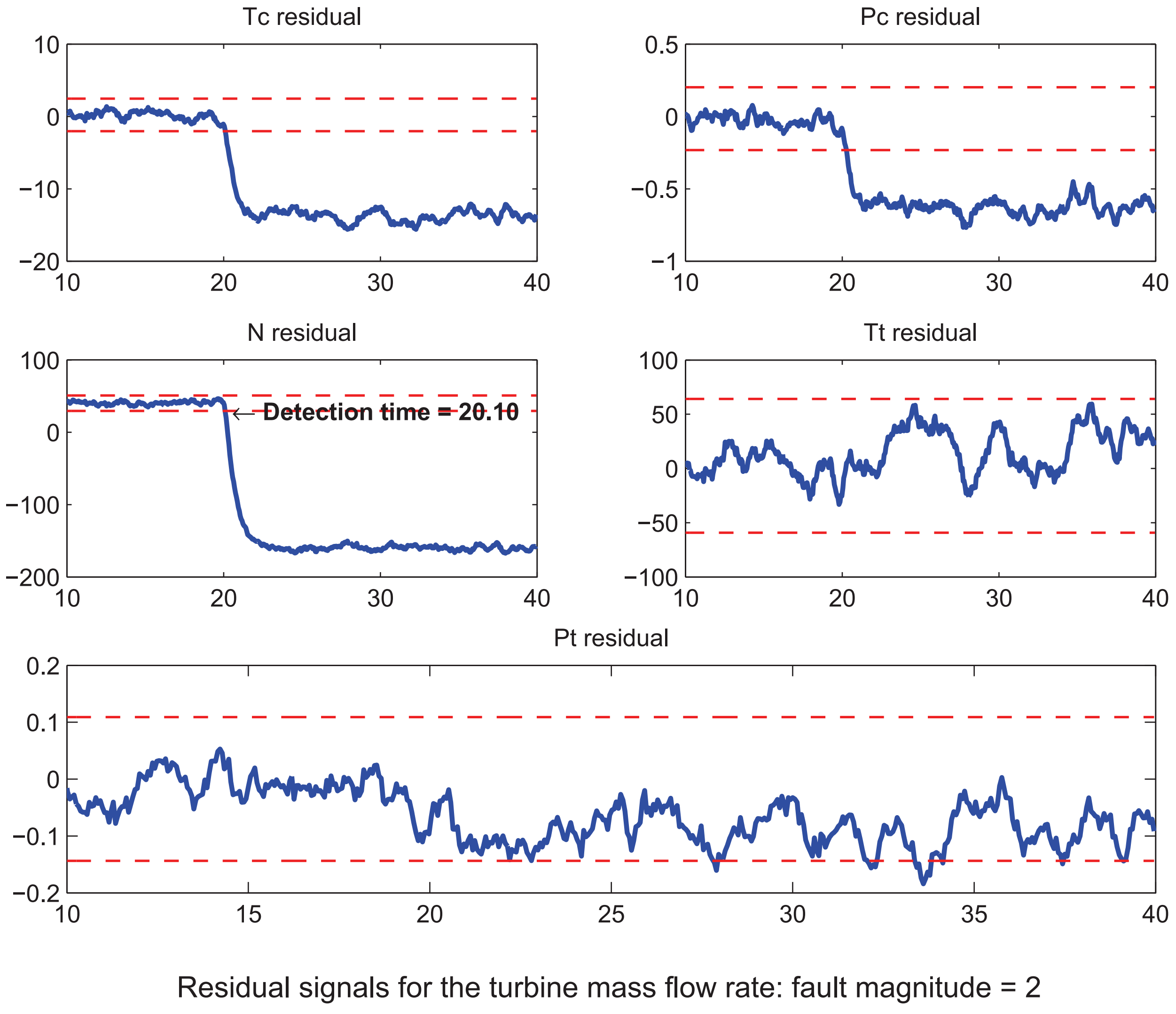
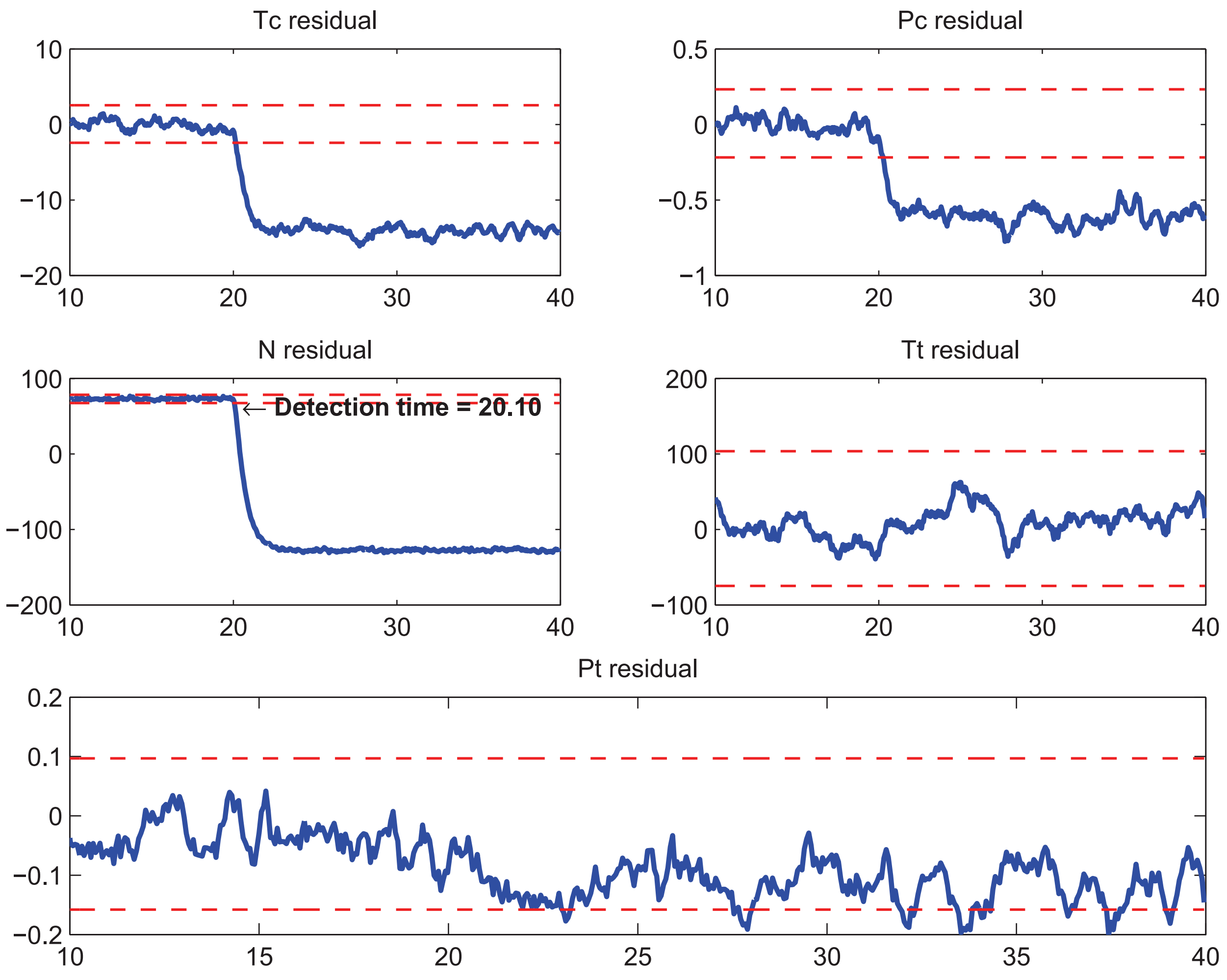


Figure 3.86: Residual generated using ensemble model with FSS pruning, 2% decrease in turbine efficiency at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.



Residual signals for the turbine mass flow rate: fault magnitude = 2

Figure 3.87: Residual generated using RBF-NARX model, 2% decrease in turbine efficiency at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.

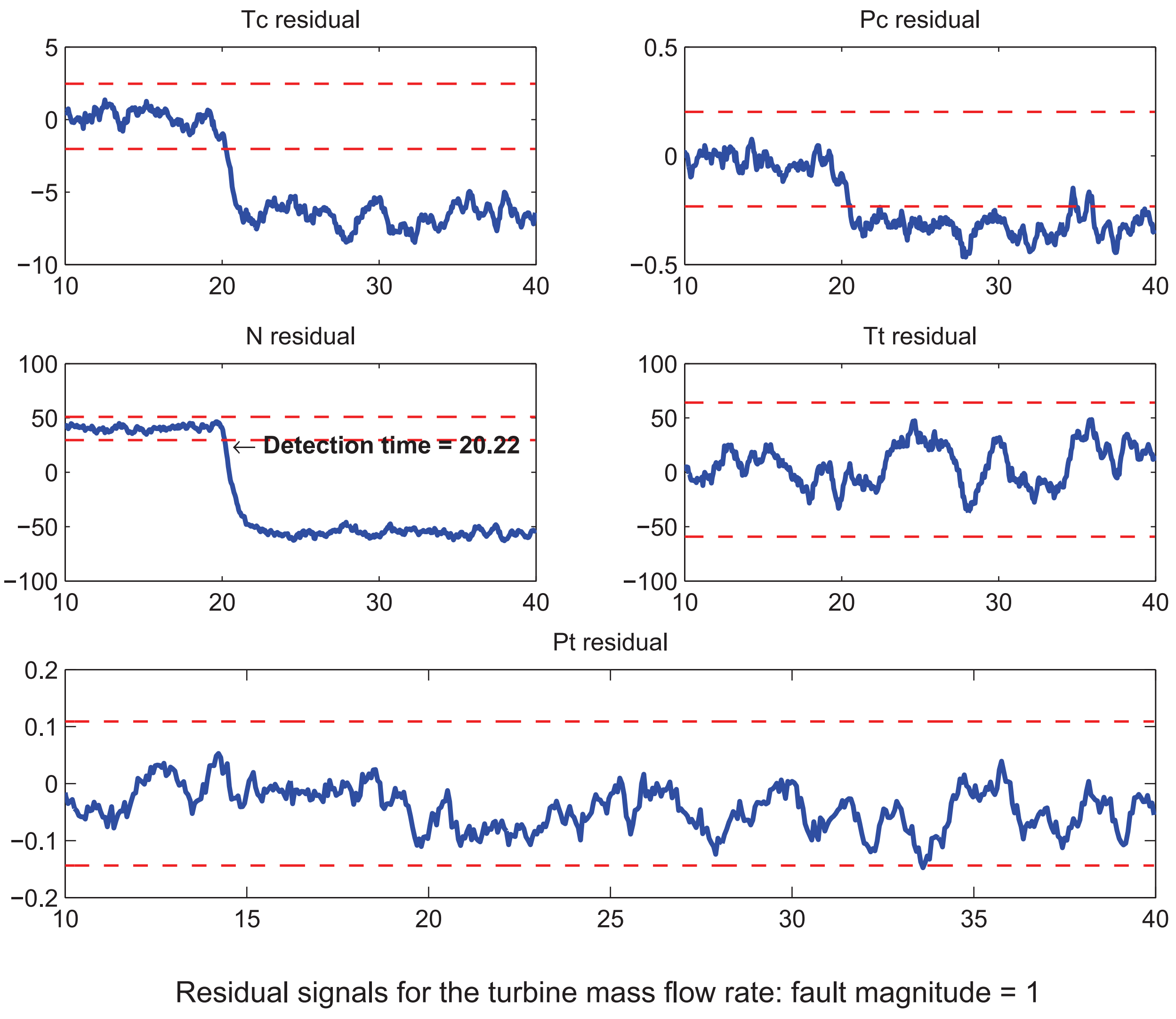


Figure 3.88: Residual generated using ensemble model with FSS pruning, 1% decrease in turbine efficiency at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.

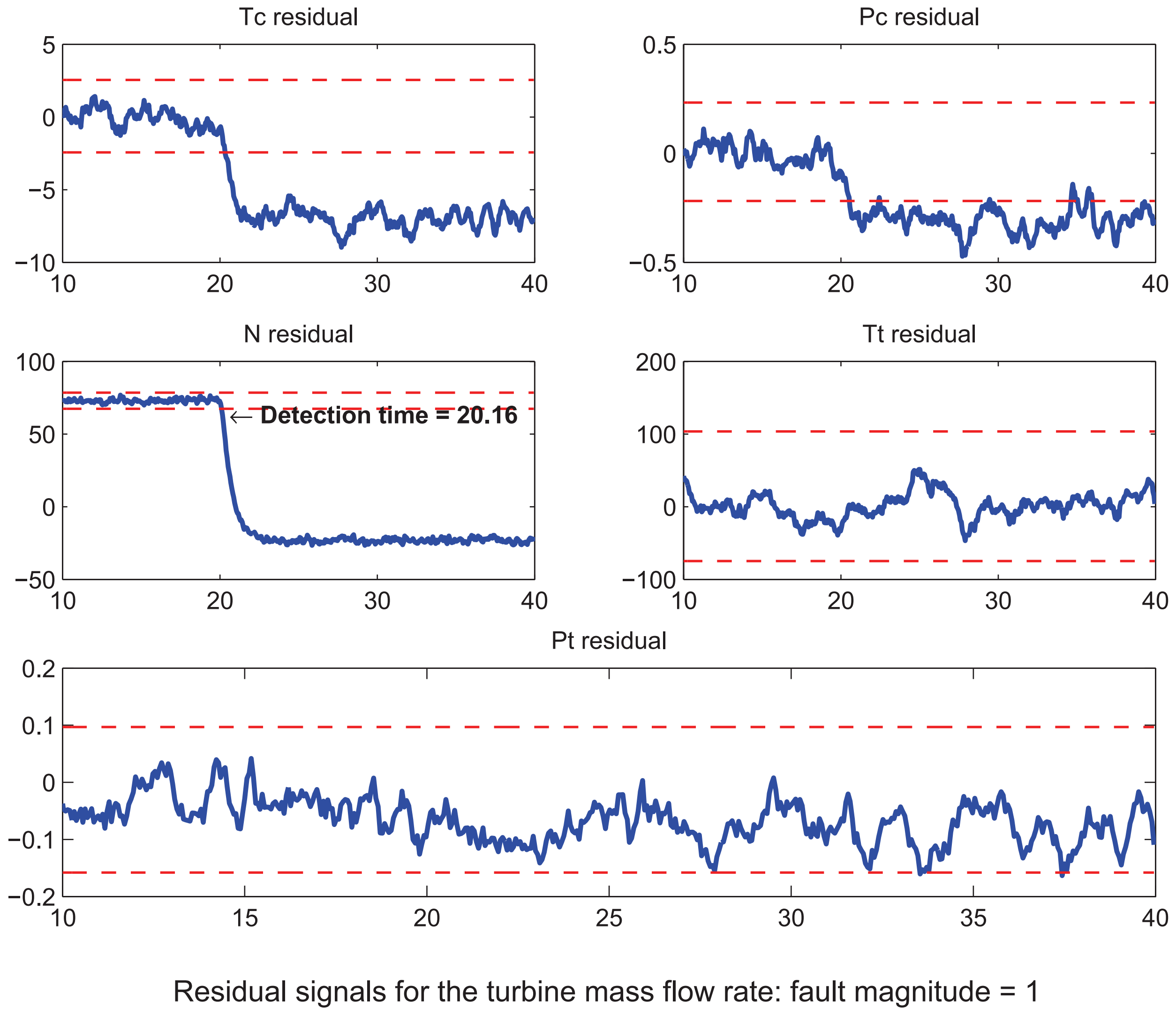


Figure 3.89: Residual generated using RBF-NARX model, 1% decrease in turbine efficiency at t = 20 sec, $\dot{m}_f = 0.85 \quad \dot{m}_{f,maximum}$.

Section 3.8.8 shows a comparative study between the fault detection results of the ensemble solution and the single-model solution which indicates an improvement in fault detection accuracy using the ensemble model. The failure has happened at $t = 20$ sec. Tables 3.66 and 3.67 summarize the fault detection time using ensemble model and RBF-NARX models respectively.

Table 3.66: Fault detection time summary using ensemble model: turbine mass ow rate fault injected at t = 20 sec, $\dot{m}_f$ = $0.85 \quad \dot{m}_{f,maximum}$.

Fault Type	Fault Severity	Fuel Rate	Injection Time	Detection Time
F_{et}	8%	$\dot{m}_f$ = 0.85 $\quad \dot{m}_{f,maximum}$	20	20.10
F_{et}	6%	$\dot{m}_f$ = 0.85 $\quad \dot{m}_{f,maximum}$	20	20.10
F_{et}	4%	$\dot{m}_f$ = 0.85 $\quad \dot{m}_{f,maximum}$	20	20.16
F_{et}	2%	$\dot{m}_f$ = 0.85 $\quad \dot{m}_{f,maximum}$	20	20.16
F_{et}	1%	$\dot{m}_f$ = 0.85 $\quad \dot{m}_{f,maximum}$	20	20.22

Table 3.67: Fault detection time summary using RBF-NARX model: turbine efficiency fault injected at t = 20 sec, $\dot{m}_f$ = $0.85 \quad \dot{m}_{f,maximum}$.

Fault Type	Fault Severity	Fuel Rate	Injection Time	Detection Time
F_{et}	8%	$\dot{m}_f$ = 0.85 $\quad \dot{m}_{f,maximum}$	20	20.10
F_{et}	6%	$\dot{m}_f$ = 0.85 $\quad \dot{m}_{f,maximum}$	20	20.10
F_{et}	4%	$\dot{m}_f$ = 0.85 $\quad \dot{m}_{f,maximum}$	20	20.10
F_{et}	2%	$\dot{m}_f$ = 0.85 $\quad \dot{m}_{f,maximum}$	20	20.16
F_{et}	1%	$\dot{m}_f$ = 0.85 $\quad \dot{m}_{f,maximum}$	20	20.16

3.8.8 Confusion Matrices and FD Analysis

As previously stated in Remark 2.4 turbine temperature may not be instrumented due to very high temperature of turbine. Though recent research has focused on development of new thermocouple which are capable of measuring turbine temperature with required accuracy. Thus, in this section we perform fault detection analysis within two different scenarios (a) turbine temperature is *measurable* (i.e. $[RES_{P_C}, RES_{T_C}, RES_N, RES_{P_T}, RES_{T_T}]$ is used for fault detection), and (b) turbine temperature is *not measurable* (i.e. $[RES_{P_C}, RES_{T_C}, RES_N, RES_{P_T}]$ is used for fault detection).

Below we summarize the fault detection results for both ensemble-based and single model-based solution in form of confusion matrices. A confusion matrix for fault detection is generally a table with four entries which are the number of true positive, true negative, false positive, and false negative classifications which are defined as follows:

- True positive (t.p.): the number of simulations classified as faulty and the engine is also faulty.
- False positive (f.p.): the number of simulations classified as faulty, but the engine is healthy.
- True negative (t.n.): the number of simulations classified as healthy and the engine is healthy.
- False negative (f.n.): the number of simulations classified as healthy but the engine is faulty.

The generic form of a confusion matrix is shown in Table 3.68.

Table 3.68: Confusion matrix general form.

	Alarm	No Alarm
Faulty	True positive	False negative
Healthy	False positive	True negative

Correct Classiˇcation Ration (CCR) (also known as *accuracy*) is presented as a measure to evaluate the *accuracy* of the fault detection which is defined as follows:

$$CCR = \frac{t.p. \quad t.n}{t.p. \quad t.n. \quad f.p. \quad f.n.}$$

Other classifier performance measures are defined as follows:

$$\begin{aligned}
Precision &= \frac{t.n}{t.n. \quad f.n.}\\
\textit{True Positive Rate (TPR)} &= \frac{t.p}{t.p. \quad f.n.}\\
\textit{False Positive Rate (FPR)} &= \frac{f.p}{f.p. \quad t.n.}\\
\textit{True Negative Rate (TNR)} &= \frac{t.n}{t.n. \quad f.p.}\\
\textit{False Negative Rate (FNR)} &= \frac{f.n}{t.p. \quad f.n.}
\end{aligned}$$

In this section a total of 200 simulations are performed under different engine operating conditions where a total of 100 experiments are made under presence of fault, and the other 100 are performed while the engine is operating under healthy condition. The 100 faulty simulations are conducted under presence of different fault types with different fault severities.

The fault types considered in this section are the component faults which were previously described in Section 2.3.3. The first considered fault scenario is the fault in the compressor mass flow rate which is an indication of the fouling in the compressor. The considered fault severities are 1%, 2%, 4%, 6% and 8% reduction in the compressor mass flow rate. Several simulations are made while the engine operates under different fuel flow rates (i.e. 70%, 75%, 80% and 85% of the maximum fuel rate). The second considered fault scenario is the fault in the compressor efficiency which is an indication of the erosion in the compressor. The considered fault severities are 1%, 2%, 4%, 6% and 8% reduction in the compressor efficiency. Several simulations are made while the engine operates under different fuel flow rates (i.e. 70%, 75%, 80% and 85% of the maximum fuel rate). The third considered fault scenario is the fault in the turbine mass flow rate which is an indication of the turbine fouling. The considered fault severities are 1%, 2%, 4%, 6% and 8% reduction in the turbine mass flow rate. Several simulations are made while the engine operates under different fuel flow rates (i.e. 70%, 75%, 80% and 85% of the maximum fuel rate). The fourth considered fault scenario is the fault in the turbine efficiency which is an indication of the erosion in the turbine. The considered fault severities are 1%, 2%, 4%, 6% and 8% reduction in the turbine efficiency. Similar to the previous scenarios, several simulations are made while the engine operates under different fuel flow rates (i.e. 70%, 75%, 80% and 85% of the maximum fuel rate).

A total of 25 simulations are considered under presence of each of the above mentioned faults. This results in a total of 100 simulations under a fault presence. Another 100 simulations are considered while the engine operates in healthy condition. The summary of fault simulations is presented in Table 3.69.

Table 3.69: Summary of fault simulations.

Fault type	Fault simulation detail	# of tests
F_{mc}	Considered fault severities are 1%, 2%, 4%, 6%, and 8% decrease in compressor flow capacity ($\dot{m}_C$). 5 simulations are conducted for each fault severity (under different fuel flow rates)	25
F_{ec}	Considered fault severities are 1%, 2%, 4%, 6%, and 8% decrease in compressor efficiency (η_C). 5 simulations are conducted for each fault severity (under different fuel flow rates)	25
F_{mt}	Considered fault severities are 1%, 2%, 4%, 6%, and 8% decrease in turbine flow capacity ($\dot{m}_T$). 5 simulations are conducted for each fault severity (under different fuel flow rates)	25
F_{et}	Considered fault severities are 1%, 2%, 4%, 6%, and 8% decrease in turbine efficiency (η_T). 5 simulations are conducted for each fault severity (under different fuel flow rates)	25
–	Engine operates in healthy condition	100
Total		**200**

The obtained results are presented in the following confusion matrices (refer to Tables 3.70 to 3.86 for confusion matrices based on single model-based solution and Tables 3.72 to 3.88 for confusion matrices based on ensemble-based solution) for each fault severity. More detailed description is presented in the following subsection. A comparison between the results of the single model-based (i.e. RBF-NARX model) fault detection and the ensemble-based (i.e. heterogenous ensemble with FSS pruning) fault detection illustrates the advantage of ensemble-based solution. We observe that for a total of 200 simulations (25 simulations for each fault type) the fault detection accuracy is increased from 85.5% for single model-based fault detection to 90.5% for ensemble-based fault detection.

Remark 3.9. *For each engine parameter, the structure of the RBF-NARX networks used for fault detection simulations is the same structure as obtained in Section 3.5 through extensive simulations (refer to Tables 3.20, 3.22, 3.24, 3.28 and 3.26 for the structures). Moreover, for each engine parameter, the ensemble architecture used for fault detection simulations is heterogenous ensemble with FSS pruning as presented in Section 3.7.5 (refer to Tables 3.45, 3.46, 3.48, 3.49 and 3.47). We should also note that (for brevity purpose) other single model-based solutions (i.e. MLP-NARX and SVM-NARX) are not considered for further fault detection simulations as they showed a less promising performance in term of modeling accuracy as compared with RBF-NARX model. This is also true for the ensemble-based solution as we only use the ensemble architecture with better performance (in term of engine parameters modeling) for fault detection simulations.*

3.8.9 Fault Detection Performance Assuming Measurable Turbine Temperature

This section evaluates the fault detection performance assuming that the turbine temperature is measurable. Confusion matrices are presented for different fault severities. Also different measures for evaluating the performance of the fault detection are presented for each fault severity. Table 3.70 presents the confusion matrices for different fault scenarios based on the single model-based solution when the fault severity is equal to 1%. Table 3.71 shows different measures for evaluating the performance of the fault detection task based on the residuals obtained from the single model-based solution when the fault severity is equal to 1%.

Table 3.70: Confusion matrixes for different fault scenarios based on single model-based solution (RBF-NARX model). Fault severity = 1%.

Decrease in compressor mass flow rate.

	Alarm	No Alarm
Faulty	2	3
Healthy	1	4

Decrease in compressor efficiency.

	Alarm	No Alarm
Faulty	4	1
Healthy	0	5

Decrease in turbine mass flow rate.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine efficiency.

	Alarm	No Alarm
Faulty	4	1
Healthy	0	5

Table 3.71: Fault detection accuracy of the single model-based (i.e. RBF-NARX) solution. Fault severity = 1%.

	F_{mc}	F_{ec}	F_{mt}	F_{et}
CCR	60%	90%	100%	90%
Precision	57.14%	83%	100%	83%
TPR	40%	80%	100%	80%
FPR	20%	0%	0%	0%
TNR	80%	100%	100%	100%
FNR	60%	20%	0%	20%

Table 3.72 presents the confusion matrices for different fault scenarios based on the ensemble-based solution when the fault severity is equal to 1%. Table 3.73 shows different measures for evaluating the performance of the fault detection task based on the residuals obtained from the ensemble-based solution when the fault severity is equal to 1%.

Table 3.72: Confusion matrixes for different fault scenarios based on ensemble-based (i.e. heterogenous ensemble with FSS pruning) solution. Fault severity = 1%.

Decrease in compressor mass flow rate.

	Alarm	No Alarm
Faulty	5	0
Healthy	3	2

Decrease in compressor efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine mass flow rate.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Table 3.73: Fault detection accuracy of the ensemble-based solution. Fault severity = 1%.

	F_{mc}	F_{ec}	F_{mt}	F_{et}
CCR	70%	100%	100%	100%
Precision	100%	100%	100%	100%
TPR	100%	100%	100%	100%
FPR	60%	0%	0%	0%
TNR	40%	100%	100%	100%
FNR	0%	0%	0%	0%

Table 3.74 presents the confusion matrices for different fault scenarios based on the single model-based solution when the fault severity is equal to 2%. Table 3.75 shows different measures for evaluating the performance of the fault detection task based on the residuals obtained from the single model-based solution when the fault severity is equal to 2%.

Table 3.74: Confusion matrixes for different fault scenarios based on single model-based solution (RBF-NARX model). Fault severity = 2%.

Decrease in compressor mass flow rate.

	Alarm	No Alarm
Faulty	4	2
Healthy	1	3

Decrease in compressor efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine mass flow rate.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Table 3.75: Fault detection accuracy of the single model-based (i.e. RBF-NARX) solution. Fault severity = 2%.

	F_{mc}	F_{ec}	F_{mt}	F_{et}
CCR	70%	100%	100%	100%
Precision	60%	100%	100%	100%
TPR	67%	100%	100%	100%
FPR	25%	0%	0%	0%
TNR	75%	100%	100%	100%
FNR	33%	0%	0%	0%

Table 3.76 presents the confusion matrices for different fault scenarios based on the ensemble-based solution when the fault severity is equal to 2%. Table 3.77 shows different measures for evaluating the performance of the fault detection task based on the residuals obtained from the ensemble-based solution when the fault severity is equal to 2%.

Table 3.76: Confusion matrixes for different fault scenarios based on ensemble-based (i.e. heterogenous ensemble with FSS pruning) solution. Fault severity = 2%.

Decrease in compressor mass flow rate.

	Alarm	No Alarm
Faulty	4	1
Healthy	1	4

Decrease in compressor efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine mass flow rate.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Table 3.77: Fault detection accuracy of the ensemble-based solution. Fault severity = 2%.

	F_{mc}	F_{ec}	F_{mt}	F_{et}
CCR	80%	100%	100%	100%
Precision	80%	100%	100%	100%
TPR	80%	100%	100%	100%
FPR	20%	0%	0%	0%
TNR	80%	100%	100%	100%
FNR	20%	0%	0%	0%

Table 3.78 presents the confusion matrices for different fault scenarios based on the single model-based solution when the fault severity is equal to 4%. Table 3.79 shows different measures for evaluating the performance of the fault detection task based on the residuals obtained from the single model-based solution when the fault severity is equal to 4%.

Table 3.78: Confusion matrixes for different fault scenarios based on single model-based solution (RBF-NARX model). Fault severity = 4%.

Decrease in compressor mass flow rate.

	Alarm	No Alarm
Faulty	5	0
Healthy	1	4

Decrease in compressor efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine mass flow rate.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Table 3.79: Fault detection accuracy of the single model-based (i.e. RBF-NARX) solution. Fault severity = 4%.

	F_{mc}	F_{ec}	F_{mt}	F_{et}
CCR	90%	100%	100%	100%
Precision	100%	100%	100%	100%
TPR	100%	100%	100%	100%
FPR	20%	0%	0%	0%
TNR	80%	100%	100%	100%
FNR	0%	0%	0%	0%

Table 3.80 presents the confusion matrices for different fault scenarios based on the ensemble-based solution when the fault severity is equal to 4%. Table 3.81 shows different measures for evaluating the performance of the fault detection task based on the residuals obtained from the ensemble-based solution when the fault severity is equal to 4%.

Table 3.80: Confusion matrixes for different fault scenarios based on ensemble-based (i.e. heterogenous ensemble with FSS pruning) solution. Fault severity = 4%.

Decrease in compressor mass flow rate.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in compressor efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine mass flow rate.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Table 3.81: Fault detection accuracy of the ensemble-based solution. Fault severity = 4%.

	F_{mc}	F_{ec}	F_{mt}	F_{et}
CCR	100%	100%	100%	100%
Precision	100%	100%	100%	100%
TPR	100%	100%	100%	100%
FPR	0%	0%	0%	0%
TNR	80%	100%	100%	100%
FNR	100%	0%	0%	0%

Table 3.82 presents the confusion matrices for different fault scenarios based on the single model-based solution when the fault severity is equal to 6%. Table 3.83 shows different measures for evaluating the performance of the fault detection task based on the residuals obtained from the single model-based solution when the fault severity is equal to 6%.

Table 3.82: Confusion matrixes for different fault scenarios based on single model-based solution (RBF-NARX model). Fault severity = 6%.

Decrease in compressor mass flow rate.

	Alarm	No Alarm
Faulty	5	0
Healthy	1	4

Decrease in compressor efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine mass flow rate.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Table 3.83: Fault detection accuracy of the single model-based (i.e. RBF-NARX) solution. Fault severity = 6%.

	F_{mc}	F_{ec}	F_{mt}	F_{et}
CCR	90%	100%	100%	100%
Precision	100%	100%	100%	100%
TPR	100%	100%	100%	100%
FPR	20%	0%	0%	0%
TNR	80%	100%	100%	100%
FNR	0%	0%	0%	0%

Table 3.84 presents the confusion matrices for different fault scenarios based on the ensemble-based solution when the fault severity is equal to 6%. Table 3.85 shows different measures for evaluating the performance of the fault detection task based on the residuals obtained from the ensemble-based solution when the fault severity is equal to 6%.

Table 3.84: Confusion matrixes for different fault scenarios based on ensemble-based (i.e. heterogenous ensemble with FSS pruning) solution. Fault severity = 6%.

Decrease in compressor mass flow rate.

	Alarm	No Alarm
Faulty	4	0
Healthy	1	5

Decrease in compressor efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine mass flow rate.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Table 3.85: Fault detection accuracy of the ensemble-based solution. Fault severity = 6%.

	F_{mc}	F_{ec}	F_{mt}	F_{et}
CCR	90%	100%	100%	100%
Precision	100%	100%	100%	100%
TPR	100%	100%	100%	100%
FPR	17%	0%	0%	0%
TNR	83%	100%	100%	100%
FNR	0%	0%	0%	0%

Table 3.86 presents the confusion matrices for different fault scenarios based on the single model-based solution when the fault severity is equal to 8%. Table 3.87 shows different measures for evaluating the performance of the fault detection task based on the residuals obtained from the single model-based solution when the fault severity is equal to 8%.

Table 3.86: Confusion matrixes for different fault scenarios based on single model-based solution (RBF-NARX model). Fault severity = 8%.

Decrease in compressor mass flow rate.

	Alarm	No Alarm
Faulty	5	0
Healthy	1	4

Decrease in compressor efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine mass flow rate.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Table 3.87: Fault detection accuracy of the single model-based (i.e. RBF-NARX) solution. Fault severity = 8%.

	F_{mc}	F_{ec}	F_{mt}	F_{et}
CCR	90%	100%	100%	100%
Precision	100%	100%	100%	100%
TPR	100%	100%	100%	100%
FPR	20%	0%	0%	0%
TNR	80%	100%	100%	100%
FNR	0%	0%	0%	0%

Table 3.88 presents the confusion matrices for different fault scenarios based on the ensemble-based solution when the fault severity is equal to 8%. Table 3.89 shows different measures for evaluating the performance of the fault detection task based on the residuals obtained from the ensemble-based solution when the fault severity is equal to 8%.

Table 3.88: Confusion matrixes for different fault scenarios based on ensemble-based (i.e. heterogenous ensemble with FSS pruning) solution. Fault severity = 8%.

Decrease in compressor mass flow rate.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in compressor efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine mass flow rate.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Table 3.89: Fault detection accuracy of the ensemble-based solution. Fault severity = 8%.

	F_{mc}	F_{ec}	F_{mt}	F_{et}
CCR	100%	100%	100%	100%
Precision	100%	100%	100%	100%
TPR	100%	100%	100%	100%
FPR	0%	0%	0%	0%
TNR	100%	100%	100%	100%
FNR	0%	0%	0%	0%

3.8.10 Fault Detection Performance Assuming Unmeasurable Turbine Temperature

This section evaluates the fault detection performance assuming that the turbine temperature is not measurable. Confusion matrices are presented for different fault severities. Also different measures for evaluating the performance of the fault detection are presented for each fault severity.

Table 3.90 presents the confusion matrices for different fault scenarios based on the single model-based solution when the fault severity is equal to 1%. Table 3.91 shows different measures for evaluating the performance of the fault detection task based on the residuals obtained from the single model-based solution when the fault severity is equal to 1%.

Table 3.90: Confusion matrixes for different fault scenarios based on single model-based solution (RBF-NARX model). Fault severity = 1%.

Decrease in compressor mass flow rate.

	Alarm	No Alarm
Faulty	2	3
Healthy	1	4

Decrease in compressor efficiency.

	Alarm	No Alarm
Faulty	4	1
Healthy	0	5

Decrease in turbine mass flow rate.

	Alarm	No Alarm
Faulty	4	1
Healthy	0	5

Decrease in turbine efficiency.

	Alarm	No Alarm
Faulty	4	1
Healthy	0	5

Table 3.91: Fault detection accuracy of the single model-based (i.e. RBF-NARX) solution. Fault severity = 1%.

	F_{mc}	F_{ec}	F_{mt}	F_{et}
CCR	60%	90%	90%	90%
Precision	57%	83%	83%	83%
TPR	40%	80%	80%	80%
FPR	20%	0%	0%	0%
TNR	80%	100%	100%	100%
FNR	60%	20%	20%	20%

Table 3.92 presents the confusion matrices for different fault scenarios based on the ensemble-based solution when the fault severity is equal to 1%. Table 3.93 shows different measures for

evaluating the performance of the fault detection task based on the residuals obtained from the ensemble-based solution when the fault severity is equal to 1%.

Table 3.92: Confusion matrixes for different fault scenarios based on ensemble-based (i.e. heterogenous ensemble with FSS pruning) solution. Fault severity = 1%.

Decrease in compressor mass flow rate.

	Alarm	No Alarm
Faulty	4	1
Healthy	1	4

Decrease in compressor efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine mass flow rate.

	Alarm	No Alarm
Faulty	4	1
Healthy	0	5

Decrease in turbine efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Table 3.93: Fault detection accuracy of the ensemble-based solution. Fault severity = 1%.

	F_{mc}	F_{ec}	F_{mt}	F_{et}
CCR	80%	100%	90%	100%
Precision	80%	100%	83%	100%
TPR	80%	100%	80%	100%
FPR	20%	0%	0%	0%
TNR	80%	100%	100%	100%
FNR	20%	0%	20%	0%

Table 3.94 presents the confusion matrices for different fault scenarios based on the single model-based solution when the fault severity is equal to 2%. Table 3.95 shows different measures for evaluating the performance of the fault detection task based on the residuals obtained from the single model-based solution when the fault severity is equal to 2%.

Table 3.94: Confusion matrixes for different fault scenarios based on single model-based solution (RBF-NARX model). Fault severity = 2%.

Decrease in compressor mass flow rate.

	Alarm	No Alarm
Faulty	4	1
Healthy	1	4

Decrease in compressor efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine mass flow rate.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Table 3.95: Fault detection accuracy of the single model-based (i.e. RBF-NARX) solution. Fault severity = 2%.

	F_{mc}	F_{ec}	F_{mt}	F_{et}
CCR	80%	100%	100%	100%
Precision	80%	100%	100%	100%
TPR	80%	100%	100%	100%
FPR	20%	0%	0%	0%
TNR	80%	100%	100%	100%
FNR	20%	0%	0%	0%

Table 3.96 presents the confusion matrices for different fault scenarios based on the ensemble-based solution when the fault severity is equal to 2%. Table 3.97 shows different measures for evaluating the performance of the fault detection task based on the residuals obtained from the ensemble-based solution when the fault severity is equal to 2%.

Table 3.96: Confusion matrixes for different fault scenarios based on ensemble-based (i.e. heterogenous ensemble with FSS pruning) solution. Fault severity = 2%.

Decrease in compressor mass flow rate.

	Alarm	No Alarm
Faulty	4	1
Healthy	0	5

Decrease in compressor efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine mass flow rate.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Table 3.97: Fault detection accuracy of the ensemble-based solution. Fault severity = 2%.

	F_{mc}	F_{ec}	F_{mt}	F_{et}
CCR	90%	100%	100%	100%
Precision	83%	100%	100%	100%
TPR	80%	100%	100%	100%
FPR	0%	0%	0%	0%
TNR	100%	100%	100%	100%
FNR	20%	0%	0%	0%

Table 3.98 presents the confusion matrices for different fault scenarios based on the single model-based solution when the fault severity is equal to 4%. Table 3.99 shows different measures for evaluating the performance of the fault detection task based on the residuals obtained from the single model-based solution when the fault severity is equal to 4%.

Table 3.98: Confusion matrixes for different fault scenarios based on single model-based solution (RBF-NARX model). Fault severity = 4%.

Decrease in compressor mass flow rate.

	Alarm	No Alarm
Faulty	5	0
Healthy	1	4

Decrease in compressor efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine mass flow rate.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Table 3.99: Fault detection accuracy of the single model-based (i.e. RBF-NARX) solution. Fault severity = 4%.

	F_{mc}	F_{ec}	F_{mt}	F_{et}
CCR	90%	100%	100%	100%
Precision	100%	100%	100%	100%
TPR	100%	100%	100%	100%
FPR	20%	0%	0%	0%
TNR	80%	100%	100%	100%
FNR	0%	0%	0%	0%

Table 3.100 presents the confusion matrices for different fault scenarios based on the ensemble-based solution when the fault severity is equal to 4%. Table 3.101 shows different measures for evaluating the performance of the fault detection task based on the residuals obtained from the ensemble-based solution when the fault severity is equal to 4%.

Table 3.100: Confusion matrixes for different fault scenarios based on ensemble-based (i.e. heterogenous ensemble with FSS pruning) solution. Fault severity = 4%.

Decrease in compressor mass flow rate.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in compressor efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine mass flow rate.

	Alarm	No Alarm
Faulty	4	1
Healthy	0	5

Decrease in turbine efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Table 3.101: Fault detection accuracy of the ensemble-based solution. Fault severity = 4%.

	F_{mc}	F_{ec}	F_{mt}	F_{et}
CCR	100%	100%	90%	100%
Precision	100%	100%	83%	100%
TPR	100%	100%	80%	100%
FPR	0%	0%	0%	0%
TNR	100%	100%	100%	100%
FNR	0%	0%	20%	0%

Table 3.102 presents the confusion matrices for different fault scenarios based on the single model-based solution when the fault severity is equal to 6%. Table 3.103 shows different measures for evaluating the performance of the fault detection task based on the residuals obtained from the single model-based solution when the fault severity is equal to 6%.

Table 3.102: Confusion matrixes for different fault scenarios based on single model-based solution (RBF-NARX model). Fault severity = 6%.

Decrease in compressor mass flow rate.

	Alarm	No Alarm
Faulty	5	0
Healthy	1	4

Decrease in compressor efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine mass flow rate.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Table 3.103: Fault detection accuracy of the single model-based (i.e. RBF-NARX) solution. Fault severity = 6%.

	F_{mc}	F_{ec}	F_{mt}	F_{et}
CCR	90%	100%	100%	100%
Precision	100%	100%	100%	100%
TPR	100%	100%	100%	100%
FPR	20%	0%	0%	0%
TNR	80%	100%	100%	100%
FNR	0%	0%	0%	0%

Table 3.104 presents the confusion matrices for different fault scenarios based on the ensemble-based solution when the fault severity is equal to 6%. Table 3.105 shows different measures for evaluating the performance of the fault detection task based on the residuals obtained from the ensemble-based solution when the fault severity is equal to 6%.

Table 3.104: Confusion matrixes for different fault scenarios based on ensemble-based (i.e. heterogenous ensemble with FSS pruning) solution. Fault severity = 6%.

Decrease in compressor mass flow rate.

	Alarm	No Alarm
Faulty	4	0
Healthy	1	5

Decrease in compressor efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine mass flow rate.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Table 3.105: Fault detection accuracy of the ensemble-based solution. Fault severity = 6%.

	F_{mc}	F_{ec}	F_{mt}	F_{et}
CCR	90%	100%	100%	100%
Precision	100%	100%	100%	100%
TPR	100%	100%	100%	100%
FPR	17%	0%	0%	0%
TNR	83%	100%	100%	100%
FNR	0%	0%	0%	0%

Table 3.106 presents the confusion matrices for different fault scenarios based on the single model-based solution when the fault severity is equal to 8%. Table 3.107 shows different measures for evaluating the performance of the fault detection task based on the residuals obtained from the single model-based solution when the fault severity is equal to 8%.

Table 3.106: Confusion matrixes for different fault scenarios based on single model-based solution (RBF-NARX model). Fault severity = 8%.

Decrease in compressor mass flow rate.

	Alarm	No Alarm
Faulty	5	0
Healthy	1	4

Decrease in compressor efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine mass flow rate.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Table 3.107: Fault detection accuracy of the single model-based (i.e. RBF-NARX) solution. Fault severity = 8%.

	F_{mc}	F_{ec}	F_{mt}	F_{et}
CCR	90%	100%	100%	100%
Precision	100%	100%	100%	100%
TPR	100%	100%	100%	100%
FPR	20%	0%	0%	0%
TNR	80%	100%	100%	100%
FNR	0%	0%	0%	0%

Table 3.108 presents the confusion matrices for different fault scenarios based on the ensemble-based solution when the fault severity is equal to 8%. Table 3.109 shows different measures for evaluating the performance of the fault detection task based on the residuals obtained from the ensemble-based solution when the fault severity is equal to 8%.

Table 3.108: Confusion matrixes for different fault scenarios based on ensemble-based (i.e. heterogenous ensemble with FSS pruning) solution. Fault severity = 8%.

Decrease in compressor mass flow rate.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in compressor efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine mass flow rate.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Decrease in turbine efficiency.

	Alarm	No Alarm
Faulty	5	0
Healthy	0	5

Table 3.109: Fault detection accuracy of the ensemble-based solution. Fault severity = 8%.

	F_{mc}	F_{ec}	F_{mt}	F_{et}
CCR	100%	100%	100%	100%
Precision	100%	100%	100%	100%
TPR	100%	100%	100%	100%
FPR	0%	0%	0%	0%
TNR	100%	100%	100%	100%
FNR	0%	0%	0%	0%

Based on the above observations, measurability of turbine temperature may improve fault detection performance. We observed that using turbine temperature residual make it possible to detect some fault scenarios which were not detectable otherwise (decrease in turbine mass flow rate with different severities). More specifically, using turbine temperature measurement improves accuracy of detecting the fault in compressor mass flow rate up to 10% for faults lower severities (1% and 2% in fault magnitude). Further investigation would be required based on specifications of new developed sensor as part of HEATTOP project [234], [235] as the accuracy of the sensor plays an important role in its effectiveness for fault detection.

3.9 Summary

In this chapter, first, jet engine dynamic was modeled using both single model-based and ensemble-based solutions. This includes identification of the jet engine dynamics using RBF-NARX, MLP-NARX, and SVM-NARX models (single model-based solutions), as well as two heterogeneous and one homogeneous ensemble systems (ensemble-based solutions). It is observed that system modeling accuracy can be improved up to 67% by using the ensemble learning over the stand-alone learning models. In the rest of the chapter an ensemble-based as well as a single model-based fault detection mechanism were developed. It was shown that the ensemble-based fault detection is generally more accurate. More specifically, it improves the fault detection accuracy by 5% on average over the single model-based solution.

Chapter 4

Ensemble-based Jet Engine Fault Isolation

In Chapter 3 we discussed modeling of jet engine outputs using both single-model solutions, and ensemble models. We showed that a more accurate model can be achieved by using the ensemble systems as compared to the stand-alone models. We also discussed the effects of more accurate modeling on accuracy of the fault detection. In this chapter we use the results of the previous chapter to perform the fault isolation task. To do so we evaluate the residuals that are generated in the previous chapter to isolate engine faults. As we previously discussed in Chapter 3 heterogeneous ensemble model with Forward Sequential Selection (FSS) pruning demonstrates the maximal improvement as compared with the single-model solutions. Also, RBF-NARX model has the best performance among the single-model solutions that are studied in the previous chapter. Thus, in this chapter we perform the fault isolation task using by the residual signals that are generated by both models. We then do comparison between the ensemble-based and single-model based fault isolation schemes.

4.1 Single Fault Isolation

In the previous chapter we modeled jet engine dynamics with the purpose of generating residual signals. We also performed a residual evaluation with the goal of performing fault detection. In this section, we propose a methodology for evaluating residuals with the objective of fault isolation of the jet engine. The fault classes covering the severity ranges discussed in the previous chapter are listed in the following.

Remark 4.1. *Note that in this research only two ranges of fault severities are considered for the sake of* <u>*illustration*</u>*, that is (a) less severe faults (* 3%*), and (b) more severe faults (* 3%*). It should be emphasized that more severity ranges can be easily added by considering more fault labels.*

- Class 1 (f_1): This class contains loss of compressor efficiency (i.e. F_{ec}), by severity between 1% to 3%. As previously described it is common to model component faults by considering some percentage reduction in either efficiency or flow capacity of the engine component. Thus, all the faults in this class can be classified as reduction in the compressor efficiency between 1% to 3%.

- Class 2 (f_2): This class contains loss of compressor efficiency (i.e. F_{ec}), by severity between 4% to 6%. As previously described it is common to model component faults by considering some percentage reduction in either efficiency or flow capacity of the engine component. Thus, all the faults in this class can be classified as reduction in the compressor efficiency between 4% to 6%.

- Class 3 (f_3): This class contains loss of compressor mass flow rate (i.e. F_{mc}), by severity between 1% to 3%. As previously described it is common to model component faults by considering some percentage reduction in either efficiency or flow capacity of the engine component. Thus, all the faults in this class can be classified as reduction in the compressor mass flow rate by 1% to 3%.

- Class 4 (f_4): This class contains loss of compressor mass flow rate (i.e. F_{mc}), by severity between 4% to 6%. As previously described it is common to model component faults by considering some percentage reduction in either efficiency or flow capacity of the engine component. Thus, all the faults in this class can be classified as reduction in the compressor mass flow rate by 4% to 6%.

- Class 5 (f_5): This class contains loss of turbine efficiency (i.e. F_{et}), by severity between 1% to 3%. As previously described it is common to model component faults by considering some percentage reduction in either efficiency or flow capacity of the engine component. Thus, all the faults in this class can be classified as reduction in the turbine efficiency between 1% to 3%.

- Class 6 (f_6): This class contains loss of turbine efficiency (i.e. F_{et}), by severity between 4% to 6%. As previously described it is common to model component faults by considering some percentage reduction in either efficiency or flow capacity of the engine component. Thus, all the faults in this class can be classified as reduction in the turbine efficiency between 4% to 6%.

- Class 7 (f_7): This class contains loss of turbine mass flow rate (i.e. F_{mt}), by severity between 1% to 3%. As previously described it is common to model component faults by considering some percentage reduction in either efficiency or flow capacity of the engine component. Thus, all the faults in this class can be classified as reduction in the turbine mass flow rate by 1% to 3%.

- Class 8 (f_8): This class contains loss of turbine mass flow rate (i.e. F_{mt}), by severity between 4% to 6%. As previously described it is common to model component faults by considering some percentage reduction in either efficiency or flow capacity of the engine component. Thus, all the faults in this class can be classified as reduction in the turbine mass flow rate by 4% to 6%.

4.1.1 Static Neural Networks for Fault Isolation

The use of static neural networks for the purpose of fault isolation of a jet engine is reported in several publications including but not limited to [143], [144], [146], [154], [155], [161], [164], [167], [172], [177]. The same methodology is used in this thesis to isolate the engine faults. Towards this end, a static neural network (multi-layer perceptron) is trained to isolate the engine faults. The inputs of the neural networks should be scalers (rather than time-series, i.e. residual signal); thus, the residuals should be preprocessed in order to be suitable for inputs of the static neural network. In the previous chapter we saw how the variations in the residual signals can be an indication of fault occurrence. Here in this section we use the same concept for determination of fault type, and estimating the fault severity. The designed neural network fault classifier receives the variations of residual signals before and after the fault occurrence, and returns the fault label corresponding to the isolated fault (the list of fault classes is presented before). This is the function of residual evaluation block in Figure 4.1. It receives the residual signals at any moment, and compares them with the fault thresholds (refer to Section 3.8.2), once at least one the residuals exceeds its threshold the residual evaluation block detects a fault. Next the residual evaluation block calculates the variation of the residual signals before and after the fault detection. This scaler vector is then used as an input to the neural network fault classifier. Figure 4.1 summarizes the above mentioned mechanism.

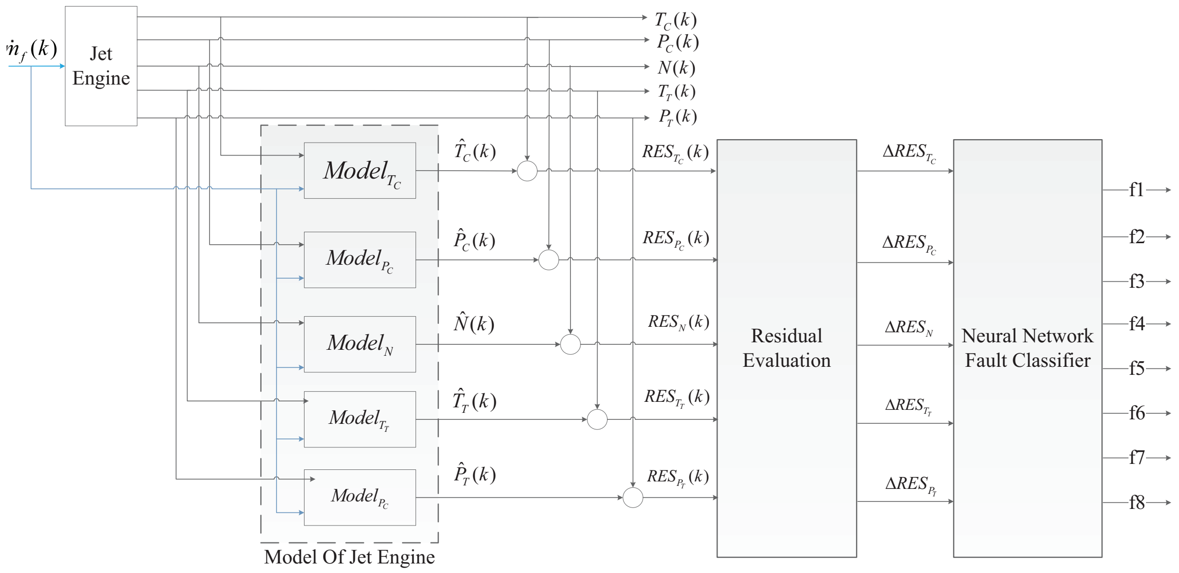


Figure 4.1: Fault isolation mechanism.

The fault isolation task is then performed twice, one time using the residual signals obtained from single model-based solution, and the second time using the residual signals obtained from the ensemble-based solution. In the next section we discuss the residual evaluation mechanism, and the neural network classifier in more details. Then we present the obtained fault isolation results.

4.1.2 Residual Evaluation

The residuals obtained by comparing the actual engine outputs and the engine model outputs are evaluated for the purpose of fault detection and isolation. First, the residuals are compared with the determined thresholds in the previous chapter to detect the fault. Now we evaluate the residuals for the purpose of fault isolation. The residual evaluation block in Figure 4.1 sends the signals into neural network classifier by determining the variations of residuals before and after the fault detection. In other words, the output of the residual evaluation block can be defined as:

$$\Delta RES = \quad \Delta RES_{T_C}, \Delta RES_{P_C}, \Delta RES_N, \Delta RES_{T_T}, \Delta RES_{P_T}$$

We should emphasize that the fault isolation neural network is only called when a fault is detected and the ΔRES vector is defined. Note that the value of ΔRES is only defined for $t \quad t_D$, and undefined otherwise; where t_D is the detection time.

4.1.3 Neural Network Fault Classifier

This section explains the structure of the MLP neural network fault classifier that is used in this research for the fault isolation task. The use of MLP neural networks for fault isolation of jet engines has been reported in many researches including but not limited to [143], [144], [146], [154], [155], [161], [164], [167], [172], [177]. The classifier employs the previously described ΔRES vector as its input, and it returns a fault label vector as listed below:

- F_{ec} with severity less than 3% (f_1) is labeled using [1,0,0,0,0,0,0,0]
- F_{ec} with severity more than 3% (f_2) is labeled using [0,1,0,0,0,0,0,0]
- F_{mc} with severity less than 3% (f_3) is labeled using [0,0,1,0,0,0,0,0]
- F_{mc} with severity more than 3% (f_4) is labeled using [0,0,0,1,0,0,0,0]
- F_{et} with severity less than 3% (f_5) is labeled using [0,0,0,0,1,0,0,0]

- F_{et} with severity more than 3% (f_6) is labeled using [0,0,0,0,0,1,0,0]

- F_{mt} with severity less than 3% (f_7) is labeled using [0,0,0,0,0,0,1,0]

- F_{mt} with severity more than 3% (f_8) is labeled using [0,0,0,0,0,0,0,1]

Training Data

The training data is labeled as mentioned above. The total number of fault scenarios considered here is the same as in Chapter 3. For each of the *four fault types* (F_{mc}, F_{ec}, F_{mt}, F_{et}) we collect the data while engine is operating using *ˇve different input proˇles* $\dot{m}_f = 0.68, 0.7, 0.75, 0.8, 0.85$ of the maximum fuel rate (range of the fuel flow rate while in cruise mode [200]). *Four different fault severities* are considered for each fault type (1%, 2%, 4%, 6% and 8%). Then the residuals are evaluated using the previously mentioned procedure. The total number of input/target pairs generated for training and validation of neural network fault classifier is $4 \quad 5 \quad 5 = 100$. Two different neural network fault classifiers are constructed based on the available data. The first network is constructed using the data obtained from the single model-based solution and the second network is constructed using the data obtained from the ensemble based solution. The summary of the construction of the neural network fault classifier using the data obtained from the single model-based and the ensemble based solutions is shown in Tables 4.1 and 4.1 respectively.

Table 4.1: Summary of the Construction of the Neural Network Fault Classifier based on residuals obtained from single model-based solution.

# neurons	**training data (%)**	**# validation data (%)**	**test data (%)**	CCR_{train}	CCR_{test}	CCR_{valid}	CCR_{total}
8	60	10	30	83	83	70	82
8	**50**	**20**	**30**	**88**	**90**	**80**	**87**
8	40	30	30	85	73	67	76
8	50	10	40	88	85	90	87
8	40	20	40	85	70	70	76
8	30	30	40	87	70	77	77
8	40	10	50	85	66	90	76
8	30	20	50	87	68	85	77
8	20	30	50	90	60	67	68
9	60	10	30	87	83	100	87
9	50	20	30	88	73	85	83
9	40	30	30	88	77	90	85
9	50	10	40	88	80	70	83
9	40	20	40	88	83	85	85
9	30	30	40	87	80	77	81
9	40	10	50	88	78	90	83
9	30	20	50	87	78	80	81
9	20	30	50	85	76	63	74
10	60	10	30	85	73	80	81
10	50	20	30	84	67	65	75

Table 4.2: Summary of the Construction of the Neural Network Fault Classifier based on residuals obtained from ensemble-based solution.

# neurons	training data (%)	# validation data (%)	test data (%)	CCR_{train} (%)	CCR_{test} (%)	CCR_{valid} (%)	CCR_{total} (%)
8	**60**	**10**	**30**	**95**	**92**	**71**	**96**
8	50	20	30	78	62	62	75
8	40	30	30	83	66	69	79
8	50	10	40	92	83	91	93
8	40	20	40	88	73	77	85
8	30	30	40	89	73	82	86
8	40	10	50	88	72	91	85
8	30	20	50	89	72	87	85
8	20	30	50	97	68	79	82
9	60	10	30	91	86	100	95
9	50	20	30	92	86	92	95
9	40	30	30	91	76	92	92
9	50	10	40	92	91	81	95
9	40	20	40	91	83	87	92
9	30	30	40	89	83	82	90
9	40	10	50	91	84	91	92
9	30	20	50	89	82	87	90
9	20	30	50	87	80	69	83
10	60	10	30	90	79	91	91
10	50	20	30	88	76	92	90

The trained neural network receives the ΔRES vector as input and returns the fault label as output. Figure 4.2 shows input-output relation of the designed neural network fault classifier.

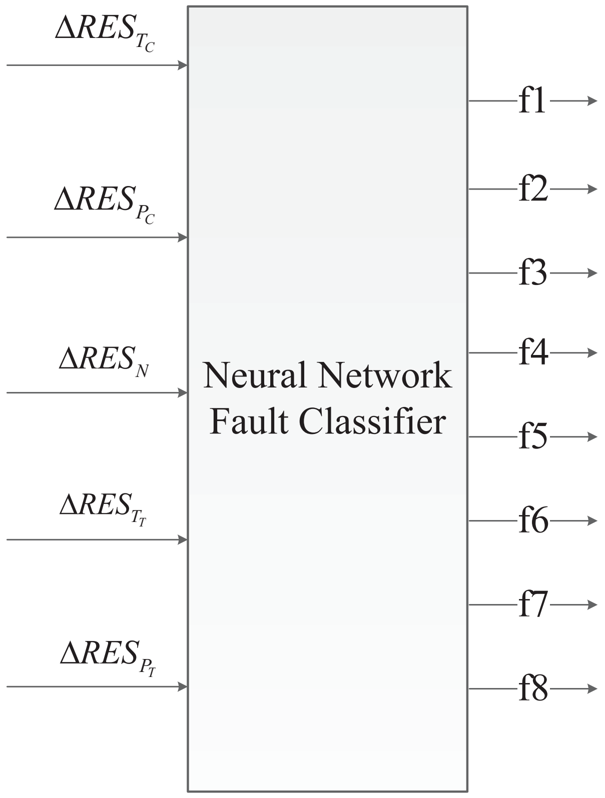


Figure 4.2: Proposed neural network fault classifier.

In the next sections the simulation results are shown for the fault isolation task by using residuals obtained from single-model and ensemble model solutions.

4.1.4 Single Model-based Fault Isolation

In this section we use the residuals that are obtained from the single-model solution (RBF-NARX) for fault isolation task. The fault scenarios are previously explained and data samples for these scenarios are collected (total number of 100 scenarios are developed for the sake of illustration). Among the collected data we randomly select 50 samples (after experimenting with different sizes of training data, i.e. 30, 40, 50, 60 samples). Starting from a small structure we construct a neural network classifier for fault isolation. We observe that an acceptable performance can be archived using a single-layer network with 15 hidden neurons. Tables 4.3 to 4.5 show the confusion matrixes for training and testing of all available data.

Table 4.3: Confusion matrix for training data using single model-based (RBF-NARX) fault isolation.

Actual / Prediction	F_{ec} more than 3%	F_{ec} less than 3%	F_{mc} more than 3%	F_{mc} less than 3%	F_{mt} more than 3%	F_{mt} less than 3%	F_{et} more than 3%	F_{et} less than 3%
F_{ec} more than 3%	3	0	0	0	1	0	0	0
F_{ec} less than 3%	0	5	0	0	0	0	0	0
F_{mc} more than 3%	0	0	7	0	0	0	0	0
F_{mc} less than 3%	0	0	0	7	0	0	0	0
F_{mt} more than 3%	2	2	0	0	8	1	0	0
F_{mt} less than 3%	0	0	0	0	0	6	0	0
F_{et} more than 3%	0	0	0	0	0	0	3	0
F_{et} less than 3%	0	0	0	0	0	0	0	5

Table 4.4: Confusion matrix for testing data using single model-based (RBF-NARX) fault isolation.

Actual / Prediction	F_{ec} more than 3%	F_{ec} less than 3%	F_{mc} more than 3%	F_{mc} less than 3%	F_{mt} more than 3%	F_{mt} less than 3%	F_{et} more than 3%	F_{et} less than 3%
F_{ec} more than 3%	7	0	0	0	1	0	1	0
F_{ec} less than 3%	0	3	0	0	0	0	0	0
F_{mc} more than 3%	0	0	6	0	0	0	0	0
F_{mc} less than 3%	0	0	0	5	0	0	0	0
F_{mt} more than 3%	1	0	0	0	5	1	1	2
F_{mt} less than 3%	0	0	0	0	0	4	0	0
F_{et} more than 3%	0	0	0	0	0	0	5	0
F_{et} less than 3%	0	0	0	0	0	0	3	5

Table 4.5: Confusion matrix for both training and testing data using single model-based (RBF-NARX) fault isolation.

Prediction \ Actual	F_{ec} more than 3%	F_{ec} less than 3%	F_{mc} more than 3%	F_{mc} less than 3%	F_{mt} more than 3%	F_{mt} less than 3%	F_{et} more than 3%	F_{et} less than 3%
F_{ec} more than 3%	10	0	0	0	2	0	1	0
F_{ec} less than 3%	0	8	0	0	0	0	0	0
F_{mc} more than 3%	0	0	13	0	0	0	0	0
F_{mc} less than 3%	0	0	0	12	0	0	0	0
F_{mt} more than 3%	3	2	0	0	13	2	1	2
F_{mt} less than 3%	0	0	0	0	0	10	0	0
F_{et} more than 3%	0	0	0	0	0	0	8	0
F_{et} less than 3%	0	0	0	0	0	0	3	10

We use *correct classiˇcation rate (CCR)* as a measure for accuracy of fault isolation task as defined below:

$$CCR = \frac{\text{total number of correctly classified samples}}{\text{total number of samples}}$$

Table 4.6 shows the fault isolation performance for single model-based fault isolation.

Table 4.6: Fault isolation performance of single model-based (RBF-NARX) solution in term of CCR.

$CCR_{training}$	88%
$CCR_{testing}$	80%
CCR_{total}	84%

4.1.5 Ensemble-based Fault Isolation

In this section we use the residuals that are obtained from the ensemble-based solution (RBF-NARX) for fault isolation task. The fault scenarios are previously explained and data samples for these scenarios are collected (total number of 100 scenarios are developed for the sake of illustration). Among the collected data we randomly select 50 samples (after experimenting with different sizes of training data i.e., 30, 40, 50, 60 samples). To have faire comparative study we use the same number of training samples, and network architecture as in the single model-based solution. Tables 4.7 to 4.9 show the confusion matrices for the training, testing for all available data.

Table 4.7: Confusion matrix for training data using ensemble-based fault isolation.

Actual / Prediction	F_{ec} more than 3%	F_{ec} less than 3%	F_{mc} more than 3%	F_{mc} less than 3%	F_{mt} more than 3%	F_{mt} less than 3%	F_{et} more than 3%	F_{et} less than 3%
F_{ec} more than 3%	5	0	0	0	0	0	0	0
F_{ec} less than 3%	0	2	0	0	0	0	0	0
F_{mc} more than 3%	0	0	9	0	0	0	0	0
F_{mc} less than 3%	0	0	0	3	0	0	0	0
F_{mt} more than 3%	0	1	0	0	9	0	0	0
F_{mt} less than 3%	0	0	0	0	0	7	0	0
F_{et} more than 3%	0	0	0	0	0	0	10	0
F_{et} less than 3%	0	0	0	0	0	0	0	4

Table 4.8: Confusion matrix for testing data using ensemble-based fault isolation.

Actual / Prediction	F_{ec} more than 3%	F_{ec} less than 3%	F_{mc} more than 3%	F_{mc} less than 3%	F_{mt} more than 3%	F_{mt} less than 3%	F_{et} more than 3%	F_{et} less than 3%
F_{ec} more than 3%	9	0	0	0	0	0	0	0
F_{ec} less than 3%	0	6	0	0	0	0	0	0
F_{mc} more than 3%	0	0	4	0	0	0	0	0
F_{mc} less than 3%	0	0	0	5	0	0	0	0
F_{mt} more than 3%	0	0	0	0	5	0	0	0
F_{mt} less than 3%	0	0	0	0	0	4	0	0
F_{et} more than 3%	0	0	0	0	0	0	4	0
F_{et} less than 3%	0	0	0	0	0	0	0	3

Table 4.9: Confusion matrix for both training and testing data using ensemble-based fault isolation.

Actual / Prediction	F_{ec} more than 3%	F_{ec} less than 3%	F_{mc} more than 3%	F_{mc} less than 3%	F_{mt} more than 3%	F_{mt} less than 3%	F_{et} more than 3%	F_{et} less than 3%
F_{ec} more than 3%	14	0	0	0	0	0	0	0
F_{ec} less than 3%	0	8	0	0	0	0	0	0
F_{mc} more than 3%	0	0	13	0	0	0	0	0
F_{mc} less than 3%	0	0	0	8	0	0	0	0
F_{mt} more than 3%	0	1	0	0	14	0	0	0
F_{mt} less than 3%	0	0	0	0	0	11	0	0
F_{et} more than 3%	0	0	0	0	0	0	14	0
F_{et} less than 3%	0	0	0	0	0	0	0	7

Table 4.10 shows the fault isolation performance for ensemble-based fault isolation in term of CCR.

Table 4.10: Fault isolation performance of ensemble-based solution in term of CCR.

CCR_{train}	98%
CCR_{test}	100%
CCR_{total}	99%

4.2 Multiple Faults Isolation

In the previous section we studied the fault isolation problem, assuming that only a single fault may be present at each time. In this section, however, the goal is to isolate simultaneous faults. Isolating multiple faults is a complex problem. Thus, in order to limit the complexity we assume that only two concurrent faults may happen. We assume that the first fault happens at t_1 = 20 sec and the second fault happens at t_2 = 30 sec. The fault scenarios studied in this section are listed in the following.

- Class 1 (F_{ec}): This class contains loss of compressor efficiency (i.e. F_{ec}), by severity between 1% to 6%. As previously described it is common to model component faults by considering some percentage reduction in either efficiency or flow capacity of the engine component. Thus, all the faults in this class can be classified as reduction in the compressor efficiency between 1% to 6%.

- Class 2 (F_{mc}): This class contains loss of compressor mass flow rate (i.e. F_{mc}), by severity between 1% to 6%. As previously described it is common to model component faults by considering some percentage reduction in either efficiency or flow capacity of the engine component. Thus, all the faults in this class can be classified as reduction in the compressor mass flow rate by 1% to 6%

- Class 3 (F_{et}): This class contains loss of turbine efficiency (i.e. F_{et}), by severity between 1% to 6%. As previously described it is common to model component faults by considering some percentage reduction in either efficiency or flow capacity of the engine component. Thus, all the faults in this class can be classified as reduction in the turbine efficiency between 1% to 6%.

- Class 4 (F_{mt}): This class contains loss of turbine mass flow rate (i.e. F_{mt}), by severity between 1% to 6%. As previously described it is common to model component faults by considering some percentage reduction in either efficiency or flow capacity of the engine component. Thus, all the faults in this class can be classified as reduction in the turbine mass flow rate by 1% to 6%.

- Class 5 (F_{ec}, F_{mc}): This fault class includes simultaneous faults in both compressor efficiency (i.e. F_{ec}) and compressor mass flow rate (i.e. F_{mc}). The severity of each fault is between 1% to 6%. In other words, there is a drop of 1% to 6% in compressor efficiency, as well as, a drop of 1% to 6% in compressor mass flow rate.

- Class 6 (F_{ec}, F_{et}): This fault class includes simultaneous faults in both compressor efficiency (i.e. F_{ec}) and turbine efficiency (i.e. F_{et}). The severity of each fault is between 1% to 6%. In other words, there is a drop of 1% to 6% in compressor efficiency, as well as, a drop of 1% to 6% in turbine efficiency.

- Class 7 (F_{ec}, F_{mt}): This fault class includes simultaneous faults in both compressor efficiency (i.e. F_{ec}) and turbine mass flow rate (i.e. F_{mt}). The severity of each fault is between 1% to 6%. In other words, there is a drop of 1% to 6% in compressor efficiency, as well as, a drop of 1% to 6% in turbine mass flow rate.

- Class 8 (F_{mc}, F_{et}): This fault class includes simultaneous faults in both compressor mass flow rate (i.e. F_{mc}) and turbine efficiency (i.e. F_{et}). The severity of each fault is between 1% to 6%. In other words, there is a drop of 1% to 6% in compressor mass flow rate, as well as, a drop of 1% to 6% in turbine efficiency.

- Class 9 (F_{mc}, F_{mt}): This fault class includes simultaneous faults in both compressor mass flow rate (i.e. F_{mc}) and turbine mass flow rate (i.e. F_{mt}). The severity of each fault is between 1% to 6%. In other words, there is a drop of 1% to 6% in compressor mass flow rate, as well as, a drop of 1% to 6% in turbine mass flow rate.

- Class 10 (F_{et}, F_{mt}): This fault class includes simultaneous faults in both turbine efficiency (i.e. F_{et}) and turbine mass flow rate (i.e. F_{mt}). The severity of each fault is between 1% to 6%. In other words, there is a drop of 1% to 6% in turbine efficiency, as well as, a drop of 1% to 6% in turbine mass flow rate.

4.2.1 Neural Network Fault Classifier

The structure of the MLP neural network fault classifier used for multiple fault isolation is shown in Figure 4.3. The neural network receives the set of variations in the value of residuals before and after the fault detection. The output vector of the classifier is labeled to identify the class of fault(s) as listed below:

- Fault class F_{ec} is labeled using [1,0,0,0,0,0,0,0,0,0]
- Fault class F_{mc} is labeled using [0,1,0,0,0,0,0,0,0,0]
- Fault class F_{mt} is labeled using [0,0,1,0,0,0,0,0,0,0]
- Fault class F_{et} is labeled using [0,0,0,1,0,0,0,0,0,0]
- Fault class F_{mc}, F_{ec} is labeled using [0,0,0,0,1,0,0,0,0,0]
- Fault class F_{mc}, F_{mt} is labeled using [0,0,0,0,0,1,0,0,0,0]

- Fault class F_{mc}, F_{et} is labeled using [0,0,0,0,0,0,1,0,0,0]
- Fault class F_{ec}, F_{et} is labeled using [0,0,0,0,0,0,0,1,0,0]
- Fault class F_{ec}, F_{mt} is labeled using [0,0,0,0,0,0,0,0,1,0]
- Fault class F_{mt}, F_{et} is labeled using [0,0,0,0,0,0,0,0,0,1]

Training Data

The training data is labeled as mentioned above. For each of the above mentioned *ten fault classes* we collect the data while engine is operating using *ˇve different input proˇles* $\dot{m}_f = 0.68, 0.7, 0.75, 0.8, 0.85$ of the maximum fuel rate (range of the fuel flow rate while in cruise mode [200]). *Four different fault severities* are considered for each fault type (1%, 2%, 4%, and 6%). We assume that the first fault happens at $t_1 = 20$ sec and the second fault happens at $t_2 = 30$ sec. Then the residuals are evaluated using the previously mentioned procedure. The total number of input/target pairs generated for training and validation of neural network fault classifier is 200. Figure 4.2 shows input-output relation of the designed neural network fault classifier.

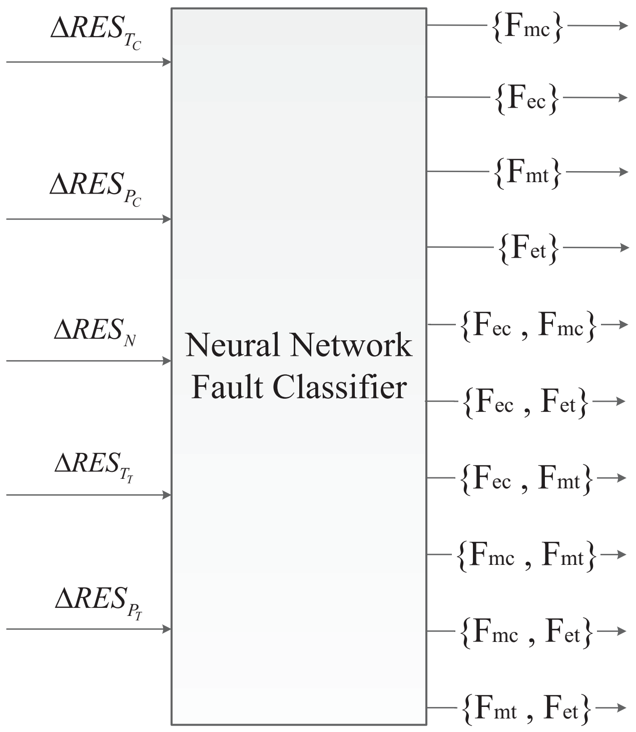


Figure 4.3: Proposed neural network for multiple fault isolation.

Two different neural network fault classifiers are constructed based on the available data. The first network is constructed using the data obtained from the single model-based solution and the second network is constructed using the data obtained from the ensemble based solution. The summary of the construction of the neural network fault classifier using the data obtained from the single model-based and the ensemble based solutions is shown in Tables 4.11 and 4.12 respectively.

Table 4.11: Summary of the construction of the neural network concurrent fault classifier based on residuals obtained from single model-based solution.

# neurons	**training data (%)**	**# validation data (%)**	**test data (%)**	CCR_{train}**(%)**	CCR_{test} **(%)**	CCR_{valid} **(%)**	CCR_{total} **(%)**
8	60	10	30	88.6667	79.5	78.5	82
8	50	20	30	96.5	90.6667	91	90.5
8	40	30	30	84.25	73.5	75.1667	75
8	50	10	40	97.5	93.25	98	92.5
8	40	20	40	84.25	75.25	78	76
8	30	30	40	98.5	81	83	83.5
8	40	10	50	84.25	74.5	72.5	74.5
8	30	20	50	98.5	82	82.5	83.5
8	20	30	50	99	79.5	84.1667	81.5
9	60	10	30	97	96.1667	98.5	94
9	50	20	30	97.5	97.3333	98.5	94.5
9	40	30	30	98	93.5	96.8333	93
9	**50**	**10**	**40**	**97.5**	**98.25**	**98**	**94.5**
9	40	20	40	98	94	98	93
9	30	30	40	98.5	96	96.3333	93.5
9	40	10	50	98	96.5	97.5	93.5
9	30	20	50	98.5	97	95	93.5
9	20	30	50	99	90.5	89.1667	88.5
10	60	10	30	97	96.1667	98.5	94
10	50	20	30	97.5	95.6667	98.5	94

Table 4.12: Summary of the construction of the neural network concurrent fault classifier based on residuals obtained from ensemble-based solution.

# neurons	training data (%)	# validation data (%)	test data (%)	CCR_{train}(%)	CCR_{test} (%)	CCR_{valid} (%)	CCR_{total} (%)
8	60	10	30	98.8	94.8	94.4	96
8	50	20	30	89	91.2667	89.4	88.5
8	40	30	30	99.2	92.7333	97.7333	95.5
8	50	10	40	89	89.8	99.2	89
8	40	20	40	99.2	93.35	99.2	95.5
8	30	30	40	91.0667	76.9	72.5333	78.5
8	40	10	50	99.2	94.8	99	95.5
8	30	20	50	91.0667	78.6	66.5	78.5
8	20	30	50	84.6	70.4	62.3333	69.5
9	60	10	30	98.8	94.8	99.4	96.5
9	50	20	30	99	97.9333	99.4	97.5
9	40	30	30	99.2	91.0667	94.4	94
9	50	10	40	99	98.55	99.2	97.5
9	40	20	40	99.2	92.1	94.2	94
9	30	30	40	97.7333	90.65	89.2	91
9	40	10	50	94.2	87.8	79	88
9	30	20	50	97.7333	91.6	86.5	91
9	20	30	50	99.6	80.4	79	82.5
10	**60**	**10**	**30**	**98.8**	**99.8**	**99.4**	**98**
10	50	20	30	99	99.6	99.4	98

In the next sections the simulation results are shown for the fault isolation task by using residuals obtained from single-model and ensemble model solutions.

4.2.2 Single Model-based Multiple Faults Isolation

In this section we use the residuals that are obtained from the single-model solution (RBF-NARX) for fault isolation task. The fault scenarios are previously explained and data samples for these scenarios are collected (total number of 100 scenarios are developed for the sake of illustration). Among the collected data we randomly select 50 samples (after experimenting with different sizes of training data, i.e. 30, 40, 50, 60 samples). Starting from a small structure we construct a neural network classifier for fault isolation. We observe that an acceptable performance can be archived using a single-layer network with 15 hidden neurons. Tables 4.13 to 4.15 show the confusion matrixes for training and testing of all available data.

Table 4.13: Confusion matrix for training data using single model-based (RBF-NARX) single fault isolation.

Actual / Prediction	F_{ec}	F_{mc}	F_{mt}	F_{et}	F_{mc} , F_{ec}	F_{mc} , F_{mt}	F_{mc} , F_{et}	F_{ec} , F_{et}	F_{ec} , F_{mt}	F_{mt} , F_{et}
F_{ec}	7	0	0	0	0	0	0	0	0	0
F_{mc}	0	9	0	0	0	1	0	0	0	0
F_{mt}	0	0	12	0	0	0	0	0	0	0
F_{et}	0	0	0	12	0	0	0	0	0	0
F_{mc} , F_{ec}	0	0	0	0	14	0	0	0	0	0
F_{mc} , F_{mt}	0	0	0	0	0	12	0	0	0	0
F_{mc} , F_{et}	0	0	0	0	0	0	12	0	0	0
F_{ec} , F_{et}	1	0	0	0	0	0	0	13	0	0
F_{ec} , F_{mt}	0	0	0	0	0	0	0	0	15	0
F_{mt} , F_{et}	0	0	0	0	0	0	0	0	0	12

Table 4.14: Confusion matrix for testing data using single model-based (RBF-NARX) single fault isolation.

Prediction \ Actual	F_{ec}	F_{mc}	F_{mt}	F_{et}	F_{mc} , F_{ec}	F_{mc} , F_{mt}	F_{mc} , F_{et}	F_{ec} , F_{et}	F_{ec} , F_{mt}	F_{mt} , F_{et}
F_{ec}	4	0	0	0	0	0	0	0	0	0
F_{mc}	0	11	0	0	0	2	0	0	0	0
F_{mt}	0	0	8	0	0	0	0	0	0	0
F_{et}	0	0	0	8	0	0	0	0	0	0
F_{mc} , F_{ec}	0	0	0	0	6	0	0	0	0	0
F_{mc} , F_{mt}	0	0	0	0	0	5	0	0	0	0
F_{mc} , F_{et}	0	0	0	0	0	0	8	0	0	0
F_{ec} , F_{et}	8	0	0	0	0	0	0	7	0	0
F_{ec} , F_{mt}	0	0	0	0	0	0	0	0	5	0
F_{mt} , F_{et}	0	0	0	0	0	0	0	0	0	8

Table 4.15: Confusion matrix for both training and testing data using single model-based (RBF-NARX) single fault isolation.

Prediction \ Actual	F_{ec}	F_{mc}	F_{mt}	F_{et}	F_{mc} , F_{ec}	F_{mc} , F_{mt}	F_{mc} , F_{et}	F_{ec} , F_{et}	F_{ec} , F_{mt}	F_{mt} , F_{et}
F_{ec}	11	0	0	0	0	0	0	0	0	0
F_{mc}	0	20	0	0	0	3	0	0	0	0
F_{mt}	0	0	20	0	0	0	0	0	0	0
F_{et}	0	0	0	20	0	0	0	0	0	0
F_{mc} , F_{ec}	0	0	0	0	20	0	0	0	0	0
F_{mc} , F_{mt}	0	0	0	0	0	17	0	0	0	0
F_{mc} , F_{et}	0	0	0	0	0	0	20	0	0	0
F_{ec} , F_{et}	9	0	0	0	0	0	0	20	0	0
F_{ec} , F_{mt}	0	0	0	0	0	0	0	0	20	0
F_{mt} , F_{et}	0	0	0	0	0	0	0	0	0	20

Table 4.16 shows the fault isolation performance for single model-based fault isolation.

Table 4.16: Multiple Fault isolation performance of single model-based (RBF-NARX) solution in term of CCR.

$CCR_{training}$	98.33%
$CCR_{testing}$	87.5%
CCR_{total}	94%

4.2.3 Ensemble-based Multiple Faults Isolation

In this section we use the residuals that are obtained from the ensemble-based solution (RBF-NARX) for fault isolation task. The fault scenarios are previously explained and data samples for these scenarios are collected (total number of 100 scenarios are developed for the sake of illustration). Among the collected data we randomly select 50 samples (after experimenting with different sizes of training data i.e., 30, 40, 50, 60 samples). To have faire comparative study we use the same number of training samples, and network architecture as in the single model-based solution. Tables 4.17 to 4.19 show the confusion matrices for the training, testing for all available data.

Table 4.17: Confusion matrix for training data using ensemble-based multiple fault isolation.

Prediction \ Actual	F_{ec}	F_{mc}	F_{mt}	F_{et}	F_{mc} , F_{ec}	F_{mc} , F_{mt}	F_{mc} , F_{et}	F_{ec} , F_{et}	F_{ec} , F_{mt}	F_{mt} , F_{et}
F_{ec}	8	0	0	0	0	0	0	0	0	0
F_{mc}	0	9	0	0	0	0	0	0	0	0
F_{mt}	0	0	12	0	0	0	0	0	0	0
F_{et}	0	0	0	12	0	0	0	0	0	0
F_{mc} , F_{ec}	0	0	0	0	14	0	0	0	0	0
F_{mc} , F_{mt}	0	0	0	0	0	12	0	0	0	0
F_{mc} , F_{et}	0	0	0	0	0	0	12	0	0	0
F_{ec} , F_{et}	0	0	0	0	0	0	0	13	0	0
F_{ec} , F_{mt}	0	0	0	0	0	0	0	0	15	0
F_{mt} , F_{et}	0	0	0	0	0	1	0	0	0	12

Table 4.18: Confusion matrix for testing data using ensemble-based multiple fault isolation.

Prediction \ Actual	F_{ec}	F_{mc}	F_{mt}	F_{et}	F_{mc} , F_{ec}	F_{mc} , F_{mt}	F_{mc} , F_{et}	F_{ec} , F_{et}	F_{ec} , F_{mt}	F_{mt} , F_{et}
F_{ec}	9	0	0	0	0	0	0	0	0	0
F_{mc}	0	11	0	0	0	2	0	0	0	0
F_{mt}	0	0	8	0	0	0	0	0	0	0
F_{et}	0	0	0	8	0	0	0	0	0	0
F_{mc} , F_{ec}	0	0	0	0	6	0	0	0	0	0
F_{mc} , F_{mt}	0	0	0	0	0	5	0	0	0	0
F_{mc} , F_{et}	0	0	0	0	0	0	8	0	0	0
F_{ec} , F_{et}	3	0	0	0	0	0	0	7	0	0
F_{ec} , F_{mt}	0	0	0	0	0	0	0	0	5	0
F_{mt} , F_{et}	0	0	0	0	0	0	0	0	0	8

Table 4.19: Confusion matrix for both training and testing data using ensemble-based multiple fault isolation.

Actual / Prediction	F_{ec}	F_{mc}	F_{mt}	F_{et}	F_{mc} , F_{ec}	F_{mc} , F_{mt}	F_{mc} , F_{et}	F_{ec} , F_{et}	F_{ec} , F_{mt}	F_{mt} , F_{et}
F_{ec}	17	0	0	0	0	0	0	0	0	0
F_{mc}	0	20	0	0	0	2	0	0	0	0
F_{mt}	0	0	20	0	0	0	0	0	0	0
F_{et}	0	0	0	20	0	0	0	0	0	0
F_{mc} , F_{ec}	0	0	0	0	20	0	0	0	0	0
F_{mc} , F_{mt}	0	0	0	0	0	17	0	0	0	0
F_{mc} , F_{et}	0	0	0	0	0	0	20	0	0	0
F_{ec} , F_{et}	3	0	0	0	0	0	0	20	0	0
F_{ec} , F_{mt}	0	0	0	0	0	0	0	0	20	0
F_{mt} , F_{et}	0	0	0	0	0	1	0	0	0	20

Table 4.20 shows the fault isolation performance for ensemble-based fault isolation in term of CCR.

Table 4.20: Multiple Fault isolation performance of ensemble-based solution in term of CCR.

CCR_{train}	99.17%
CCR_{test}	93.75%
CCR_{total}	97%

4.3 Summary

In this chapter we have performed the fault isolation task using the residual signals generated by both ensemble model as well as stand-alone model. We then do comparison between the ensemble-based and single-model based fault isolation schemes. The fault isolation mechanism used in this chapter is a static neural network, the network evaluates the changes in engine parameters in order to isolate the occurred fault. It is observed that the ensemble-based fault isolation solution is generally more accurate, and it can improve the single fault isolation accuracy up to 12% and the multiple fault isolation by up to 4% as compared with the single model-based fault isolation scheme. It should be noted that the single fault isolation problem is formulated such that the severity of the fault has to be estimated. Based on the observations estimating the fault severity is generally a more complex task.

Chapter 5

Conclusion and Summary

5.1 Summary

The objective of this thesis was to develop ensemble-based approach for fault-detection and isolation (FDI) of aircraft jet engines and compare the results with conventional single-model-based FDI solutions. It was shown that by combining stand-alone models, more accurate ensemble models can be designed to model the jet engine dynamics without the need of ad-hoc fine tunings required for single-model-based solutions.

For the purpose of jet engine health monitoring, first we modeled the jet engine dynamics using three different stand-alone learning algorithm. Specifically, MLP-NARX, RBF-NARX, and SVM-NARX models are trained to individually model jet engine parameters. A separate model was trained for each of the engine outputs using each individual learning algorithm. Input parameters of the individual learning algorithm (e.g. number of neural network neurons) are optimized by performing several trials. We observed that the RBF-NARX model shows a better performance (in term of modeling accuracy) among the stand-alone models.

In Chapter 3, three different ensemble methods were designed to identify the jet engine dynamics. The first trained ensemble model is a heterogeneous ensemble with ranked pruning. In this approach, first a pool of individual learners were trained using different learning algorithms (MLP-NARX, RBF-NARX, SVM-NARX). Then, the most accurate models are selected for each learning algorithm in order to be aggregated and generate the final prediction. The predictions of ensemble members were combined using weighted averaging where the weights are optimized using gradient descent method. The second ensemble approach is the heterogeneous ensemble with Forward Sequential Selection (FSS) pruning. Similar to the heterogeneous ensemble with ranked pruning, we first trained a pool of stand-alone models for identifying engine outputs. Then, the ensemble initially use the model with best performance (in term of modeling accuracy), and then other models were added to the ensemble based on their contribution to improve the ensemble performance (in each iteration, all candidates in the pool were tested and the one with the maximal improvement to the ensemble performance was selected). The third ensemble model which attempted to model engine dynamics was a homogeneous ensemble with bagging where several RBF-NARX models were trained using different subsets of the training data which are generated by the bootstrap sampling to model the engine dynamics. Also, the effects of the number of models in an ensemble on its accuracy was studied. It was observed that by increasing the number of models in an ensemble the prediction error decreases in general. We also observed that all three ensemble models outperform the stand-alone models in term of modeling accuracy. More specifically, we observed than modeling error was reduced by up to 67% using ensemble methods as compared with single-model-based solutions.

In Chapter 3, we selected heterogenous ensembles with FSS pruning (as it performed the best among the other ensembles), and RBF-NARX as a stand-alone model with better performance to solve the FDI problem. Engine residual signals were generated using both single-model-based and ensemble-based solutions under different engine health conditions. Different fault scenarios were developed based on fault type, fault severity, and engine's input profile (fuel flow rate). The obtained residuals were evaluated in order detect engine faults. Our experiments showed that the fault detection task using residuals obtained from the ensemble model results is more accurate. In Chapter 4, the fault isolation task was performed by evaluating variations in the residual signals (before and after a fault detection) by using a neural network classifier. Eight different fault classes were defined (based on the fault type and fault severity). Since the inputs of the neural network fault classifier should be scaler vector (rather than time-series, i.e. residual signal), we preprocessed the residuals so we could make them suitable for inputs of a static neural network classifier. As in the fault detection case, it is observed that the ensemble-based fault isolation task results in a more promising performance. Specifically, we observed that ensemble-based fault isolation solution improves the fault isolation accuracy by 10% as compared with single-model-based fault isolation scheme.

5.2 Conclusions

In this thesis, first jet engine dynamic was modeled using both single model-based and ensemble-based solutions. It is observed that system modeling accuracy can be improved up to 67% by using the ensemble learning over the stand-alone learning models. Second, an ensemble-based as well as a single model-based fault detection mechanism were developed. It was shown that the ensemble-based fault detection is generally more accurate. Specifically, it improves the fault detection accuracy by 5% on average over the single model-based solution. We also performed the fault isolation task using the residual signals generated by both ensemble model as well as stand-alone model. We then did comparison between the ensemble-based and single-model based fault isolation schemes. It was observed that the ensemble-based fault isolation solution is generally more accurate, and it can improve the single fault isolation accuracy up to 12% and the multiple fault isolation by up to 4% as compared with the single model-based fault isolation scheme.

5.3 Suggestions for Future Work

This research can be extended in a number directions. Some suggestions for future work are explained below:

- In this research we used external dynamics (external delays) to model dynamics of the system. An alternative would be to use learning methods with internal dynamics such as neural networks with dynamic neurons or recurrent neural networks with local feedback and then combine them to build an ensemble system.
- As described briefly in this thesis, increasing the number of models in an ensemble can reduce the prediction error (theoretically any arbitrary level of accuracy can be achieved by increasing the number of ensemble members [187]). A future research could be to increase the number of ensemble members with the goal of achieving more accurate FDI system.
- In this thesis we studied the ensemble models where source of diversity is either the variation in the training data (homogeneous with bagging) or employment of different learning methods (heterogenous ensemble). An alternative would be studying homogenous ensemble

with different architectures (e.g. using several feedforward neural networks with different number of layers and/or neurons).

- In this thesis batch learning (off-line) techniques are applied for identifying the jet engine dynamics. A potential future work is to study the online ensemble learning to identify the jet engine dynamics while the engine is operating.

- In this thesis, the combining weights of the ensemble are assumed to be constant as well as identical for all instances of the input. A potential future work is to study the ensembles where the combining weights differ from a sample to another. For example the *model A* may have more contribution in the output at the instance x_A while the *model B* may have more contribution in the output at the instance x_B.

Bibliography

[1] H. Rong, G. Zhang, C. Zhang, "Application of support vector machines to nonlinear system identification," *Autonomous Decentralized Systems*, pp. 501–507, 2005.

[2] T. Falck, P. Dreesen, K. Brabanter, K. Pelckmans, B. Moor, J. Suykens, "Least-Squares Support Vector Machines for the identification of WienerHammerstein systems", *Control Engineering Practice*, vol. 20, no. 11, pp. 1165–1174, 2012.

[3] M. Martinez-Ramon, J. L. Rojo-Alvarez, G. Camps-Valls, J. Munoz-Mari, A. Navia-Vazquez, E. Soria-Olivas, A.R. Figueiras-Vidal, "Support Vector Machines for Nonlinear Kernel ARMA System Identification," *IEEE Transactions on Neural Networks*, vol. 17, no.6, pp.1617–1622, 2006.

[4] M. Vogt, "System identification techniques based on support vector machines without bias term," *International Journal of Adaptive Control and Signal*, vol. 27, no.9, pp.1099–1115, 2013.

[5] R. Salat, M. Awtoniuk, K. Korpysz, "Black-Box system identification by means of Support Vector Regression and Imperialist Competitive Algorithm", vol. 9, pp. 223–226, *Przeglkad Elektrotechniczny*, 2013.

[6] P. M. L. Drezet, R. F. Harrison, “Support vector machines for system identification,” UKACC International Conference on Control, vol. 1, pp. 688–692, 1998.

[7] J. D. Bomberger, D. E. Seborg, “Determination of model order for NARX models directly from input-output data”, *Journal of Process Control*, vol. 8, no. 56, pp. 459–468, 1998.

[8] S. Tan, J. Hao, J. Vandewalle, “Efficient identification of RBF neural net models for nonlinear discrete-time multivariable dynamical systems”, *Neurocomputing*, vol. 9, no. 1, pp. 11–26, 1995.

[9] S. Lu, T. Basar, “Robust nonlinear system identification using neural-network models,” *IEEE Transactions on Neural Networks*, vol.9, no.3, pp. 407–429, 1998.

[10] V. T. Elanayar, Y. C. Shin, “Radial basis function neural network for approximation and estimation of nonlinear stochastic dynamic systems,” *IEEE Transactions on Neural Networks*, vol. 5, no. 4, pp. 594–603, 1994.

[11] S. Tan, J. Hao, J. Vandewalle, “Nonlinear systems identification using RBF neural networks,”, *Proceedings of International Joint Conference on Neural Networks*, vol. 2, pp. 1833–1836, 1993.

[12] J. Li, F. Zhao, “Identification of dynamical systems using radial basis function neural networks with hybrid learning algorithm,”, *International Symposium on Systems and Control in Aerospace and Astronautics*, pp. 1115–1118, 2006.

[13] S. Chen, X. X. Wang, C. J. Harris, “NARX-Based Nonlinear System Identification Using Orthogonal Least Squares Basis Hunting,” *IEEE Transactions on Control Systems Technology*, vol. 16, no. 1, pp.78–84, 2008.

[14] I. M. Yassin, M. N. Taib, M. Z. Abdul Aziz, “Identification of DC motor drive system model using Radial Basis Function (RBF) Neural Network,” *IEEE Symposium on Industrial Electronics and Applications*, pp. 13–18, 2011.

[15] S. Chen, S. A. Billings , P. M. Grant, “Recursive hybrid algorithm for non-linear system identification using radial basis function networks”, *International Journal of Control*, vol. 55, no. 5, pp. 1051–1070, 1992.

[16] S. Chen, S. A. Billings , C.F. Cowan, P. M. Grant, “Practical identification of NARMAX models using radial basis functions”, *International Journal of Control*, vol. 52, no. 6, pp. 1327–1350, 1990.

[17] J. G. Kuschewski, S. Hui, S.H. Zak, “Application of feedforward neural networks to dynamical system identification and control,” *IEEE Transactions on Control Systems Technology*, vol. 1, no. 1, pp. 37–49, Mar 1993.

[18] V. Prasad, B. W. Bequette, “Nonlinear system identification and model reduction using artificial neural networks”, *Computers & Chemical Engineering*, vol. 27, no. 12, pp. 1741–1754, 2003.

[19] H. Zhao, J. Zhang, “Nonlinear dynamic system identification using pipelined functional link artificial recurrent neural network”, *Neurocomputing*, vol. 72, no. 13, pp. 3046-3054, 2009.

[20] S. Chen, S. A. Billings, P. M. Grant, “Non-linear system identification using neural networks”, *International Journal of Control*, vol. 51, no. 6, pp. 1191-1214, 1990.

[21] C. C. Huang, C. Loh, “Nonlinear Identification of Dynamic Systems Using Neural Networks”, *Computer-Aided Civil and Infrastructure Engineering*, vol. 16, no. 1, pp. 28-41, 2001.

[22] J. C. Patra, R. N. Pal, B. N. Chatterji, G. Panda, "Identification of nonlinear dynamic systems using functional link artificial neural networks," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 29, no. 2, pp.254–262, 1999.

[23] S. Chen, S. A. Billings, "Neural networks for nonlinear dynamic system modelling and identification", *International Journal of Control*, vol. 56, no. 2, pp. 319–346, 1992.

[24] N. Chiras, C. Evans, D. Rees, "Nonlinear Gas Turbine Modelling Using NARMAX Structures," *IEEE Transaction on Instrumentation & Measeurement*, vol. 50, no. 4 , pp. 893-898, 2001.

[25] A. Ruano, P. J. Fleming, C. Teixeira, K. R. Rodrguez-Vzquez, C. M. Fonseca, "Nonlinear Identification of Aircraft Gas Turbine Dynamics," *Neurocomputing*, vol. 55, no. 3, pp. 551-579, 2003.

[26] G. Torella, F. Gamma, and G. Palmesano, "Neural Networks for the Study of Gas Turbine Engines Air System," *Proceedings of the International Gas Turbine Congress*, 2003.

[27] R. Bettocchi, M. Pinelli, P. R. Spina, M. Venturini, "Artificial Intelligent for the Diagnostics of Gas Turbines: Part 1Neural Network Approach," *ASME Turbo Expo 2005*, pp. 9-18, 2005.

[28] H. Asgari, X. Chen, R. Sainudiin, M. Morini, M. Pinelli, P. R. Spina, M. Venturini,"Modeling and Simulation of the Start-Up Operation of a Heavy-Duty Gas Turbine by Using NARX Models," *ASME Turbo Expo 2014*, 10 pages, 2014.

[29] N. Rooney, D. Patterson, S. Anand, and A. Tsymbal, "Dynamic integration of regression models", *International Workshop on Multiple Classiˇer Systems*. vol. LNCS 3181, pp. 164-173, Springer, 2004.

[30] G. I. Webb and Z. Zheng, “Multistrategy ensemble learning: reducing error by combining ensemble learning techniques”, *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 8, pp. 980-991, 2004.

[31] J. Liu, “RBF Neural Network Control for Mechanical Systems”, *Springer*, 2013.

[32] V. Venkatasubramanian, R. Rengaswamy, K. Yin, S. N. Kavuri, “A review of process fault detection and diagnosis: Part I: Quantitative model-based methods”, *Computers & Chemical Engineering*, vol. 27, no. 3, pp 293–311.

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

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

[35] B. V. Dasarathy and B. V. Sheela, “Composite classifier system design: concepts and methodology,” *Proceedings of the IEEE*, vol. 67, no. 5, pp. 708–771, 1979.

[36] S. Geman, E. Bienenstock, R. Doursat, “Neural networks and the bias/variance dilemma”. *Neural Computation*, vol. 4, no. 1, pp 1–58, 1992.

[37] A. Bouchachia, S. Bouchachia, “Ensemble Learning for Time Series Prediction”, *Workshop on Nonlinear Dynamics and Synchronization*, Aachen, pp. 205–212, 2008.

[38] A. Bouchachia, “Radial Basis Function Nets for Time Series Prediction”, *International Journal of Computational Intelligence Systems*, vol. 2, no. 2, pp. 147–157, 2012.

[39] P. Melin, J. Soto, O. Castillo, J. Soria, "A new approach for time series prediction using ensembles of ANFIS models", *Expert Systems with Applications*, vol. 39, pp. 3494–3506, 2012.

[40] R. Avnimelech, N. Intrator, "Boosting Regression Estimators", *Neural Computation*, vol. 11, no. 2, pp. 491–513, 1999.

[41] T. Hatanaka, N. Kondo, K. Uosaki, "Multi-Objective Structure Selection for RBF Networks and Its Application to Nonlinear System Identification", *Studies in Computational Intelligence*, vol. 16, pp. 491–505, Springer, 2006.

[42] S. C. Chiam, K. C. Tan, A. Al-Mamun, "Multiobjective Evolutionary Neural Networks for Time Series Forecasting", *Lecture Notes in Computer Science*, vol. 4403, pp. 346–360, 2007.

[43] W. M. Azmy, N. Gayar, A. F. Atiya, H El-Shishiny, "MLP, Gaussian Processes and Negative Correlation Learning for Time Series Prediction", *MultipleClassiˇer Systems 2009, LNCS*, vol. 5519, pp. 428–437, 2009.

[44] S. Oeda, I. Kurimoto, T. Ichimura, "Time Series Data Classification Using Recurrent Neural Network with Ensemble Learning", *Knowledge-Based Intelligent Information and Engineering Systems, LNCS*, vol. 4253, pp. 742–748, Springer, 2006.

[45] A. Chitra and S. Uma, "An Ensemble Model of Multiple Classifiers for Time Series Prediction", *International Journal of Computer Theory and Engineering*, vol. 2, no. 3, pp. 454–458, 2010.

[46] Md. M. Islam, X. Yao, K. Murase, "A Constructive Algorithm for Training Cooperative Neural Network Ensembles", *IEEE Transactions On Neural Network*, vol. 14, no. 4, pp. 820–834, 2003.

[47] Z. S. H. Chan, N. Kasabov, “Fast neural network ensemble learning via negative-correlation data correction,” *IEEE Transactions on Neural Networks*, vol.16, no.6, pp.1707–1710, 2005.

[48] J. D. Wichard and M. Ogorzalek “Time Series Prediction with Ensemble Models”, *Proceedings of IEEE Joint Conference on Neural Networks*, vol. 2, pp. 1625–1630, 2004.

[49] J. D. Wichard “Model Selection in an Ensemble Framework”, *Proceedings of IEEE Joint Conference on Neural Networks*, vol. 2, pp. 2187–2192, 2006.

[50] N. Kondo, T. Hatanaka, K. Uosaki, “Nonlinear Dynamic System Identification Based on Multiobjectively Selected RBF Networks”, *Proceedings of IEEE Symposium on Computational Intelligence in Multicriteria Decision Making (MCDM 2007)*, pp. 122–127, 2007.

[51] D. Wang and Y. Li, “A Novel Nonlinear RBF Neural Network Ensemble Model for Financial Time Series Forecasting”, *Proceedings of Third International Workshop on Advanced Computational Intelligence*, pp. 86–90, 2010.

[52] Y. Xiao, J. Xiao, F. Lu, S. Wang, “Ensemble ANNs-PSO-GA Approach for Day-ahead Stock E-exchange Prices Forecasting”, *International Journal of Computational Intelligence Systems*, vol. 6, no. 1, pp. 96–114, 2013.

[53] X. Caia, N. Zhang, G. K. Venayagamoorthya, D. C. Wunsch, “Time series prediction with recurrent neural networks trained by a hybrid PSO-EA algorithm”, *Neurocomputing*, vol. 70, no. 13, pp. 2342–2353, 2007.

[54] I. A. Gheyas, L. S. Smith, “A Neural Network Approach to Time Series Forecasting”, *Proceedings of the World Congress on Engineering*, vol. 2, 5 pages, 2009.

[55] J. T. Connor, R. D. Martin, L. E. Atlas, "Recurrent neural networks and robust time series prediction,", *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp.240–254, 1994.

[56] C. L. Giles, S. Lawrence, A. Tsoi, "Noisy Time Series Prediction using Recurrent Neural Networks and Grammatical Inference", *Machine Learning*, vol. 44, no. 1, pp 161-183, Springer, 2001.

[57] X. Cai, N. Zhang, G. K. Venayagamoorthy, D. C. Wunsch, "Time series prediction with recurrent neural networks using a hybrid PSO-EA algorithm", *Proceedings of IEEE International Joint Conference on Neural Networks*, vol.2, pp.1647–1652, 2004.

[58] X. Cai, N. Zhang, G. K. Venayagamoorthy, D. C. Wunsch, "Time series prediction with recurrent neural networks trained by a hybrid PSOEA algorithm", *Neurocomputing*, vol.70, pp.2347–2353, 2007.

[59] R. Bone, M. Assaad, M. Crucianu. "Boosting Recurrent Neural Networks for Time Series Prediction". RFAI Publication, Artificial Neural Nets and Genetic Algorithms, Proceedings of the International Conference, *Lecture Nontes in Computer Science*, pp. 18–22, Springer.

[60] N. I. Sapankevych, R. Sankar, "Time Series Prediction Using Support Vector Machines: A Survey", *IEEE Computational Intelligence Magazine*, vol. 4, no. 2, pp. 24–38, 2009.

[61] T. B. Trafalis and H. Ince, "Support vector machine for regression and applications to financial forecasting, in *Proc. IEEE-INNS-ENNS Int. Joint Conf. on Neural Networks (IJCNN 2000)*, vol. 6, pp. 348–353.

[62] F. E. H. Tay and L. J. Cao, "Application of support vector machines in financial time series forecasting, *Omega*, vol. 29, no. 4, pp. 309–317, 2001.

[63] T. Van Gestel, J. A. K. Suykens, D. E. Baestaens, A. Lambrechts, G. Lanckriet, B. Vandaele, B. De Moor, and J. Vandewall, “Financial time series prediction using least squares support vector machines within the evidence framework, *IEEE Transaction on Neural Networks*, vol. 12, no. 4, pp. 809–821, 2001.

[64] F. E. H. Tay and L. J. Cao, “Improved financial time series forecasting by combining support vector machines with self-organizing feature map”, *Intelligent Data Analysis*, vol. 5, no. 4, pp. 339–354, 2001.

[65] F. E. H. Tay and L. J. Cao, “Modified support vector machines in financial time series forecasting”, *Neurocomputing*, vol. 48, pp. 847–861, Oct. 2002.

[66] F. E. H. Tay and L. J. Cao, “ϵ-descending support vector machines for financial time series forecasting”, *Neural Processessing Letters*, vol. 15, no. 2, pp. 179–195, 2002.

[67] H. Yang, I. King, and L. Chan, “Non-fixed and asymetrical margin approach to stock market prediction using support vector regression”, in *Proceedings of 9th International Conference on Neural Information Processing*, vol. 3, pp. 1398–1402, 2002.

[68] H. Yang, L. Chan, and I. King, “Support vector machine regression for volatile stock market prediction,” in *Proceedings 3rd Internationa Conference on Intelligent Data Engineering and Automated Learning*, Springer-Verlag, pp. 391–396, 2002.

[69] A. Abraham, N. S. Philip, and P. Saratchandran, “Modeling chaotic behavior of stock indices using intelligent paradigms,” *International Journal of Neural, Parallel, and Scientiˇc Computatation*, vol. 11, no. 1, pp. 143–160, 2003.

[70] H. Yang, "Margin variations in support vector regression for the stock market prediction", Ph.D. dissertation, Chinese Univ. of Hong Kong, June 2003.

[71] P. Ongsritrakul and N. Soonthornphisaj, "Apply decision tree and support vector regression to predict the gold price," *Proceedings of the International Joint Conference on Neural Networks*, vol. 4, pp. 2488–2492, 2003.

[72] W. Lu, W. Wang, A. Y. T. Leung, S.-M. Lo, R. K. K. Yuen, Z. Xu, and H. Fan, "Air pollutant parameter forecasting using support vector machines," *Proceedings of the International Joint Conference on Neural Networks*, vol. 1, pp. 630–635, 2002.

[73] T. B. Trafalis, B. Santosa, and M. B. Richman, "Prediction of rainfall from WSR radar using kernel-based methods", *International Journal of Smart Engineering System Design*, vol. 5, no. 4, pp. 429–438, 2003.

[74] W. Wang, Z. Xu, and J. W. Lu, "Three improved neural network models for air quality forecasting", *Engineering Computatation*, vol. 20, no. 2, pp. 192–210, 2003.

[75] M. Mohandes, "Support vector machines for short-term load forecasting," *International Journal of Energy Resources*, vol. 26, no. 4, pp. 335–345, Mar. 2002.

[76] D. C. Sansom and T. K. Saha, "Energy constrained generation dispatch based on price forecasts including expected values and risk", *Proceedings of the IEEE Power Energy Society General Meeting* vol. 1, pp. 261–266, 2004.

[77] L. Tian and A. Noore, "A novel approach for short-term load forecasting using support vector machines", *International Journal of Neural Systems*, vol. 14, no. 5, pp. 329–335, 2004.

[78] B. J. Chen, M. W. Chang, and C. J. Lin, “Load forecasting using support vector machines: A study on EUNITE competition 2001”, *IEEE Transaction on Power Systems*, vol. 19, no. 4, pp. 1821–1830, Nov. 2004.

[79] J. Yang and Y. Zhang, “Application research of support vector machines in condition trend prediction of mechanical equipment,” *Proceedings of the 2nd International Symposiom on Neural Networks*, Lecture Notes in Computer Science, vol. 3498, pp. 857–864, 2005.

[80] D. Mandic and J. Chambers, “Recurrent neural networks for prediction: learning algorithms, architectures and stability”, *John Wiley & Sons, Inc.*, 2001.

[81] L. Tsungnan, B. G. Horne, C.Lee Giles, “How Embedded Nlemory in Recurrent Neural Network Architectures Helps Learning Long-term Temporal Dependencies”, Computer Science Technical Report CS-TR-3626 and UMIACS, University of Maryland, College Park, Ml 20742, 1996.

[82] D. R. Seidl, R. D. Lorenz, “A structure by which a recurrent neural network can approximate a nonlinear dynamic system,” *International Joint Conference on Neural Networks*, pp. 709–714 vol. 2, no. 1, 1991.

[83] K. S. Narendra, K. Parthasarathy, “Identification and control of dynamical systems using neural networks,” *IEEE Transactions on Neural Networks*, vol. 1, no. 1, pp.4–27, 1990.

[84] J. Sjberg, Q. Zhang, L. Ljung, A. Benveniste, B. Delyon, P. Glorennec, H. Hjalmarsson, A. Juditsky, “Nonlinear black-box modeling in system identification: a unified overview”, *Automatica*, vol. 31, no. 12, pp. 1691–1724, 1995.

[85] W. C. Hong, P. F. Pai, C. T. Chen, and P. T. Chang, “Recurrent support vector machines in reliability prediction,” *Lecture Notes in Computer Science*, vol. 3610, pp. 619–7629, 2005.

[86] W. C. Hong and P.-F. Pai, “Predicting engine reliability using support vector machines,” *International Journal of Advanced Manufufacturing Technology*, vol. 28, no. 1, pp. 154–161, 2006.

[87] S. Gezici, H. Kobayashi, and H. V. Poor, “A new approach to mobile position tracking,” *Proceedings of IEEE Sarnoff Symposiom on Advances in Wired and Wireless Communications*, pp. 204–207, 2003.

[88] C. J. Huang and C. L. Cheng, “Application of support vector machines to admission control for proportional differentiated services enabled internet servers,” *Proceedings of International Conference on Hybrid Intelligent Systems*, pp. 248–253, 2004.

[89] X. Liu, J. Yi, and D. Zhao, “Adaptive inverse disturbance cancelling control system based on least square support vector machines”, *American Control Conference*, pp. 2625–2629, 2005.

[90] Q. Yang and S. Xie, “An application of support vector regression on narrow-band interference suppression in spread spectrum systems”, *Lecture Notes Computer Science*, vol. 3611, pp. 442–450, 2005.

[91] Y. F. Deng, X. Jin, and Y. X. Zhong, “Ensemble SVR for prediction of time series”, *Proceedings of the 4th International Conference on Machine Learning and Cybernetics*, pp. 3528–3534, 2005.

[92] J. Ni, W. Tang, Y. Xing, “A Simple Algebra for Fault Tree Analysis of Static and Dynamic Systems,” *IEEE Transactions on Reliability*, vol.62, no.4, pp. 846–861, 2013.

[93] P. M. Frank, B. Kppen-Seliger, “Fuzzy logic and neural network applications to fault diagnosis”, *International Journal of Approximate Reasoning*, vol 16, no. 1, pp. 67–88, 1997.

[94] W. Yan, F. Xue, “Jet engine gas path fault diagnosis using dynamic fusion of multiple classifiers”, *in the proceedings of IEEE World Congress on Computational Intelligence*, pp.1585–1591, 2008.

[95] T. Kobayashi, D. L. Simon, “Application of a Bank of Kalman Filters for Aircraft Engine Fault Diagnostics”, *Proceedings ASME TurboExpo*, pp. 461–470, 2003.

[96] W. Xue, Y. Guo, X. Zhang, “A Bank of Kalman Filters and a Robust Kalman Filter Applied in Fault Diagnosis of Aircraft Engine Sensor/Actuator,” *Innovative Computing, Information and Control*, 10 pages, 2007.

[97] E. Naderi, N. Meskin, K. Khorasani, “Nonlinear fault diagnosis of jet engines by using a multiple model-based approach”, *Transactions of the ASME Engineering for Gas Turbines and Power*, vol. 134, no. 1, pp. 319–329, 2012.

[98] H. Valpola and J. Karhunen, “An Unsupervised Ensemble Learning Method for Nonlinear Dynamic State-Space Models”, *Neural Computation*, vol. 14, pp. 2647–2692 , 2001.

[99] V. Palade, C. D. Bocaniala, and L. C. Jain, “Computational Intelligence in Fault Diagnosis”, *Advanced Information and Knowledge Processing*, Springer, 2006.

[100] R. B. Joly, S. O. T. Ogaji, R. Singh, S. D. Probert, “Gas-turbine diagnostics using artificial neural-networks for a high bypass ratio military turbofan engine”, *Applied Energy*, vol. 78, no. 4, pp. 397–418, 2004.

[101] S. O. T. Ogaji, R. Singh, “Advanced engine diagnostics using artificial neural networks”, *Applied Soft Computing*, vol. 3, no. 3, pp. 259–271, 2003.

[102] P. Alexander and R. Singh, “Gas Turbine Engine Fault Diagnostics Using Fuzzy Concepts”, *Proceedings of AIAA* 1^{st} *Intelligent Systems Technical Conference*, ,2004.

[103] V. Palade, R. J. Patton, F. J. Uppal, J. Quevedo, S. Daley, “Fault diagnosis of an industrial gas turbine using neuro-fuzzy methods”, *Proceedings of the 15th IFAC World Congress*, pp. 2477–2482, 2002.

[104] W. Yang, K. Y. Lee, S. T. Junker, H. Ghezel-Ayagh, “Fault diagnosis and accommodation system with a hybrid model for fuel cell power plant,” *IEEE Power and Energy Society General Meeting - Conversion and Delivery of Electrical Energy in the 21st Century* , 8 pages, 2008.

[105] S. O. T. Ogaji, L. Marinai, S. Sampath, R. Singh, S.D. Prober, “Gas-turbine fault diagnostics: a fuzzy-logic approach,” *Applied Energy*, vol. 82, no. 1, pp. 81–89, 2004.

[106] P. J. Fleming, C. M., Fonseca, “Genetic algorithms in control systems engineering: a brief introduction,” *IEE Colloquium on Genetic Algorithms for Control Systems Engineering*, pp.1–5, 1993.

[107] J. M. Rogero, “A Genetic algorithms based optimisation tool for the preliminary design of gas turbine combustors,” PhD Thesis, Cranfield University, 2002.

[108] A. Chipperfield, P. Fleming, “Multiobjective gas turbine engine controller design using genetic algorithms,” *IEEE Transactions on Industrial Electronics*, vol.43, no.5, pp.583–587, 1996.

[109] S. O. T. Ogaji, S. Sampath, L. Marinai, R. Singh, S.D. Probert, “Evolution strategy for gas-turbine fault-diagnoses,” *Applied Energy*, vol. 81, no. 2, pp. 222–230, 2005.

[110] G. Biswas and G. Simon, N. Mahadevan, S. Narasimhan, J. Ramirez, G. Karsai, “A robust method for hybrid diagnosis of complex systems”, *5th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes*, pp. 1125–1131, 2005.

[111] S. Gupta, G. Biswas and J. W. Ramirez, “An Improved Algorithm for Hybrid Diagnosis of Complex Systems,” *15th International Workshop on Principles of Diagnosis*, 2004.

[112] J. F. D. Addison, S. Wermter, J. MacIntyre, “Effectiveness of feature extraction in neural network architectures for novelty detection,” *Ninth International Conference on Artiˇcial Neural Networks*, vol.2, pp. 976–981, 1999.

[113] K. Anders and J. Vedelsby, “Neural network ensembles, cross validation, and active learning,” *In Advances in Neural Information Processing Systems*, MIT Press, pp. 231–238, 1995.

[114] S. W. Christensen, “Ensemble construction via designed output distortion,” *Lecture Notes in Computer Science*, pp. 286–295, 2003.

[115] G. Brown, J. L. Wyatt, and T. Peter “Managing diversity in regression ensembles,” *Journal of Machine Learning Research*, pp. 1621–1650, 2005.

[116] H. Dutta, “Measuring diversity in regression ensembles,” *Proceedings of 4th Indian international conference on artiˇcial intelligence*, 2009.

[117] M. Bishop, “Pattern recognition and machine learning,” Springer, 2006.

[118] R. Polikar, “Ensemble Learning”, *Ensemble Machine Learning*, Editors: Zhang, Cha and Ma, Yunqian, pp. 1–34, 2012.

[119] J. Friedman, “On bias, variance, 0/1-loss, and the curse-of dimensionality”, *Data Mining and Knowledge Discovery*, 1997.

[120] G. Brown, “Diversity in neural network ensembles,” PhD, University of Birmingham, UK, 2004.

[121] A. Chandra and X. Yao, “Evolving hybrid ensembles of learning machines for better generalisation,”*Neurocomputing*, vol. 69, no. 7–9, pp. 686–700, Mar. 2006.

[122] W. Yan, Feng Xue, “Jet engine gas path fault diagnosis using dynamic fusion of multiple classifiers,” *Proceedings of the International Joint Conference on Neural Networks*, pp. 1585–1591, 2008.

[123] W. Yan, “A Multiple Classifier System for Aircraft Engine Fault Diagnosis,” PhD Thesis, *Rensselaer Polytechnic Institute*, 2008.

[124] L. I. Kuncheva, “Combining pattern classifiers, methods and algorithms,” New York, *Wiley*, 2005.

[125] R. E. Banfield, L. O. Hall, K.W. Bowyer, andW. P. Kegelmeyer, “Ensemble diversity measures and their application to thinning,” *Information Fusion*, vol. 6, no. 1, pp. 49–62, 2005.

[126] L. I. Kuncheva and C. J. Whitaker, “Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy,” *Machine Learning*, vol. 51, no. 2, pp. 181–207, 2003.

[127] L. I. Kuncheva, “That elusive diversity in classifier ensembles,” *Pattern Recognition and Image Analysis*, Lecture Notes in Computer Science, vol. 2652, 2003, pp. 1126–1138

[128] L. Breiman, “Bagging predictors,”*Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.

[129] P. Domingos, “Why does bagging work? a bayesian account and its implications”, *International Conference on Knowledge Discovery and Data Mining.*, pp. 155-158, AAAI Press, 1997.

[130] P. M. Granitto, P. F. Verdes, and H. A. Ceccatto, “Neural network ensembles: evaluation of aggregation algorithms”, *Artiˇcial Intelligence*, vol. 163, no. 2, pp. 139-162, 2005.

[131] D. Partridge and W. B. Yates, “Engineering multiversion neural-net systems”, *Neural Computation*, vol. 8, no. 4, pp. 869-893, 1996.

[132] S. B. Kotsiantis and P. E. Pintelas, “Selective averaging of regression models”, *Annals of Mathematics, Computing & Teleinformatics*, vol. 1, no. 3, pp. 65-74, 2005.

[133] N. Rooney, D. Patterson, S. Anand, A. Tsymbal, “Dynamic integration of regression models”, *International Workshop on Multiple Classiˇer Systems*, vol. LNCS 3181, pp. 164-173, Springer, 2004.

[134] P. Guilherme, F. J. V. Zuben, “The influence of the pool of candidates on the performance of selection and combination techniques in ensembles”, *International Joint Conference on Neural Networks*, pp. 10588-10595, 2006.

[135] H. Drucker, C. Cortes, L. D. Jackel, Y. LeCun, and V. Vapnik, “Boosting and other ensemble methods,” *Neural Computation*, vol. 6, no. 6, pp. 1289–1301, 1994.

[136] D. H. Wolpert, “Stacked generalization,”*Neural Networks*, vol. 5, no. 2, pp. 241–259, 1992.

[137] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton, “Adaptive mixtures of local experts,” *Neural Computation*, vol. 3, no. 1, pp. 79–87, 1991.

[138] M. J. Jordan and R. A. Jacobs, “Hierarchical mixtures of experts and the EMalgorithm,” *Neural Computation*, vol. 6, no. 2, pp. 181–214, 1994.

[139] S. Sina Tayarani-Bathaie, Z.N. Sadough Vanini, K. Khorasani, “Dynamic neural network-based fault diagnosis of gas turbine engines”, *Neurocomputing*, vol. 125, pp. 153–165, 2014.

[140] S. Tayarani-Bathaie, Z. S. Vanini, K. Khorasani, “Fault detection of gas turbine engines using dynamic neural networks,” *IEEE Canadian Conference on Electrical & Computer Engineering (CCECE)*, 5 pages, 2012.

[141] A. J. Volponi, “Use of hybrid engine modeling for on-board module performance tracking”, *Proceedings of the ASME Turbo Expo*, pp. 525–533, 2005.

[142] T. Chen, J. G. Sun, “Rough set and neural network based fault diagnosis for aeroengine gas path” ,*Proceedings of the ASME Turbo Expo*, pp. 535–539, 2005.

[143] I. Loboda, Y. Feldshteyn, V. Ponomaryov, “Neural networks for gas turbine fault identification: Multilayer perceptron or radial basis network?” *Proceedings of the ASME Turbo Expo*, pp. 465–475, 2011.

[144] H. Xiao, N. Eklund, K. Goebel, W. Cheetham, “Hybrid Change Detection for Aircraft Engine Fault Diagnostics”, *Proceedings of IEEE Aerospace Conference*, pp.1–10, 2007.

[145] J. Zhang, “Improved on-line process fault diagnosis through information fusion in multiple neural networks”, *Computers & Chemical Engineering*, vol. 30, no. 3, pp. 558–571, 2005.

[146] W. Yan, F. Xue, “Jet engine gas path fault diagnosis using dynamic fusion of multiple classifiers,”, *Proceedings of IEEE World Congress on Computational Intelligence*, pp.1585–1591, 2008.

[147] R. Mohammadi, E. Naderi, K. Khorasani, S. Hashtrudi-Zad, “Fault diagnosis of gas turbine engines by using dynamic neural networks”, *Proceedings of the ASME Turbo Expo*, pp. 365–373, 2010.

[148] H. Xiao, N. Eklund, K. Goebel, “A data fusion approach for aircraft engine fault diagnostics,” *Proceedings of the ASME Turbo Expo*, pp. 767–775, 2007.

[149] A. Varma, P. Bonissone, W. Yan, N. Eklund, K. Goebel, N.Iyer, S.Bonissone, “Anomaly detection using non-parametric information”, *Proceedings of the ASME Turbo Expo*, pp. 813–821, 2007.

[150] W. Donat, K. Choi, W. An, S. Singh, K. Pattipati, “Data visualization, data reduction and classifier fusion for intelligent fault detection and diagnosis in gas turbine engines”, *Proceedings of the ASME Turbo Expo*, pp. 883–892, 2007.

[151] A. J. Volponi, “Use of hybrid engine modeling for on-board module performance tracking” *Proceedings of the ASME Turbo Expo*, pp. 525–533.

[152] J. Huang, M. Wang, “Multiple Classifiers Combination Model for Fault Diagnosis Using Within-class Decision Support,”, *Proceedings of Information Science and Management Engineering (ISME)*, pp.226–229, 2010.

[153] S. C. Gu, Y. Tan and X. G. He, “Orthogonal quadratic discriminant functions for face recognition”, *Advances in Neural Networks*, pp. 466–475, Springer, 2009.

[154] J. Amanda, C. Sharkey, “Types of Multinet System”, *Multiple Classiˇer Systems, Lecture Notes in Computer Science*, pp. 108–117, Springer, 2002.

[155] A. J. C.Sharkey, G. O. Chandroth, N. E. Sharkey, "A Multi-Net System for the Fault Diagnosis of a Diesel Engine", *Neural Computing and Applications*, vol. 9, pp 152–160, 2000.

[156] K. Choi, S. Singh, A. Kodali, K. Pattipati, R. Sheppard, J. W. Namburu, S. M. Chigusa et al., "Novel Classifier Fusion Approaches for Fault Diagnosis in Automotive Systems", *IEEE Transaction on Instrumentation and Measurement*, vol. 58, no. 3, pp. 602–611, 2009.

[157] Y. Lei, M. J. Zuo, Z. He, Y. Zi, "A multidimensional hybrid intelligent method for gear fault diagnosis", *Expert Systems with Applications*, vol. 37, no. 2, pp. 1419–1430, 2010.

[158] G. Niu, T. Han, B. S. Yang, A. Chit, Ch. Tan, "Multi-agent decision fusion for motor fault diagnosis", *Mechanical Systems and Signal Processing*, vol. 21, no. 3, pp. 1285–1299, 2007.

[159] L. Oukhellou, A. Debiolles, T. Denaux, P. Aknin, "Fault diagnosis in railway track circuits using Dempster-Shafer classifier fusion", *Engineering Applications of Artiˇcial Intelligence*, vol. 23, no. 1, pp. 117–128, 2010.

[160] W. C. Chen, P. P. K. Chan, W. W. Y. Ng, D. S. Yeung, "Multiple classifier systems combined with localized generalization error for fault diagnosis of power transformers", *Proceedings of International Conference on Machine Learning and Cybernetics*, vol. 3, pp. 1464–1469, 2010.

[161] P. P. Bonissone, N. Iyer, "Soft computing applications to prognostics and health management (PHM): leveraging field data and domain knowledge", *Proceedings of the international work conference on Artiˇcial neural networks*, pp. 928–939, Springer, 2007.

[162] P. Bonissone, X. Hu, R. Subbu, "A Systematic PHM Approach for Anomaly Resolution: A Hybrid Neural Fuzzy System for Model Construction", *Proceeding of Annual Conference of the Prognostics and Health Management Society*, San Diego, pp. 2009.

[163] H. A. Nozari, M. A. Shoorehdeli, S. Simani, H. D. Banadaki, “Model-based robust fault detection and isolation of an industrial gas turbine prototype using soft computing techniques”, *Neurocomputing*, vol. 91, pp. 29–47, 2012.

[164] D. Xu, M. Wu, J. An, “Design of an expert system based on neural network ensembles for missile fault diagnosis,” *Proceedings of IEEE International Conference on Robotics, Intelligent Systems and Signal Processing*, vol. 2, pp. 903–908, 2003.

[165] N. C. Oza, K. Tumer, I. Y. Tumer, E. M. Huff, “Classification of Aircraft Maneuvers for Fault Detection”, *Multiple Classiˇer Systems, Lecture Notes in Computer Science*, pp. 375–384, Springer, 2003.

[166] A. Lipnickas, “Two-Stage Neural Networks Based Classifier System for Fault Diagnosis”, *Computational Intelligence in Fault Diagnosis, Advanced Information and Knowledge Processing*, pp 209–230, Springer, 2006.

[167] E. Filippi, M. Costa, E. Pasero, “Multi-layer perceptron ensembles for increased performance and fault-tolerance in pattern recognition tasks”, *Proceedings of IEEE International Conference on Neural Networks*, pp. 2901–2906, 1994.

[168] R. Lowen, A. Verschoren, “An Integrated Fuzzy Inference-based Monitoring, Diagnostic, and Prognostic System for Intelligent Control and Maintenance”, *Foundations of Generic Optimization, Mathematical Modelling: Theory and Applications*, pp. 203–222, Springer, 2008.

[169] W. Z. Yan, J. C. Li and K. F. Goebel, “On improving performance of aircraft engine gas path fault diagnosis”, *Transactions of the Institute of Measurement and Control*, vol. 31, no.3, pp. 275–291, 2009.

[170] C. Alippi, G. Boracchi, V. Puig, M. Roveri, "An Ensemble Approach to Estimate the Fault-Time Instant, ", *Proceedings of IEEE International Conference on Intelligent Control and Information Processing*, 2013.

[171] Ch. Ren, J. F. Yan, Z. H Li, "Improved ensemble learning in fault diagnosis system", *IEEE International Conference on Machine Learning and Cybernetics*, vol. 1, pp. 54–60, 2009.

[172] Y.L. Murphey, Z. Chen, M. Abou-Nasr, R. Baker, T. Feldkamp, I. Kolmanovsky, "Ensembles of neural networks with generalization capabilities for vehicle fault diagnostics", *IEEE International Joint Conference on Neural Networks*, pp. 2188–2194, 2009.

[173] R. K. Shahzad, N. Lavesson, "Veto-based Malware Detection," *IEEE International Conference on Availability, Reliability and Security*, pp.47–54, 2012.

[174] G. O. Chandroth, "Diagnostic Classifier Ensembles: Enforcing Diversity for Reliebility in the Combination," *Sheˇeld University, PhD Dissretation*, 1999.

[175] Q. Yang, C. Liu, D. Zhang, D. Wu, "A New Ensemble Fault Diagnosis Method Based on K-means Algorithm", *International Journal of Intelligent Engineering & Systems*, vol. 5, no. 5, 2012.

[176] G. Georgoulas, T. Loutas, Ch. D. Stylios, V. Kostopoulos, "Bearing fault detection based on hybrid ensemble detector and empirical mode decomposition", *Mechanical Systems and Signal Processing*, vol. 41, no. 1, pp. 510–525, 2013.

[177] Y. Xu, D. Zhang, Y. Wang, "Active Diverse Learning Neural Network Ensemble Approach for Power Transformer Fault Diagnosis", *Journal of Networks*, vol. 5, no. 10, October 2010.

[178] B. Y. Dong, G. Ren, “Analog Circuit Fault Diagnosis Using AdaBoost with SVM-Based Component Classifers”, *Advanced Materials Research*, pp. 1414–1417, 2012.

[179] F. Lu, T. B. Zhu, Y. Q. Lv, “Data-Driven Based Gas Path Fault Diagnosis for Turbo-Shaft Engine”, *Applied Mechanics and Materials*, vol. 249, pp. 400–404,2012.

[180] B. K. Kestner, Y. K. Lee, G. Voleti, D. N. Mavris, V. Kumar, T. Lin, “Diagnostics of Highly Degraded Industrial Gas Turbines Using Bayesian Networks”, *Proceedings of ASME Turbo Expo*, pp. 39–49, 2011.

[181] R. Ganguli, R. Verma, N. Roy, “Soft Computing Application for Gas Path Fault Isolation”, *Proceedings of ASME Turbo Expo*, pp. 499–508, 2004.

[182] C. Romessis, K. Mathioudakis, “Bayesian Network Approach for Gas Path Fault Diagnosis”, *Proceedings of ASME Turbo Expo*, pp. 691–699, 2004.

[183] S. Sampath, R. Singh, “An Integrated Fault Diagnostics Model Using Genetic Algorithm and Neural Networks”, *Proceedings of ASME Turbo Expo*, pp. 749–758, 2004.

[184] A. J. C. Sharkey and N. E. Sharkey, “Combining diverse neural nets,” *Knowledge Eng. Rev.*, vol. 12, no. 3, pp. 1—17, 1997.

[185] A. J. C. Sharkey, “On combining artificial neural nets,” *Connection Sci.* vol. 8, no. 3, pp. 299–314, 1996.

[186] M. Md. Islam, Y. Xin Yao, K. Murase, “A constructive algorithm for training cooperative neural network ensembles,” *IEEE Transactions on Neural Networks* , vol.14, no.4, pp. 820–834, 2003.

[187] L. K. Hansen, P. Salamon, ”Neural Network Ensembles,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 10, pp. 993–1001, October, 1990.

[188] A. Khotanzad and C. Chung, “Hand written digit recognition using BKS combination of neural network classifiers,” *in Proc. IEEE Southwest Symp. Image Anal. Interpretation*, pp. 94—99, 1994.

[189] S. Hashem, “Optimal linear combinations of neural networks,” *Neural Networks*, vol. 10, no. 4, pp. 599—614, 1997.

[190] D. W. Opitz and J. W. Shavlik, “Actively searching for an effective neural-network ensemble,” *Connection Science*, vol. 8, no. 3, pp. 337—353, 1996.

[191] D. H. Wolpert. “Stacked generalization,” *Neural Networks*, 5:241–259, 1992.

[192] M. P. Perrone, L. N. Cooper, “When networks disagree: Ensemble methods for hybrid neural networks”, *Neural Networks for Speech and Image Processing* pp. 126–142, 1994.

[193] G. Brown, “Diversity in neural network ensembles”, PhD, University of Birmingham, UK, 2004.

[194] G. Brown, J. Wyatt, R. Harris, X. Yao, “Diversity creation methods: a survey and categorisation,” *Information Fusion*, vol. 6, no. 1, pp. 5–20, 2005.

[195] Y. Liu and X. Yao, “Ensemble learning via negative correlation,” *Neural Networks*, vol. 12, no. 10, pp. 1399–1404, 1999.

[196] Y. Freund and R. E. Schapire, "Decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[197] S. Bathaei, "Fault Detection and Isolation of Jet Engines Using Neural Networks, Master Thesis", *Concordia University*, 2012.

[198] Y. A. Borgne, "Bias-Variance trade-off characterization in a classification problem What differences with regression ", *Technical Report, Universite Libre de Bruxelles - Belgium.*

[199] E. Naderi, N. Meskin, and K. Khorasani, "Nonlinear Fault Diagnosis of Jet Engines by Using a Multiple Model-Based Approach," *J. Eng. Gas Turbines Power*, 2012.

[200] R. Mohammadi, "Fault Diagnosis of Hybrid Systems with Applications to Gas Turbine Engines," PhD Thesis, *Concordia University*, 2009.

[201] F. Roli, G. Giacinto, and G. Vernazza, "Methods for Designing Multiple Classifier Systems", *Multiple Classiˇer Systems*, vol. 2096, pp.78–87, 2001.

[202] A. J. Volponi, H. Depold, R. Ganguli, "The use of Kalman Filters and Nural Networks Methodologies in Gas Turbine Fault Diagnosis: A Comparison Study," *Proceeding of Turbo Expo*, pp. 8-11, 2000.

[203] P. J. Lu, M. C. Zhang , T. C. Su, "An Evaluation of Engine Fault Diagnostics Using Artificial Neural Networks," *Proceedings of ASME Turbo Expo*, 2000.

[204] V. Vapnik, "The Nature of Statistical Learning Theory," *Springer-Verlag*, New York, 1995.

[205] B. Sun, J. Zhang, S. Zhang, "An investigation of artificial neural network (ANN) in quantitative fault diagnosis for turbofan engine," *proceedings of ASME Turbo Expo*, 2000.

[206] A. M. Ison, W. Li, C. J. Spanos, "Fault diagnosis of plasma etch equipment," *Semiconductor Manufacturing Conference Proceedings, IEEE International Symposium on* , vol., no., pp.B49–B52, 6–8 Oct 1997.

[207] E. J. Weyuker, T. J. Ostrand, and R. M. Bell. "Comparing negative binomial and recursive partitioning models for fault prediction," *In Proceedings of the 4th international workshop on Predictor models in software engineering* , 2008.

[208] D. Jearkpaporn, D. C. Montgomery, G. C. Runger, and C. M. Borror, "Process monitoring for correlated gamma-distributed data using generalized-linear-modelbased control charts," *Quality and Reliability Engineering International*, pp. 477–491, 2003.

[209] K. R. Skinner, D. C. Montgomery, and G. C. Runger, "Process monitoring for multiple count data using generalized linear model-based control charts," *International Journal of Production Research*, pp. 1167–1180, 2003.

[210] T. Brotherton, G. Jahns, J. Jacobs, D. Wroblewski, "Prognosis of faults in gas turbine engines," *Aerospace Conference Proceedings*, vol. 6, pp. 163–171, 2000.

[211] A. Bajwa, and D. Kulkarni, "Engine Data Analysis Using Decision Trees, " *36th Joint Propulsion Conference and Exhibit*, 2000.

[212] M. Aksela, "Comparison of Classifier Selection Methods for Improving Committee Performance", Technical Report, *Helsinki University of Technology.*

[213] X. Yao and Y. Liu, "Evolving neural network ensembles by minimization of mutual information," *International Journal of Hybrid Intelligent Systems*, 2004.

[214] R. E. Schapire, "The strength of weak learnability," *Machine Learning*, vol. 5, no. 2, pp. 197–227, June 1990

[215] A. Yazdizadeh, K. Khorasani, Adaptive time delay neural network structures for nonlinear system identification, *Neurocomputing*, Volume 47, Issues 14, Pages 207–240, ISSN 0925–2312, 2002.

[216] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of Control, Signals and Systems*, 1989.

[217] S. A. Billings. "Nonlinear System Identification: NARMAX Methods in the Time, Frequency, and Spatio-Temporal Domains", Wiley, ISBN 978-1-1199-4359-4, 2013.

[218] J. Sjoberg, Q. Zhang, L. Ljung, A. Benveniste, B. Delyon, P.Y. Glorennec, H. Hjalmarsson, A. Juditsky, "Nonlinear black-box modeling in system identification: a unified overview", *Automatica*, vol. 31, no. 12, Pages 1691–1724.

[219] G. Brown, J. Wyatt, R. Harris, X. Yao, "Diversity creation methods: a survey and categorisation", *Information Fusion*, vol. 6, no. 1, pp. 5–20, 2005.

[220] W. Yates, D. Partridge, "Use of methodological diversity to improve neural network generalization", *Neural Computing and Applications*, vol. 4, no.2, pp. 114-128, 1996.

[221] D. Opitz, R. Macli, "Popular Ensemble Methods: An Empirical Study", *Journal of Artiˇcial Intelligence Research*, vol. 11, pp. 169–198, 1999.

[222] D. Partridge, “Network generalization differences quantified”, *Neural Networks*, vol. 9, no. 2, pp. 263–271, 1996.

[223] J. Mendes-Moreira, C. Soares, A. M. Jorge, J. F. D. Sousa, “Ensemble approaches for regression: A survey”, *ACM Computing Surveys (CSUR)*, vol. 45, no. 1, article no. 10, 2012.

[224] C. J. Merz, M. J. Pazzani, “A principal components approach to combining regression estimates”, *Machine Learning*, vol. 36, pp. 9-32, 1999.

[225] J. Kivinen, M. K. Warmuth, “Exponentiated gradient versus gradient descent for linear predictors”, *Information and Computation*, vol. 132, no. 1, pp. 1-63, 1997.

[226] W. Visser and M. Broonhead, “GSP, a generic object-oriented gas turbine simulation environment,” *ASME Turbo Expo*, 2000.

[227] H. Saravanamuttoo, G. Rogers, and H. Cohen, “Gas Turbine Theory,” *Pearson Education*, 2001.

[228] L. Tang, I. Technol, X. Zhang, J. DeCastro, “Diagnosis of engine sensor, actuator and component faults using a bank of adaptive nonlinear estimators,”, *IEEE Aerospace Conference*, 11 pages, 2011.

[229] I. S. Diakunchak, “Performance deteriorations in industrial gas turbines,”*ASME Journal of Engineering for Gas turbines and Power*, vol. 114, no. 2, pp. 161–168, 1992.

[230] Turbotect, Internet: “http:www.turbotect.comgalleryexamples_of_some_severe_gas_turbine _compressor_fou.html”, Jan 12, 2015.

[231] C. B. Meher-Homji, A. Bromley, “Gas Turbine Axial Compressor Fouling and Washing,”*Turbomachinery Symposium*, 2004.

[232] N. Cumpsty, “Jet Propulsion”, *Cambridge University Press*, Cambridge, UK, 2003.

[233] HEATTOP Project, Work Package 1, 2007, “Definitions and Sensor Specifications.”

[234] HEATTOP Project, “Accurate High Temperature Engine Aero-Thermal Measurements for Gas-Turbine Life Optimization, Performance and Condition Monitoring”, 2011.

[235] M. Massini, R. J. Miller, H. P. Hodson, “A New Intermittent Aspirated Probe for the Measurement of Stagnation Quantities in High Temperature Gases”, Journal of Tourbomachinery, vol. 133, no.4, 6 pages, 2011.

[236] M. Scervini, C. Rae, “An Improved Nickel Based MIMS Thermocouple for High Temperature Gas Turbine Applications”, *Journal of Engineering for Gas Turbines and Power*, vol. 135, 6 pages, 2013.

[237] Rolls Royce, “The Jet Engine”, ISBN 0902121235, 1996.

Chapter 6

Appendix

Table 6.1: Summary of MLP-NARX model construction for identification of compressor temperature

# neurons	**# delays**	**# training samples**	**RMSE**$^{T_C}_{total}$	**RMSE**$^{T_C}_{training}$	**RMSE**$^{T_C}_{test}$	**%RMSE**$^{T_C}_{total}$	**mean** (μ_{ae})	**Std** (σ_{ae})
10	4	1200	2.7879	3.0934	2.5642	0.42469	2.1886	1.7272
10	5	1200	20.9207	21.4008	20.5946	3.1769	16.2987	13.118
10	6	1200	2.3186	3.0833	1.6198	0.35368	1.4804	1.7848
10	7	1200	19.9439	31.4826	1.5435	3.0499	6.9422	18.6998
10	8	1200	2.7822	3.6898	1.9563	0.42456	1.7735	2.144
10	4	1501	50.7861	51.1885	50.3802	7.6758	49.3208	12.1133
10	5	1501	2.4312	2.4207	2.4417	0.36956	2.0321	1.3349
10	6	1501	2.2036	2.5874	1.7366	0.33616	1.431	1.676
10	7	1501	3.6805	3.4781	3.8725	0.55892	3.205	1.8098
10	8	1501	2.177	2.4418	1.8749	0.33189	1.5631	1.5155
10	4	1801	1.5442	1.578	1.4921	0.23453	1.3122	0.81427
10	5	1801	1.6296	1.7687	1.3951	0.24789	1.3153	0.9623

Table 6.1: Summary of MLP-NARX model construction for identification of compressor temperature

# neurons	# delays	# training samples	RMSE$^{T_C}_{total}$	RMSE$^{T_C}_{training}$	RMSE$^{T_C}_{test}$	%RMSE$^{T_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
10	6	1801	43.6162	43.5828	43.6664	6.6003	43.6042	1.0263
10	7	1801	80.5395	79.7868	81.6561	12.1774	80.3288	5.8234
10	8	1801	2.7925	3.0573	2.3394	0.42497	2.2723	1.6235
11	4	1200	2.3929	2.8235	2.0566	0.36452	1.8417	1.528
11	5	1200	86.0789	85.6501	86.3633	13.0184	85.9729	4.2713
11	6	1200	75.2982	75.0649	75.4532	11.3917	75.2709	2.0268
11	7	1200	3.5717	4.1891	3.0927	0.54406	2.7173	2.3185
11	8	1200	4.6462	6.7439	2.3806	0.71023	2.2969	4.0394
11	4	1501	2.6398	2.5909	2.6879	0.40145	2.2234	1.4232
11	5	1501	54.435	54.7195	54.1488	8.2313	54.3311	3.3613
11	6	1501	4.0782	4.2999	3.8437	0.62113	3.1776	2.5568
11	7	1501	1.4337	1.8878	0.7389	0.21885	0.64245	1.2819
11	8	1501	9.6467	9.2376	10.0394	1.472	5.8961	7.6363
11	4	1801	1.7293	2.0072	1.1966	0.26349	1.2099	1.2357
11	5	1801	118.2379	117.8377	118.836	17.8814	118.0842	6.0289
11	6	1801	2.194	2.5873	1.4109	0.33452	1.5441	1.5588
11	7	1801	3.663	4.2249	2.6011	0.55827	2.7901	2.3738
11	8	1801	1.8026	1.8981	1.649	0.27394	1.5337	0.94741
12	4	1200	95.7778	95.3331	96.0729	14.4851	95.6603	4.7432
12	5	1200	2.2768	2.6802	1.9626	0.34685	1.7141	1.4988
12	6	1200	54.5425	54.4929	54.5755	8.2538	50.0748	21.6231
12	7	1200	2.2538	2.4209	2.1353	0.34275	1.8984	1.215
12	8	1200	2.972	3.6885	2.3778	0.45271	2.2093	1.9883

Table 6.1: Summary of MLP-NARX model construction for identification of compressor temperature

# neurons	# delays	# training samples	RMSE$^{T_C}_{total}$	RMSE$^{T_C}_{training}$	RMSE$^{T_C}_{test}$	%RMSE$^{T_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
12	4	1501	1.6918	1.6993	1.6842	0.25716	1.3875	0.96808
12	5	1501	18.5113	26.1153	1.7614	2.8315	6.5389	17.3208
12	6	1501	3.0609	3.0336	3.0881	0.46546	2.5609	1.677
12	7	1501	2.4458	2.7593	2.0855	0.3722	1.797	1.6594
12	8	1501	4.0489	4.1655	3.9287	0.61464	3.3299	2.3036
12	4	1801	1.8869	1.9021	1.8637	0.28662	1.5925	1.0121
12	5	1801	56.0854	55.7542	56.5789	8.4777	55.8554	5.0751
12	6	1801	69.025	68.3138	70.0787	10.4312	68.0593	11.5075
12	7	1801	115.6588	114.8802	116.8178	17.4908	114.0775	19.0634
12	8	1801	1.2931	1.4344	1.0459	0.19672	0.9086	0.92024
13	4	1200	2.6436	3.2199	2.1765	0.4035	1.7738	1.9605
13	5	1200	119.4697	118.8057	119.9101	18.0659	119.2628	7.0291
13	6	1200	1.7008	2.3915	1.0046	0.25952	0.92274	1.4289
13	7	1200	3.4807	5.1903	1.4959	0.53189	1.4496	3.1649
13	8	1200	162.0597	161.0352	162.7388	24.5091	160.0364	25.5331
13	4	1501	32.7057	33.0116	32.3966	4.943	32.5351	3.3362
13	5	1501	63.4672	63.7249	63.2083	9.5987	63.3847	3.235
13	6	1501	28.0316	31.0267	24.6732	4.2694	14.996	23.6871
13	7	1501	109.006	109.3235	108.6874	16.4872	108.6279	9.0721
13	8	1501	11.8155	12.0117	11.6159	1.7982	9.2896	7.3026
13	4	1801	109.5966	108.9625	110.5414	16.5723	109.3952	6.6417
13	5	1801	1.8502	2.167	1.2301	0.28207	0.98749	1.5649
13	6	1801	2.7206	3.0914	2.0414	0.41443	1.999	1.8457

Table 6.1: Summary of MLP-NARX model construction for identification of compressor temperature

# neurons	# delays	# training samples	RMSE$^{T_C}_{total}$	RMSE$^{T_C}_{training}$	RMSE$^{T_C}_{test}$	%RMSE$^{T_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
13	7	1801	1.6001	1.8165	1.2044	0.24369	1.1783	1.0827
13	8	1801	86.597	86.0953	87.3445	13.0916	86.3317	6.7745
14	4	1200	2.2693	2.4864	2.1123	0.3449	1.8897	1.2567
14	5	1200	73.5512	73.0571	73.8786	11.1207	73.3719	5.1334
14	6	1200	2.5582	3.7001	1.3351	0.39059	1.2711	2.2204
14	7	1200	64.9872	64.8008	65.1112	9.8309	64.9361	2.5782
14	8	1200	1.9399	2.3602	1.5998	0.29581	1.4046	1.3383
14	4	1501	3.0572	3.152	2.9594	0.46553	2.3882	1.9091
14	5	1501	1.0016	1.0817	0.91441	0.15228	0.73773	0.67756
14	6	1501	105.5055	105.9321	105.0769	15.9566	105.3147	6.3436
14	7	1501	108.7845	109.3749	108.1905	16.4491	106.325	23.0051
14	8	1501	111.0706	109.9238	112.2064	16.8075	105.2197	35.5797
14	4	1801	1.3272	1.4965	1.0216	0.202	1.0032	0.86898
14	5	1801	5.1862	5.8258	4.0405	0.7897	3.9202	3.396
14	6	1801	63.8587	63.2542	64.7554	9.653	63.6005	5.7383
14	7	1801	4.0198	4.6657	2.7821	0.61273	3.0276	2.6448
14	8	1801	78.5176	78.3627	78.7496	11.88	78.4966	1.8173
15	4	1200	3.0191	4.0894	2.0113	0.46038	1.839	2.3948
15	5	1200	11.5477	12.252	11.0536	1.7562	9.3723	6.7472
15	6	1200	2.6182	3.3374	2.0003	0.39905	1.8458	1.8572
15	**7**	**1200**	**3.5106**	**5.5082**	**0.56647**	**0.53711**	**0.74164**	**3.432**
15	8	1200	2.0647	2.3486	1.8516	0.31423	1.6327	1.2641
15	4	1501	70.2922	70.4178	70.1664	10.6334	58.1478	39.5013

Table 6.1: Summary of MLP-NARX model construction for identification of compressor temperature

# neurons	# delays	# training samples	RMSE$^{T_C}_{total}$	RMSE$^{T_C}_{training}$	RMSE$^{T_C}_{test}$	%RMSE$^{T_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
15	5	1501	44.7489	44.8696	44.6278	6.768	34.1267	28.9501
15	6	1501	2.0963	2.493	1.6039	0.31954	1.3973	1.5629
15	7	1501	4.1934	4.6624	3.6644	0.63987	2.8925	3.0366
15	8	1501	4.8707	5.5448	4.0862	0.74261	3.4636	3.425
15	4	1801	2.128	2.4083	1.6187	0.32415	1.5784	1.4274
15	5	1801	3.4745	4.2668	1.6931	0.53056	1.7334	3.0117
15	6	1801	2.2914	2.8764	0.84487	0.34997	0.87841	2.1167
15	7	1801	5.401	5.8805	4.5882	0.82196	4.3454	3.2081
15	8	1801	2.6791	2.8942	2.319	0.4076	1.9931	1.7905

Table 6.2: Summary of MLP-NARX model construction for identification of compressor pressure

# neurons	# delays	# training samples	RMSE$^{P_C}_{total}$	RMSE$^{P_C}_{training}$	RMSE$^{P_C}_{test}$	%RMSE$^{P_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
10	4	1200	0.18213	0.26799	0.086148	1.7474	0.088784	0.15905
10	5	1200	1.0284	1.6244	0.06558	8.6029	0.46376	0.91807
10	6	1200	0.086456	0.10144	0.07482	0.7319	0.065487	0.056454
10	7	1200	0.10925	0.14425	0.077618	1.0546	0.068825	0.084862
10	8	1200	4.783	4.7566	4.8004	39.6577	4.137	2.4009
10	4	1501	0.11202	0.13932	0.075388	1.0006	0.062874	0.092727
10	5	1501	0.18394	0.19742	0.16939	1.5638	0.14116	0.11796
10	6	1501	0.22605	0.31705	0.040604	1.9488	0.053844	0.21958
10	7	1501	0.71708	0.74943	0.68318	6.1644	0.38489	0.60513
10	8	1501	0.34356	0.41064	0.25963	2.9939	0.19477	0.28306

Table 6.2: Summary of MLP-NARX model construction for identification of compressor pressure

# neurons	# delays	# training samples	RMSE$^{P_C}_{total}$	RMSE$^{P_C}_{training}$	RMSE$^{P_C}_{test}$	%RMSE$^{P_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
10	4	1801	0.19124	0.23213	0.1029	1.6368	0.12894	0.14126
10	5	1801	0.087609	0.084868	0.091569	0.73536	0.069388	0.053494
10	6	1801	0.99772	1.2858	0.090768	8.6028	0.37033	0.9266
10	7	1801	1.6202	1.7103	1.4746	13.4828	1.4548	0.71315
10	8	1801	2.5661	2.4638	2.7123	21.5761	1.8921	1.7337
11	4	1200	0.12354	0.16832	0.080952	1.0549	0.073862	0.099042
11	5	1200	1.1083	1.7426	0.15357	9.5529	0.40053	1.0336
11	6	1200	0.12312	0.18319	0.053833	1.0557	0.04865	0.11312
11	7	1200	0.18312	0.24641	0.12419	1.5965	0.099219	0.15394
11	8	1200	0.19815	0.30157	0.069496	1.7042	0.077238	0.18251
11	4	1501	4.438	4.4617	4.4142	36.7173	4.4304	0.26021
11	5	1501	0.15432	0.1564	0.15221	1.3054	0.12637	0.088587
11	6	1501	5.3541	5.3673	5.3409	44.3119	5.0162	1.8724
11	7	1501	1.7701	2.502	0.069162	15.2137	0.53807	1.6867
11	8	1501	0.29538	0.3922	0.14367	2.6522	0.12672	0.26686
11	4	1801	0.17184	0.19401	0.13175	1.4687	0.13447	0.10701
11	5	1801	0.0531	0.060215	0.040119	0.44925	0.033786	0.040972
11	6	1801	0.20253	0.25712	0.057921	1.745	0.063521	0.19234
11	7	1801	0.28186	0.3321	0.18207	2.4086	0.19848	0.20015
11	8	1801	0.096033	0.11708	0.049887	0.8213	0.043779	0.085488
12	4	1200	0.10272	0.13776	0.070256	0.87585	0.061253	0.08247
12	5	1200	4.7372	4.6983	4.7629	39.127	4.7172	0.43431
12	6	1200	4.8941	4.9334	4.8677	40.6131	4.7393	1.221

Table 6.2: Summary of MLP-NARX model construction for identification of compressor pressure

# neurons	**# delays**	**# training samples**	**RMSE**$^{P_C}_{total}$	**RMSE**$^{P_C}_{training}$	**RMSE**$^{P_C}_{test}$	**%RMSE**$^{P_C}_{total}$	**mean** (μ_{ae})	**Std** (σ_{ae})
12	7	1200	0.06571	0.087446	0.045822	0.55726	0.038364	0.053357
12	8	1200	0.13281	0.18103	0.086906	1.1381	0.078402	0.10721
12	4	1501	7.2386	7.2594	7.2178	59.9556	7.2338	0.26471
12	5	1501	0.075964	0.095119	0.049915	0.64918	0.042239	0.063149
12	6	1501	0.17796	0.24437	0.060041	1.53	0.061056	0.16719
12	7	1501	7.7776	7.8118	7.7433	64.3566	7.7663	0.42043
12	8	1501	0.068131	0.083435	0.048173	0.57905	0.03969	0.055385
12	4	1801	0.30503	0.33921	0.24495	2.6813	0.24049	0.18766
12	5	1801	1.0244	1.0791	0.93632	8.6699	0.74842	0.69957
12	**6**	**1801**	**0.07754**	**0.095055**	**0.038409**	**0.70861**	**0.034802**	**0.069302**
12	7	1801	0.10034	0.12331	0.048562	0.85987	0.051821	0.08594
12	8	1801	0.26226	0.33004	0.092335	2.2502	0.15067	0.21469
13	4	1200	6.5354	6.4513	6.5909	53.8924	6.2991	1.7418
13	5	1200	0.12091	0.17572	0.061522	1.035	0.05951	0.10526
13	6	1200	0.13152	0.15096	0.11679	1.1204	0.1015	0.083656
13	7	1200	9.1264	9.0765	9.1595	75.682	8.226	3.9534
13	8	1200	0.12625	0.1864	0.058403	1.2959	0.048202	0.11671
13	4	1501	0.11394	0.15435	0.046155	0.97906	0.042831	0.1056
13	5	1501	0.21434	0.21912	0.20944	1.8191	0.16612	0.13547
13	6	1501	0.34614	0.44288	0.20843	2.9776	0.18655	0.29163
13	7	1501	0.12039	0.13573	0.10278	1.0239	0.087382	0.082833
13	8	1501	0.18682	0.19134	0.18218	1.5853	0.15063	0.11053
13	4	1801	0.082647	0.093705	0.062483	0.70218	0.060456	0.056362

Table 6.2: Summary of MLP-NARX model construction for identification of compressor pressure

# neurons	# delays	# training samples	RMSE$^{P_C}_{total}$	RMSE$^{P_C}_{training}$	RMSE$^{P_C}_{test}$	%RMSE$^{P_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
13	5	1801	7.98	7.9231	8.0645	65.8928	7.942	0.77771
13	6	1801	0.11234	0.13316	0.070332	0.95974	0.078713	0.080161
13	7	1801	0.35358	0.44185	0.14012	3.0747	0.15876	0.31599
13	8	1801	0.14306	0.17946	0.053318	1.2319	0.047365	0.13501
14	4	1200	3.0543	3.0171	3.0789	25.166	3.0217	0.44535
14	5	1200	0.17808	0.23261	0.12957	1.5245	0.11519	0.13582
14	6	1200	3.177	3.1626	3.1865	26.298	2.9583	1.1585
14	7	1200	1.0844	1.7109	0.094754	9.3402	0.38975	1.0121
14	8	1200	0.19612	0.27806	0.11214	1.6802	0.107	0.16438
14	4	1501	6.0358	6.0773	5.9939	49.8937	6.0186	0.45463
14	5	1501	3.0994	3.1435	3.0546	25.5519	2.4406	1.9107
14	6	1501	5.564	5.551	5.577	46.1213	4.7338	2.9245
14	7	1501	0.12707	0.15962	0.082525	1.0881	0.077944	0.10038
14	8	1501	6.7051	6.5953	6.8133	55.8471	6.4094	1.9698
14	4	1801	0.11488	0.13506	0.075026	0.97998	0.073882	0.087987
14	5	1801	5.258	5.2637	5.2495	43.6147	3.851	3.5807
14	6	1801	7.2773	7.2458	7.3245	60.2648	7.2292	0.83576
14	7	1801	6.7024	6.6705	6.75	55.4286	6.6879	0.44111
14	8	1801	0.29521	0.33017	0.23312	2.5236	0.22838	0.1871
15	4	1200	0.45821	0.70645	0.13161	3.9451	0.20759	0.40855
15	5	1200	0.14851	0.20881	0.087736	1.2759	0.077464	0.12673
15	6	1200	2.4671	2.0366	2.7163	20.6687	1.3214	2.0837
15	7	1200	0.26738	0.37361	0.16163	2.339	0.15321	0.21917

Table 6.2: Summary of MLP-NARX model construction for identification of compressor pressure

# neurons	# delays	# training samples	RMSE$^{P_C}_{total}$	RMSE$^{P_C}_{training}$	RMSE$^{P_C}_{test}$	%RMSE$^{P_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
15	8	1200	0.16821	0.18365	0.15708	1.4242	0.13848	0.095501
15	4	1501	0.10891	0.12253	0.093327	0.92813	0.073243	0.080623
15	5	1501	1.9863	2.8078	0.068087	17.1378	0.62537	1.8856
15	6	1501	6.6395	6.6501	6.6288	55.0155	5.2845	4.0202
15	7	1501	0.1662	0.19207	0.13546	1.4277	0.095222	0.13624
15	8	1501	0.1144	0.15527	0.045379	0.98157	0.039342	0.10744
15	4	1801	1.6933	2.1853	0.058908	14.4662	0.60032	1.5836
15	5	1801	0.093528	0.10992	0.061178	0.79657	0.049086	0.079625
15	6	1801	0.1709	0.20455	0.10122	1.463	0.10857	0.13201
15	7	1801	1.4032	1.5127	1.2205	11.8839	1.0235	0.95995
15	8	1801	0.14999	0.18796	0.05692	1.2874	0.06694	0.13425

Table 6.3: Summary of MLP-NARX model construction for identification of rotational speed

# neurons	# delays	# training samples	RMSE$^{N}_{total}$	RMSE$^{N}_{training}$	RMSE$^{N}_{test}$	%RMSE$^{N}_{total}$	mean (μ_{ae})	Std (σ_{ae})
10	4	1200	33.0806	36.2859	30.76	0.27904	26.7032	19.5293
10	5	1200	35.5009	37.8877	33.8172	0.29961	28.6901	20.9126
10	6	1200	36.9265	40.1687	34.598	0.31214	29.6183	22.0564
10	7	1200	296.6163	300.5436	293.9705	2.4957	294.5564	34.9023
10	8	1200	35.0706	36.4016	34.1549	0.29608	28.9322	19.8243
10	4	1501	450.7895	447.4808	454.0762	3.7936	449.116	38.8127
10	5	1501	36.5406	40.6434	31.9113	0.30842	29.8534	21.0745
10	6	1501	400.5267	398.5496	402.4953	3.3711	399.6232	26.892
10	7	1501	37.6383	42.1131	32.5504	0.31812	30.6131	21.9006

Table 6.3: Summary of MLP-NARX model construction for identification of rotational speed

# neurons	**# delays**	**# training samples**	**RMSE$^N_{total}$**	**RMSE$^N_{training}$**	**RMSE$^N_{test}$**	**%RMSE$^N_{total}$**	**mean (μ_{ae})**	**Std (σ_{ae})**
10	8	1501	46.1702	50.8953	40.899	0.38999	32.591	32.7088
10	4	1801	33.0561	35.247	29.4638	0.27903	27.0061	19.0656
10	**5**	**1801**	**27.2712**	**32.668**	**16.0695**	**0.23157**	**16.9852**	**21.3395**
10	6	1801	33.9831	35.9672	30.7663	0.28692	28.2217	18.9344
10	7	1801	33.5627	35.5145	30.3992	0.28338	27.7218	18.923
10	8	1801	43.4787	50.0322	31.1554	0.36597	30.8818	30.6108
11	4	1200	29.9508	32.6795	27.9854	0.25259	24.7664	16.8455
11	5	1200	37.26	38.6579	36.2987	0.31462	30.0035	22.0966
11	6	1200	487.0859	489.2236	485.6564	4.0997	486.0575	31.6409
11	7	1200	34.7142	42.2669	28.5952	0.29399	24.9275	24.1639
11	8	1200	32.9792	34.8175	31.6952	0.27836	27.0523	18.8659
11	4	1501	34.5558	38.073	30.6348	0.29169	28.7501	19.1745
11	5	1501	37.1712	41.6389	32.0838	0.31366	30.0774	21.8451
11	6	1501	34.7302	38.3597	30.6712	0.29319	28.6172	19.6817
11	7	1501	36.4697	42.0848	29.8102	0.30877	28.2442	23.0756
11	8	1501	34.4748	38.1447	30.3616	0.291	28.3878	19.5646
11	4	1801	37.7851	40.5383	33.2276	0.3189	30.3886	22.4592
11	5	1801	35.4457	37.5555	32.0193	0.29925	29.2598	20.0099
11	6	1801	28.6433	30.836	24.9939	0.24207	24.2875	15.1865
11	7	1801	38.6031	42.3721	32.1269	0.32549	30.1513	24.11
11	8	1801	35.892	38.3071	31.9263	0.303	28.9653	21.199
12	4	1200	33.3506	34.5145	32.552	0.28155	27.5514	18.7963
12	5	1200	22.7076	24.7451	21.2417	0.19127	19.597	11.4732

Table 6.3: Summary of MLP-NARX model construction for identification of rotational speed

# neurons	# delays	# training samples	RMSE$_{total}^{N}$	RMSE$_{training}^{N}$	RMSE$_{test}^{N}$	%RMSE$_{total}^{N}$	mean (μ_{ae})	Std (σ_{ae})
12	6	1200	38.2502	39.8544	37.1429	0.32296	30.8839	22.5706
12	7	1200	35.4754	38.3312	33.4375	0.29933	28.8403	20.6611
12	8	1200	36.7194	38.6374	35.3838	0.30998	29.8033	21.453
12	4	1501	38.526	43.1089	33.3149	0.32513	31.132	22.6986
12	5	1501	54.5015	69.7232	32.8368	0.45814	35.5807	41.2917
12	6	1501	35.6932	40.9638	29.4908	0.30207	28.2363	21.8374
12	7	1501	37.3729	42.2716	31.7226	0.31531	30.0193	22.2654
12	8	1501	34.9172	38.652	30.7288	0.29474	28.8567	19.6629
12	4	1801	35.1419	37.7757	30.7687	0.29653	28.6724	20.322
12	5	1801	33.468	36.0716	29.1269	0.28292	26.817	20.0271
12	6	1801	32.1788	34.1415	28.9847	0.27168	26.3386	18.4897
12	7	1801	36.1885	38.7392	31.9808	0.30542	29.4998	20.9646
12	8	1801	38.4828	43.0119	30.4461	0.32429	29.0541	25.2388
13	4	1200	30.3273	32.5552	28.7471	0.25624	25.2177	16.8495
13	5	1200	36.9635	40.7537	34.2057	0.31184	29.2038	22.663
13	6	1200	37.9818	39.7273	36.7728	0.32063	31.1684	21.7095
13	7	1200	36.8154	37.8332	36.1214	0.31086	30.3545	20.8357
13	8	1200	535.0246	538.3124	532.8227	4.5031	533.7736	36.5728
13	4	1501	36.7224	40.3748	32.6612	0.31007	30.2088	20.8832
13	5	1501	37.1962	41.1086	32.8176	0.31413	29.4022	22.7868
13	6	1501	50.6195	64.5005	31.0369	0.42555	31.4409	39.6778
13	7	1501	32.8622	37.8066	27.0237	0.27799	26.1423	19.9158
13	8	1501	328.2111	332.4521	323.9117	2.7643	256.0869	205.3194

Table 6.3: Summary of MLP-NARX model construction for identification of rotational speed

# neurons	# delays	# training samples	RMSE$^N_{total}$	RMSE$^N_{training}$	RMSE$^N_{test}$	%RMSE$^N_{total}$	mean (μ_{ae})	Std (σ_{ae})
13	4	1801	35.7897	37.8966	32.3713	0.30219	29.4593	20.3271
13	5	1801	35.4896	37.5681	32.1187	0.29964	29.3142	20.008
13	6	1801	35.2558	37.8341	30.9863	0.29749	29.1605	19.8183
13	7	1801	35.8739	37.9694	32.4761	0.30288	29.8119	19.9579
13	8	1801	35.6625	37.9595	31.9062	0.30103	29.27	20.377
14	4	1200	36.1804	37.6448	35.1709	0.30545	29.5687	20.8533
14	5	1200	49.6096	67.0212	33.2873	0.41704	32.6444	37.362
14	6	1200	38.4341	43.8615	34.3449	0.32408	29.474	24.6711
14	7	1200	36.8356	41.247	33.5761	0.31076	28.5198	23.3166
14	8	1200	36.251	39.1543	34.18	0.30591	29.075	21.655
14	4	1501	37.8188	41.6659	33.5301	0.31932	30.836	21.8988
14	5	1501	38.6098	43.9471	32.4005	0.32567	30.531	23.6381
14	6	1501	37.3452	41.0259	33.2568	0.31536	30.3182	21.8091
14	7	1501	35.2852	39.8261	30.0624	0.29771	28.2574	21.1355
14	8	1501	34.1887	38.061	29.8146	0.28855	27.9037	19.7582
14	4	1801	712.4439	714.7679	708.9416	5.9976	696.3584	150.5617
14	5	1801	59.9799	60.5449	59.1218	0.50693	39.5671	45.0855
14	6	1801	31.0426	33.5547	26.8344	0.26237	25.6354	17.5091
14	7	1801	32.3665	34.4462	28.9665	0.27318	27.0184	17.8242
14	8	1801	37.9365	40.5811	33.5789	0.32019	30.9341	21.9641
15	4	1200	33.4166	34.9402	32.3616	0.2821	27.4471	19.0643
15	5	1200	37.1318	38.0837	36.4838	0.31356	30.0931	21.7562
15	6	1200	713.1613	713.904	712.666	6.0037	712.6353	27.3897

Table 6.3: Summary of MLP-NARX model construction for identification of rotational speed

# neurons	# delays	# training samples	RMSE$^{N}_{total}$	RMSE$^{N}_{training}$	RMSE$^{N}_{test}$	%RMSE$^{N}_{total}$	mean (μ_{ae})	Std (σ_{ae})
15	7	1200	36.0034	39.0281	33.8383	0.30378	28.8403	21.5555
15	8	1200	35.9989	39.9228	33.1274	0.30419	28.7657	21.6474
15	4	1501	37.8108	41.6367	33.5485	0.31928	30.6306	22.1718
15	5	1501	33.6943	37.7805	29.0353	0.28434	27.3686	19.6569
15	6	1501	33.0052	36.1817	29.4862	0.27866	27.3713	18.4465
15	7	1501	40.6191	49.4791	29.1738	0.34338	28.7374	28.7115
15	8	1501	36.169	40.2415	31.5722	0.30564	29.2187	21.3217
15	4	1801	34.3201	36.4397	30.867	0.28979	28.6673	18.8725
15	5	1801	36.4057	38.5832	32.8681	0.30735	30.2351	20.2818
15	6	1801	38.3364	41.0148	33.922	0.32357	31.2112	22.2645
15	7	1801	35.8865	38.54	31.4873	0.30286	29.1371	20.9527
15	8	1801	36.1209	38.2355	32.6914	0.30498	29.4963	20.8526

Table 6.4: Summary of MLP-NARX model construction for identification of turbine temperature

# neurons	# delays	# training samples	RMSE$^{T_T}_{total}$	RMSE$^{T_T}_{training}$	RMSE$^{T_T}_{test}$	%RMSE$^{T_T}_{total}$	mean (μ_{ae})	Std (σ_{ae})
10	4	1200	86.2842	84.782	87.2708	4.7602	78.8147	35.1228
10	5	1200	114.9699	110.2181	118.0298	6.3535	105.4733	45.7618
10	6	1200	86.2803	84.7756	87.2684	4.8021	77.8133	37.2805
10	7	1200	930.9048	920.7743	937.5939	50.5021	774.085	517.1704
10	8	1200	1095.165	1088.7004	1099.4513	58.987	1000.969	444.4248
10	4	1501	577.7018	579.4093	575.9881	31.4064	528.7571	232.7517
10	5	1501	2364.3461	2370.6231	2358.0483	128.8574	2362.5123	93.1191
10	6	1501	229.9371	222.5503	237.0985	13.0265	200.8767	111.9097

Table 6.4: Summary of MLP-NARX model construction for identification of turbine temperature

# neurons	# delays	# training samples	RMSE$^{T_T}_{total}$	RMSE$^{T_T}_{training}$	RMSE$^{T_T}_{test}$	%RMSE$^{T_T}_{total}$	mean (μ_{ae})	Std (σ_{ae})
10	7	1501	1181.8417	1192.916	1170.6551	64.0139	1176.33	114.0256
10	8	1501	130.4287	133.0368	127.7656	7.3695	114.9828	61.578
10	4	1801	70.7158	69.4678	72.5486	3.9678	62.8399	32.438
10	5	1801	656.2278	642.3161	676.5703	35.5456	648.928	97.6245
10	6	1801	233.7606	230.9564	237.9072	13.1615	205.3947	111.6297
10	7	1801	1005.5096	997.4026	1017.5558	54.4685	1000.8644	96.5564
10	8	1801	256.2628	271.5278	231.471	14.6062	215.3638	138.9081
11	4	1200	139.4849	143.23	136.9328	7.869	125.017	61.8714
11	5	1200	1261.2341	1253.7716	1266.1819	69.0058	1113.1521	593.0608
11	6	1200	378.4427	379.7617	377.5612	21.1857	355.8449	128.8362
11	7	1200	429.5365	420.8968	435.1979	23.7517	386.4996	187.4336
11	8	1200	455.3462	469.0084	446.0108	25.2523	443.7013	102.3366
11	4	1501	740.6979	740.5685	740.8273	40.5325	731.7059	115.084
11	5	1501	80.8497	82.3675	79.3018	4.4279	72.4651	35.8597
11	6	1501	131.4283	124.118	138.3574	7.3447	118.1782	57.5189
11	7	1501	214.2128	215.6458	212.7692	12.1863	184.8479	108.2694
11	8	1501	801.0144	815.2247	786.5377	43.3282	661.5941	451.6474
11	4	1801	1495.9311	1487.2264	1508.9012	81.2455	1492.1397	106.4552
11	5	1801	1962.6822	1956.7733	1971.5173	107.3316	1962.3915	33.7822
11	6	1801	182.0535	173.7044	193.9106	10.2063	157.2423	91.7668
11	7	1801	1383.8495	1374.9559	1397.091	75.0955	1379.3474	111.5538
11	8	1801	88.8737	86.5153	92.3001	4.9587	80.6425	37.3602
12	4	1200	417.1126	412.9852	419.8402	22.6221	415.5116	36.5166

Table 6.4: **Summary of MLP-NARX model construction for identification of turbine temperature**

# neurons	# delays	# training samples	RMSE$^{T_T}_{total}$	RMSE$^{T_T}_{training}$	RMSE$^{T_T}_{test}$	%RMSE$^{T_T}_{total}$	mean (μ_{ae})	Std (σ_{ae})
12	5	1200	228.9008	233.9692	225.4605	12.8634	204.535	102.7838
12	6	1200	301.6147	290.2484	308.9559	16.6357	277.4068	118.4125
12	7	1200	273.0264	264.6687	278.4559	15.2701	239.1907	131.6702
12	8	1200	96.9854	93.8887	98.995	5.3556	91.7	31.585
12	4	1501	153.2034	142.4869	163.2243	8.6239	133.9317	74.4004
12	5	1501	90.9468	84.5294	96.9441	5.0702	81.4169	40.5358
12	6	1501	1302.4153	1301.6046	1303.226	71.2834	1290.9371	172.5598
12	7	1501	133.5628	127.9617	138.9417	7.5297	120.6802	57.2398
12	8	1501	890.1844	900.5047	879.7361	48.23	658.0775	599.5683
12	4	1801	1361.443	1351.1735	1376.7119	73.9571	1338.3089	249.9539
12	5	1801	460.9805	454.8895	469.9741	24.6622	449.3605	102.8677
12	6	1801	768.8118	771.8882	764.1713	42.5231	709.318	296.5953
12	7	1801	155.7693	159.994	149.2043	8.8018	129.8104	86.1151
12	8	1801	159.2104	154.2648	166.3573	8.7893	150.7838	51.1184
13	4	1200	222.7684	227.2427	219.7365	12.5666	194.0663	109.3984
13	5	1200	1054.4226	1036.3916	1066.2673	57.1032	993.248	353.9877
13	6	1200	74.6131	73.1871	75.5483	4.1297	67.4487	31.908
13	7	1200	1616.5065	1609.1162	1621.4119	88.0136	1392.8149	820.6
13	8	1200	346.7517	336.1981	353.6087	18.9061	321.841	129.0761
13	4	1501	652.3556	655.5736	649.1194	35.4576	613.8091	220.9581
13	**5**	**1501**	**41.9615**	**40.4995**	**43.3752**	**2.317**	**37.1441**	**19.5246**
13	6	1501	650.2679	653.681	646.8345	34.8309	604.7047	239.1645
13	7	1501	1060.3711	1066.0517	1054.656	58.2532	1048.166	160.4476

Table 6.4: **Summary of MLP-NARX model construction for identification of turbine temperature**

# neurons	# delays	# training samples	RMSE$^{T_T}_{total}$	RMSE$^{T_T}_{training}$	RMSE$^{T_T}_{test}$	%RMSE$^{T_T}_{total}$	mean (μ_{ae})	Std (σ_{ae})
13	8	1501	1048.6294	1058.7777	1038.3751	56.7266	1042.463	113.573
13	4	1801	1218.9509	1213.006	1227.8192	66.3182	983.1694	720.6888
13	5	1801	260.3611	258.2062	263.5621	14.482	241.341	97.7015
13	6	1801	243.6969	227.401	266.2894	13.4599	220.9837	102.7516
13	7	1801	878.8761	874.1466	885.927	48.1715	777.9259	409.0358
13	8	1801	239.4609	243.237	233.6791	13.1819	219.1003	96.6421
14	4	1200	130.0401	123.4004	134.282	7.1211	117.3991	55.9367
14	5	1200	173.9732	168.3939	177.5934	9.6026	158.1745	72.4519
14	6	1200	754.8172	747.7665	759.4788	40.8737	750.8711	77.0947
14	7	1200	911.8461	901.548	918.6436	50.1024	727.9573	549.2194
14	8	1200	247.1411	293.9019	210.2895	14.3173	183.7063	165.3476
14	4	1501	808.2084	818.3531	797.928	43.5568	799.5518	117.9933
14	5	1501	351.15	419.351	265.9369	19.1711	303.198	177.1661
14	6	1501	910.9752	921.3106	900.5142	49.2092	904.0685	111.9824
14	7	1501	228.2557	209.1028	245.9327	12.6991	203.1249	104.1371
14	8	1501	826.0468	816.3956	835.5929	46.2767	573.3683	594.7436
14	4	1801	217.5181	191.1598	251.9547	11.9087	190.1916	105.5698
14	5	1801	115.208	108.5372	124.5511	6.3797	106.3547	44.2969
14	6	1801	803.6739	791.8736	821.066	43.49	620.2961	511.0885
14	7	1801	1892.8008	1884.2143	1905.615	103.0585	1890.6459	90.3081
14	8	1801	786.7352	782.2075	793.4819	42.7907	615.6425	489.9147
15	4	1200	735.0109	725.2283	741.4574	39.7693	592.5796	434.9178
15	5	1200	72.3041	89.5903	57.9933	4.2701	54.71	47.2805

Table 6.4: Summary of MLP-NARX model construction for identification of turbine temperature

# neurons	# delays	# training samples	RMSE$_{total}^{T_T}$	RMSE$_{training}^{T_T}$	RMSE$_{test}^{T_T}$	%RMSE$_{total}^{T_T}$	mean (μ_{ae})	Std (σ_{ae})
15	6	1200	132.7953	125.7867	137.2665	7.3261	123.4316	48.9901
15	7	1200	570.6931	562.4994	576.0877	30.5359	556.9198	124.644
15	8	1200	1056.8093	1049.6544	1061.5499	57.4449	1054.6707	67.211
15	4	1501	69.4952	66.8344	72.0596	3.8014	62.6495	30.0821
15	5	1501	1264.9367	1276.0041	1253.7641	68.5253	1258.453	127.9308
15	6	1501	63.7402	62.3916	65.0617	3.5802	57.0702	28.3914
15	7	1501	2615.1995	2627.8768	2602.452	142.2172	2610.3347	159.4672
15	8	1501	979.3735	981.0944	977.6485	53.3414	778.5604	594.2507
15	4	1801	225.0481	221.0978	230.8501	12.2838	220.775	43.654
15	5	1801	175.4078	169.6901	183.6554	9.5226	160.1606	71.5416
15	6	1801	58.1133	60.3584	54.5705	3.2513	46.0157	35.4982
15	7	1801	1155.4365	1144.4554	1171.7242	62.6479	953.6273	652.5108
15	8	1801	1302.7874	1294.6493	1314.9068	70.9242	1301.2054	64.1943

Table 6.5: Summary of MLP-NARX model construction for identification of turbine pressure

# neurons	# delays	# training samples	RMSE$_{total}^{P_T}$	RMSE$_{training}^{P_T}$	RMSE$_{test}^{P_T}$	%RMSE$_{total}^{P_T}$	Mean (μ_{ae})	Std(σ_{ae})
10	4	1200	0.61736	0.84759	0.39549	16.6077	0.34541	0.51178
10	5	1200	0.74108	0.9734	0.53274	16.2669	0.48182	0.56317
10	6	1200	23.6913	23.5281	23.7995	488.3952	23.6296	1.7098
10	7	1200	2.3141	2.2167	2.3768	48.6664	2.0344	1.103
10	8	1200	0.86353	1.3064	0.32455	19.4462	0.32792	0.79898
10	4	1501	16.231	16.2678	16.194	335.6864	14.3999	7.4904
10	5	1501	0.51204	0.54052	0.48185	10.9378	0.4084	0.3089

Table 6.5: Summary of MLP-NARX model construction for identification of turbine pressure

# neurons	# delays	# training samples	RMSE$^{P_T}_{total}$	RMSE$^{P_T}_{training}$	RMSE$^{P_T}_{test}$	%RMSE$^{P_T}_{total}$	Mean (μ_{ae})	Std(σ_{ae})
10	6	1501	1.6483	1.677	1.6191	36.5713	1.2807	1.0378
10	7	1501	12.7909	12.9199	12.6605	262.9978	9.9073	8.0917
10	8	1501	32.7956	32.8578	32.7332	678.0984	32.7683	1.3393
10	4	1801	0.29078	0.34912	0.16889	6.3238	0.18091	0.22769
10	5	1801	0.41103	0.45438	0.33563	8.8664	0.29668	0.28453
10	6	1801	25.8255	25.6233	26.1261	531.1693	25.6994	2.5502
10	7	1801	14.0682	13.9438	14.253	289.5548	14.0089	1.291
10	8	1801	1.3633	1.7122	0.49871	30.0466	0.79375	1.1086
11	4	1200	1.336	1.4292	1.2702	28.9235	1.066	0.80557
11	5	1200	1.8308	1.8435	1.8222	38.9351	1.5011	1.0482
11	6	1200	14.8961	14.8353	14.9365	307.627	11.9244	8.9291
11	7	1200	0.37046	0.53256	0.19929	8.1656	0.18305	0.32213
11	8	1200	2.4268	2.9471	2.0065	53.3629	1.5214	1.8909
11	4	1501	0.93895	0.9567	0.92084	20.2668	0.74825	0.56732
11	5	1501	0.96676	0.98362	0.94958	22.753	0.77654	0.57594
11	6	1501	0.28118	0.33468	0.21468	7.1195	0.1725	0.22208
11	7	1501	0.6162	0.71211	0.50223	13.3094	0.41847	0.45239
11	8	1501	9.7346	9.9208	9.5446	198.9315	9.5818	1.718
11	4	1801	17.3306	17.2907	17.3904	358.081	12.6387	11.86
11	5	1801	23.606	23.4847	23.7869	487.1657	19.1527	13.8015
11	6	1801	0.91825	1.0085	0.76293	19.8547	0.74643	0.53489
11	7	1801	25.6889	25.5127	25.9511	529.416	25.6146	1.9525
11	8	1801	0.79725	0.965	0.43809	17.5232	0.50798	0.61456

Table 6.5: Summary of MLP-NARX model construction for identification of turbine pressure

# neurons	# delays	# training samples	RMSE$^{P_T}_{total}$	RMSE$^{P_T}_{training}$	RMSE$^{P_T}_{test}$	%RMSE$^{P_T}_{total}$	Mean (μ_{ae})	Std(σ_{ae})
12	4	1200	0.68867	0.90247	0.49761	15.193	0.44739	0.52364
12	5	1200	1.0516	1.2086	0.93251	23.0321	0.76095	0.72601
12	6	1200	0.93991	1.4069	0.39147	20.8212	0.38121	0.85927
12	7	1200	0.62931	0.95104	0.23929	17.1666	0.21893	0.5901
12	8	1200	1.5321	1.7145	1.3974	33.2073	1.1832	0.97346
12	4	1501	6.3269	6.5764	6.067	138.2268	3.6285	5.1839
12	5	1501	0.858	0.98273	0.71164	19.1013	0.67116	0.53461
12	6	1501	0.92128	1.2058	0.49312	20.2576	0.4038	0.82821
12	7	1501	16.0708	16.1816	15.9593	330.8824	16.0099	1.3979
12	8	1501	10.0099	10.1097	9.909	205.8049	9.9552	1.0452
12	4	1801	0.49299	0.44647	0.55554	10.3286	0.4192	0.25949
12	5	1801	0.33884	0.40479	0.20299	8.3306	0.20063	0.2731
12	6	1801	9.4812	9.0519	10.0912	198.5974	7.7466	5.4674
12	7	1801	0.38883	0.46908	0.21877	8.517	0.2126	0.32562
12	**8**	**1801**	**0.50515**	**0.64118**	**0.14539**	**11.2103**	**0.16013**	**0.47917**
13	4	1200	11.9467	11.7735	12.0607	244.2561	11.4902	3.2713
13	5	1200	30.3314	30.205	30.4153	626.7339	30.2972	1.4392
13	6	1200	0.89159	0.96697	0.83761	19.4837	0.73062	0.5111
13	7	1200	0.3794	0.51465	0.25175	8.2715	0.22773	0.3035
13	8	1200	4.3377	6.411	1.9915	95.4077	2.5099	3.5384
13	4	1501	0.87233	1.08	0.59608	22.0664	0.45508	0.74434
13	5	1501	34.6874	34.7931	34.5814	717.0989	34.6576	1.4377
13	6	1501	0.29812	0.34907	0.23639	6.4572	0.20653	0.21502

Table 6.5: **Summary of MLP-NARX model construction for identification of turbine pressure**

# neurons	# delays	# training samples	RMSE$^{P_T}_{total}$	RMSE$^{P_T}_{training}$	RMSE$^{P_T}_{test}$	%RMSE$^{P_T}_{total}$	Mean (μ_{ae})	Std(σ_{ae})
13	7	1501	9.6314	9.8002	9.4594	197.1338	9.506	1.5491
13	8	1501	0.66389	0.80207	0.48789	14.5958	0.39815	0.53133
13	4	1801	26.3962	26.2536	26.6087	544.9101	26.3524	1.5206
13	5	1801	4.0733	4.5273	3.2757	90.9811	2.8369	2.9234
13	6	1801	1.2687	1.3849	1.0709	29.2593	0.9786	0.80752
13	7	1801	3.4399	4.3475	1.1073	76.2508	1.6667	3.0097
13	8	1801	13.908	13.9529	13.8404	288.8155	13.8941	0.62199
14	4	1200	1.2754	1.3011	1.258	27.2448	1.083	0.67374
14	5	1200	0.71459	0.95191	0.49712	17.192	0.42187	0.57687
14	6	1200	1.5257	1.7439	1.361	33.8264	0.73711	1.336
14	7	1200	17.4743	17.5339	17.4345	361.3258	15.4905	8.0881
14	8	1200	6.3131	7.5319	5.349	137.3551	2.9154	5.6005
14	4	1501	1.1009	1.0743	1.1269	23.6802	0.9099	0.61987
14	5	1501	26.9401	27.0032	26.8769	557.362	26.9103	1.2686
14	6	1501	21.7712	21.8919	21.6497	448.5972	17.4501	13.0205
14	7	1501	28.2245	28.3741	28.074	581.8819	28.1509	2.0375
14	8	1501	3.0235	3.9004	1.7508	66.5406	1.3466	2.7075
14	4	1801	21.9488	21.7796	22.2004	452.0385	21.8723	1.8322
14	5	1801	1.5451	1.9074	0.71433	34.0482	0.99151	1.1852
14	6	1801	0.39968	0.49498	0.17828	8.7999	0.22656	0.32932
14	7	1801	1.5624	1.8584	0.95995	34.5175	1.0387	1.1674
14	8	1801	38.1375	37.9635	38.3972	787.4985	38.07	2.2677
15	4	1200	48.9207	48.579	49.1471	1011.6081	48.8172	3.1811

Table 6.5: **Summary of MLP-NARX model construction for identification of turbine pressure**

# neurons	# delays	# training samples	RMSE$_{total}^{P_T}$	RMSE$_{training}^{P_T}$	RMSE$_{test}^{P_T}$	%RMSE$_{total}^{P_T}$	Mean (μ_{ae})	Std(σ_{ae})
15	5	1200	0.68459	0.77571	0.61644	14.7063	0.54185	0.41847
15	6	1200	19.1261	19.0117	19.2019	394.8097	19.0931	1.1227
15	7	1200	21.5211	21.4103	21.5946	444.6303	21.4786	1.3518
15	8	1200	11.3364	11.2663	11.3829	233.8526	11.2221	1.6061
15	4	1501	16.4788	16.8723	16.0754	337.7952	15.4934	5.614
15	5	1501	20.3829	20.4886	20.2766	419.8953	19.4243	6.1784
15	6	1501	0.92138	0.93858	0.90384	19.9011	0.71491	0.58134
15	7	1501	1.5704	1.8169	1.277	34.3909	1.0678	1.1517
15	8	1501	19.2311	19.3101	19.1517	396.3926	16.4121	10.0254
15	4	1801	1.0687	1.2234	0.78097	23.3224	0.70858	0.80016
15	5	1801	36.1745	36.1267	36.2461	748.913	35.5076	6.9153
15	6	1801	15.1982	15.0784	15.3761	312.8291	12.9857	7.8979
15	7	1801	3.2803	3.133	3.4896	69.4147	2.8556	1.6144
15	8	1801	0.78905	0.86334	0.66211	17.0243	0.64414	0.4558

Table 6.6: **Summary of RBF-NARX model construction for identification of compressor temperature**

# neurons	# delays	# training samples	RMSE$_{total}^{T_C}$	RMSE$_{training}^{T_C}$	RMSE$_{test}^{T_C}$	%RMSE$_{total}^{T_C}$	mean (μ_{ae})	Std (σ_{ae})
10	4	1200	37.1059	37.0562	37.139	5.6904	37.0772	1.4608
10	5	1200	39.874	39.8252	39.9065	6.1147	39.8422	1.5937
10	6	1200	12.0564	11.4695	12.432	1.8955	10.2058	6.4196
10	7	1200	12.1397	11.3374	12.646	1.9114	10.2748	6.4663
10	8	1200	13.1253	13.5237	12.8529	2.0499	10.6265	7.7051
10	4	1501	11.4065	12.0461	10.7283	1.7111	9.9734	5.5362

Table 6.6: Summary of RBF-NARX model construction for identification of compressor temperature

# neurons	# delays	# training samples	RMSE$^{T_C}_{total}$	RMSE$^{T_C}_{training}$	RMSE$^{T_C}_{test}$	%RMSE$^{T_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
10	5	1501	14.0235	13.9928	14.054	2.1509	13.9973	0.85648
10	6	1501	11.8029	11.1789	12.3959	1.8569	10.0777	6.1449
10	7	1501	15.7584	15.7401	15.7768	2.4168	15.7445	0.66356
10	8	1501	12.2912	11.6931	12.8619	1.9323	10.4459	6.4785
10	4	1801	63.811	63.7799	63.8576	9.7862	63.7726	2.214
10	5	1801	11.7002	11.4894	12.0097	1.8296	9.4859	6.8503
10	6	1801	11.996	11.2131	13.0834	1.887	10.2332	6.2609
10	7	1801	12.2488	11.5384	13.2437	1.9238	10.3576	6.5396
10	8	1801	12.1271	11.4897	13.0253	1.9028	10.2038	6.5547
11	4	1200	56.9408	56.8806	56.9809	8.7324	56.9053	2.0113
11	5	1200	25.2383	27.8302	23.3522	3.7799	21.6501	12.9731
11	6	1200	11.8853	11.6766	12.0224	1.8638	9.9086	6.5646
11	7	1200	11.6016	11.2514	11.8291	1.8211	9.7301	6.3194
11	8	1200	12.1749	12.1503	12.1912	1.8673	12.1626	0.54588
11	4	1501	37.2969	37.2635	37.3302	5.7197	37.2683	1.4594
11	5	1501	13.399	13.3544	13.4435	2.0551	13.3593	1.0313
11	6	1501	6.6793	6.0077	7.2897	1.0626	4.9412	4.495
11	7	1501	12.0362	11.5564	12.4979	1.8908	10.2024	6.3871
11	8	1501	7.8293	7.8104	7.8483	1.2009	7.815	0.47431
11	4	1801	29.0455	29.0235	29.0784	4.4547	29.0199	1.2184
11	5	1801	12.364	13.0864	11.1927	1.8554	10.87	5.8927
11	6	1801	19.2164	19.1991	19.2424	2.9472	19.1974	0.85332
11	7	1801	29.0256	29.0069	29.0537	4.4516	29.0049	1.098

Table 6.6: Summary of RBF-NARX model construction for identification of compressor temperature

# neurons	# delays	# training samples	RMSE$^{T_C}_{total}$	RMSE$^{T_C}_{training}$	RMSE$^{T_C}_{test}$	%RMSE$^{T_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
11	8	1801	12.4443	11.7598	13.4063	1.9533	10.5	6.6803
12	4	1200	45.7386	45.6973	45.7662	7.014	45.7157	1.4474
12	5	1200	42.7625	42.7259	42.7868	6.5578	42.7415	1.3376
12	6	1200	48.1793	44.6089	50.418	7.4113	41.0336	25.2527
12	7	1200	15.0275	16.1381	14.2394	2.2546	13.2045	7.1752
12	8	1200	12.2575	11.787	12.5612	1.9251	10.3639	6.546
12	4	1501	14.0441	14.9103	13.1202	2.1049	12.1566	7.0335
12	5	1501	79.246	79.1979	79.2941	12.1528	79.2065	2.5028
12	6	1501	6.3564	6.4083	6.3039	0.96243	5.8674	2.4452
12	7	1501	20.0476	20.0226	20.0726	3.0746	20.0277	0.89241
12	8	1501	9.4697	9.8834	9.0367	1.4226	8.373	4.4242
12	4	1801	13.1183	13.1136	13.1253	2.0118	13.112	0.40691
12	5	1801	9.983	10.5289	9.1026	1.4995	8.8604	4.6001
12	6	1801	8.3703	8.8575	7.5806	1.2595	7.416	3.8821
12	7	1801	12.7812	13.5392	11.5508	1.918	11.2396	6.0864
12	8	1801	18.9443	18.9289	18.9674	2.9055	18.9278	0.79126
13	4	1200	43.6067	43.5666	43.6334	6.6873	43.5847	1.385
13	5	1200	47.9875	47.9494	48.0129	7.3591	47.9644	1.4907
13	6	1200	14.2436	14.1938	14.2766	2.1846	14.2147	0.90659
13	7	1200	11.5982	12.2702	11.1279	1.7414	10.2404	5.4462
13	8	1200	24.139	24.1057	24.1611	3.702	24.1213	0.925
13	4	1501	46.4519	46.4246	46.4792	7.1235	46.4289	1.4618
13	5	1501	9.4095	9.7953	9.007	1.4151	8.3934	4.254

Table 6.6: Summary of RBF-NARX model construction for identification of compressor temperature

# neurons	# delays	# training samples	RMSE$^{T_C}_{total}$	RMSE$^{T_C}_{training}$	RMSE$^{T_C}_{test}$	%RMSE$^{T_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
13	6	1501	12.3984	12.368	12.4287	1.9016	12.3726	0.79832
13	7	1501	34.4792	34.4539	34.5045	5.2874	34.455	1.2909
13	8	1501	13.2331	13.2156	13.2506	2.0296	13.2201	0.58747
13	4	1801	23.3831	13.3664	33.1547	3.7213	14.6371	18.2382
13	5	1801	60.4617	60.4332	60.5046	9.2727	60.429	1.9889
13	6	1801	16.0629	17.2239	14.1427	2.4069	13.8547	8.1294
13	7	1801	10.9775	11.6518	9.8795	1.6473	9.6438	5.2452
13	8	1801	21.8927	21.8778	21.915	3.3576	21.8771	0.82585
14	4	1200	45.7869	45.7387	45.819	7.0215	45.7609	1.5415
14	5	1200	67.014	66.9447	67.0601	10.2768	66.9751	2.2809
14	6	1200	11.2758	12.1485	10.6547	1.6916	9.8599	5.4713
14	7	1200	8.9071	9.3756	8.5807	1.3392	7.9384	4.0402
14	8	1200	16.784	18.1357	15.8194	2.5173	14.702	8.098
14	4	1501	79.8436	79.7902	79.8971	12.2445	79.7999	2.6432
14	5	1501	46.0183	45.9928	46.0437	7.0571	45.9959	1.435
14	6	1501	13.3598	13.9538	12.7378	2.005	11.7543	6.3511
14	7	1501	6.5458	6.3632	6.7235	1.0328	4.5504	4.7062
14	8	1501	20.8594	20.8366	20.8822	3.1987	20.8365	0.97668
14	4	1801	58.5788	58.55	58.622	8.9837	58.5435	2.0316
14	5	1801	14.8626	15.8977	13.157	2.2276	12.8685	7.4375
14	6	1801	55.332	55.3061	55.3709	8.4857	55.303	1.792
14	7	1801	75.0613	75.0327	75.1041	11.5113	75.0253	2.325
14	8	1801	17.8698	19.0079	16.0106	2.6806	15.6776	8.5772

Table 6.6: Summary of RBF-NARX model construction for identification of compressor temperature

# neurons	# delays	# training samples	RMSE$^{T_C}_{total}$	RMSE$^{T_C}_{training}$	RMSE$^{T_C}_{test}$	%RMSE$^{T_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
15	4	1200	11.6869	11.6778	11.693	1.7922	11.6813	0.36322
15	5	1200	68.5693	68.5011	68.6146	10.5155	68.5326	2.243
15	6	1200	15.5082	16.9006	14.5065	2.3241	13.3914	7.8229
15	7	1200	12.6432	13.4383	12.0844	1.8977	11.1385	5.9831
15	8	1200	17.5365	17.4983	17.5619	2.6892	17.5116	0.93515
15	4	1501	61.7851	61.7418	61.8285	9.4752	61.748	2.1426
15	5	1501	14.3016	15.1684	13.3781	2.1437	12.3917	7.1412
15	6	1501	10.3437	10.7724	9.8961	1.5535	9.1452	4.8338
15	7	1501	18.489	18.4641	18.5138	2.8356	18.4688	0.8639
15	8	1501	11.1698	11.145	11.1945	1.7132	11.1494	0.67529
15	4	1801	26.4547	26.4409	26.4755	4.0573	26.4412	0.84657
15	5	1801	35.5596	35.5403	35.5887	5.4535	35.5373	1.2608
15	6	1801	48.6086	48.591	48.6351	7.4545	48.5855	1.4992
15	7	1801	9.8646	9.8459	9.8926	1.5131	9.843	0.65294
15	8	1801	13.8776	13.8611	13.9023	2.1285	13.8595	0.70739
16	4	1200	13.7285	13.733	13.7255	2.1037	7.1855	11.6998
16	5	1200	75.0503	74.9818	75.096	11.5093	75.0129	2.371
16	6	1200	14.5399	15.8235	13.6177	2.1796	12.6224	7.2182
16	7	1200	69.4502	69.3915	69.4893	10.6502	69.4167	2.1583
16	8	1200	15.8408	15.8061	15.8638	2.4295	15.822	0.77032
16	4	1501	3.6593	3.6546	3.6641	0.56134	3.6565	0.14423
16	5	1501	78.2691	78.221	78.3172	12.0026	78.2299	2.4789
16	6	1501	13.1059	13.0767	13.1351	2.0101	13.0814	0.80203

Table 6.6: Summary of RBF-NARX model construction for identification of compressor temperature

# neurons	# delays	# training samples	RMSE$^{T_C}_{total}$	RMSE$^{T_C}_{training}$	RMSE$^{T_C}_{test}$	%RMSE$^{T_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
16	7	1501	14.2931	14.2692	14.317	2.1922	14.2735	0.74841
16	8	1501	13.7794	14.3995	13.1296	2.0683	12.1486	6.5037
16	4	1801	33.953	33.932	33.9844	5.2071	33.9296	1.2588
16	5	1801	48.1168	48.0993	48.1431	7.3789	48.094	1.4814
16	6	1801	12.6367	13.4062	11.3846	1.8958	11.0585	6.1162
16	7	1801	64.5851	64.5581	64.6256	9.9049	64.5535	2.02
16	8	1801	11.1573	11.7728	10.1639	1.6756	9.8167	5.3035
17	4	1200	42.091	42.0505	42.118	6.4548	42.0689	1.3665
17	5	1200	53.4853	53.4287	53.523	8.2025	53.4529	1.8609
17	6	1200	64.2304	64.1697	64.2709	9.8502	64.198	2.0416
17	7	1200	21.1758	21.1415	21.1986	3.2476	21.1578	0.87324
17	8	1200	14.1546	15.0486	13.5262	2.1253	12.5055	6.6318
17	4	1501	39.3369	39.3128	39.3611	6.0323	39.3127	1.3806
17	5	1501	61.5826	61.5434	61.6219	9.444	61.5514	1.9606
17	6	1501	57.1463	57.1076	57.1849	8.7636	57.1156	1.8737
17	7	1501	69.0996	69.0603	69.1388	10.5968	69.0662	2.148
17	8	1501	21.6675	22.835	20.4326	3.2508	19.0703	10.2878
17	4	1801	12.541	12.5366	12.5478	1.9232	12.535	0.39016
17	5	1801	18.193	16.2431	20.7788	2.8048	14.8732	10.4789
17	6	1801	17.4117	18.6292	15.4048	2.6095	15.0821	8.7018
17	7	1801	12.0782	12.8119	10.8846	1.8126	10.6159	5.7616
17	8	1801	15.4996	16.3867	14.0637	2.3269	13.6952	7.2592
18	4	1200	11.4663	11.4393	11.4842	1.7583	11.4481	0.64484

Table 6.6: Summary of RBF-NARX model construction for identification of compressor temperature

# neurons	# delays	# training samples	RMSE$^{T_C}_{total}$	RMSE$^{T_C}_{training}$	RMSE$^{T_C}_{test}$	%RMSE$^{T_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
18	5	1200	32.8715	32.8369	32.8946	5.041	32.8541	1.0707
18	6	1200	41.2334	41.2004	41.2554	6.3233	41.2136	1.2776
18	7	1200	36.0543	36.0147	36.0807	5.5291	36.0321	1.2657
18	8	1200	14.3084	14.2748	14.3308	2.1944	14.2905	0.71583
18	4	1501	11.0357	11.0242	11.0473	1.6926	11.0272	0.43441
18	5	1501	78.3479	78.3017	78.3941	12.0152	78.3094	2.4583
18	6	1501	67.9008	67.8587	67.9428	10.4129	67.8668	2.147
18	7	1501	69.37	69.3307	69.4092	10.6382	69.3365	2.1556
18	8	1501	11.0717	11.5356	10.5872	1.6621	9.7577	5.2326
18	4	1801	10.5559	10.552	10.5618	1.6188	10.5506	0.33704
18	5	1801	67.2312	67.2056	67.2697	10.3108	67.1992	2.0747
18	6	1801	42.3215	42.3059	42.345	6.4904	42.3013	1.3083
18	7	1801	9.7916	10.5509	8.5259	1.4716	8.4621	4.927
18	8	1801	43.6666	43.6447	43.6995	6.6966	43.6419	1.4697
19	4	1200	13.5707	13.5599	13.5779	2.0811	13.564	0.42636
19	5	1200	65.2731	65.2112	65.3143	10.0098	65.24	2.0784
19	6	1200	47.0096	46.9726	47.0342	7.2091	46.9872	1.4511
19	7	1200	54.234	54.1801	54.2698	8.317	54.2056	1.7529
19	8	1200	59.3057	59.2522	59.3414	9.0949	59.2767	1.8555
19	4	1501	18.7134	18.7038	18.723	2.8697	18.7043	0.58252
19	5	1501	51.9912	51.9592	52.0232	7.9732	51.9657	1.6281
19	6	1501	52.1396	52.1033	52.1759	7.9959	52.1103	1.7486
19	7	1501	56.2939	56.2569	56.3308	8.6329	56.2648	1.8087

Table 6.6: Summary of RBF-NARX model construction for identification of compressor temperature

# neurons	# delays	# training samples	RMSE$^{T_C}_{total}$	RMSE$^{T_C}_{training}$	RMSE$^{T_C}_{test}$	%RMSE$^{T_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
19	8	1501	11.8008	11.7771	11.8245	1.8099	11.7816	0.67322
19	4	1801	9.5582	9.5518	9.5678	1.466	9.5537	0.29317
19	5	1801	15.4643	15.4549	15.4785	2.3718	15.4556	0.52102
19	6	1801	79.7739	79.7443	79.8182	12.2339	79.736	2.4571
19	7	1801	8.7064	8.6968	8.7207	1.3354	8.6974	0.39373
19	8	1801	19.0332	19.0175	19.0568	2.9191	19.0159	0.81098
20	4	1200	15.672	15.6599	15.68	2.4033	15.6644	0.48657
20	5	1200	53.0984	53.0446	53.1343	8.143	53.07	1.7391
20	6	1200	39.0374	38.9944	39.066	5.9868	39.0129	1.3806
20	7	1200	10.5105	10.4759	10.5335	1.6121	10.4915	0.63185
20	8	1200	50.1478	50.0975	50.1813	7.6907	50.1215	1.6245
20	4	1501	55.8549	55.8188	55.891	8.5656	55.8265	1.7801
20	5	1501	66.7542	66.7114	66.797	10.2372	66.72	2.136
20	6	1501	17.8533	17.8276	17.879	2.7381	17.8324	0.86377
20	7	1501	49.1491	49.1152	49.183	7.5374	49.1223	1.6245
20	8	1501	25.3704	26.9835	23.6463	3.8028	22.0646	12.5245
20	4	1801	10.7061	10.7022	10.7118	1.6419	10.7011	0.32566
20	5	1801	46.0865	46.0689	46.1129	7.0678	46.0641	1.4376
20	6	1801	44.1258	44.1045	44.1578	6.7672	44.1027	1.4268
20	7	1801	70.2318	70.2052	70.2717	10.7706	70.1984	2.1666
20	8	1801	38.5068	38.4868	38.5368	5.9055	38.4833	1.346
10	4	1200	1.8188	1.9454	1.7292	0.27661	1.5266	0.98879
10	5	1200	2.1915	2.7306	1.7421	0.33391	1.5908	1.5077

Table 6.6: Summary of RBF-NARX model construction for identification of compressor temperature

# neurons	# delays	# training samples	RMSE$^{T_C}_{total}$	RMSE$^{T_C}_{training}$	RMSE$^{T_C}_{test}$	%RMSE$^{T_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
10	6	1200	0.95653	1.1003	0.84731	0.14544	0.72307	0.6263
10	7	1200	2.2127	2.5475	1.9581	0.33685	1.7548	1.3482
10	8	1200	2.4748	2.8173	2.2174	0.37669	2.001	1.4565
10	4	1501	2.1221	2.1281	2.116	0.32288	1.7637	1.1803
10	5	1501	2.7501	2.9006	2.5908	0.41872	2.1888	1.6654
10	6	1501	1.6867	1.7608	1.6091	0.25666	1.3559	1.0034
10	7	1501	2.8032	2.8708	2.7339	0.42667	2.2943	1.6109
10	8	1501	1.5578	1.6449	1.4655	0.23702	1.2465	0.93455
10	4	1801	1.6739	1.8018	1.4611	0.25456	1.4078	0.90572
10	5	1801	2.7066	3.0159	2.1608	0.4121	2.1352	1.6635
10	6	1801	1.2448	1.3573	1.0536	0.18931	1.0106	0.72697
10	**7**	**1801**	**0.56069**	**0.60188**	**0.49245**	**0.085012**	**0.44584**	**0.34007**
10	8	1801	0.76254	0.82931	0.64958	0.11582	0.60207	0.46803
11	4	1200	45.4889	45.4371	45.5234	6.8836	45.4588	1.6552
11	5	1200	3.0822	3.5464	2.7294	0.46925	2.4649	1.8507
11	6	1200	1.4912	1.7501	1.2901	0.22692	1.1631	0.93331
11	7	1200	1.6108	1.8436	1.4348	0.24512	1.2955	0.95744
11	8	1200	1.2776	1.4696	1.1317	0.19436	1.0121	0.77985
11	4	1501	54.1284	54.0921	54.1647	8.191	54.095	1.9
11	5	1501	3.1071	3.2399	2.9683	0.47296	2.5174	1.8215
11	6	1501	1.7994	1.9042	1.688	0.27388	1.4164	1.1099
11	7	1501	1.4374	1.5362	1.3312	0.21875	1.1301	0.88849
11	8	1501	2.0893	2.2507	1.9142	0.3181	1.6294	1.3079

Table 6.6: Summary of RBF-NARX model construction for identification of compressor temperature

# neurons	# delays	# training samples	RMSE$^{T_C}_{total}$	RMSE$^{T_C}_{training}$	RMSE$^{T_C}_{test}$	%RMSE$^{T_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
11	4	1801	2.1869	2.4166	1.7875	0.3329	1.727	1.3419
11	5	1801	2.3996	2.6633	1.9376	0.36529	1.921	1.4382
11	6	1801	1.6637	1.8499	1.3364	0.25323	1.2925	1.0477
11	7	1801	1.4928	1.6711	1.1756	0.22725	1.1435	0.95983
11	8	1801	1.4931	1.6556	1.2091	0.22719	1.2009	0.88751
12	4	1200	31.8172	31.7683	31.8497	4.8147	31.7869	1.3891
12	5	1200	2.5468	2.8568	2.3172	0.38768	2.0666	1.4885
12	6	1200	1.5706	1.6786	1.4942	0.23885	1.3162	0.85706
12	7	1200	1.0923	1.199	1.015	0.16608	0.87641	0.65199
12	8	1200	1.6724	1.9087	1.4944	0.25453	1.3335	1.0095
12	4	1501	36.8247	36.7942	36.8552	5.5725	36.7976	1.4117
12	5	1501	2.9844	3.0319	2.9361	0.45415	2.4711	1.6737
12	6	1501	2.2253	2.2574	2.1927	0.3386	1.8362	1.2572
12	7	1501	1.5064	1.5509	1.4605	0.22918	1.2232	0.87932
12	8	1501	1.75	1.8437	1.651	0.26632	1.3974	1.0538
12	4	1801	31.1707	31.1519	31.1988	4.7168	31.1452	1.2608
12	5	1801	2.0925	2.3587	1.6125	0.31866	1.6032	1.3449
12	6	1801	2.3492	2.605	1.902	0.3576	1.8923	1.3924
12	7	1801	2.4381	2.7469	1.8818	0.37129	1.8876	1.5435
12	8	1801	1.2231	1.348	1.0068	0.18606	0.97606	0.73713
13	4	1200	1.734	1.8262	1.6697	0.26369	1.4618	0.93282
13	5	1200	1.9642	2.3439	1.6638	0.29915	1.5049	1.2625
13	6	1200	1.3912	1.4892	1.3219	0.21152	1.1653	0.7601

Table 6.6: Summary of RBF-NARX model construction for identification of compressor temperature

# neurons	# delays	# training samples	RMSE$^{T_C}_{total}$	RMSE$^{T_C}_{training}$	RMSE$^{T_C}_{test}$	%RMSE$^{T_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
13	7	1200	1.824	2.1602	1.5603	0.27771	1.4136	1.1529
13	8	1200	2.2697	2.7246	1.9074	0.34567	1.7255	1.4748
13	4	1501	39.2468	39.2152	39.2784	5.9388	39.2183	1.4956
13	5	1501	1.8953	1.8459	1.9436	0.28825	1.602	1.013
13	6	1501	4.2071	4.4422	3.9579	0.64051	3.378	2.5082
13	7	1501	2.9497	3.3332	2.5079	0.44947	2.1652	2.0035
13	8	1501	1.7758	1.8509	1.6974	0.27024	1.4384	1.0417
13	4	1801	2.9932	3.5804	1.7792	0.45676	1.8668	2.3401
13	5	1801	2.2647	2.4674	1.9208	0.34457	1.8741	1.2717
13	6	1801	2.7406	2.9344	2.4209	0.41684	2.3275	1.4473
13	7	1801	2.0785	2.2825	1.7278	0.31628	1.7023	1.1928
13	8	1801	2.2654	2.5389	1.7778	0.34495	1.765	1.4204
14	4	1200	33.2243	33.1831	33.2518	5.0276	33.1984	1.3138
14	5	1200	18.6368	18.6	18.6613	2.8202	18.6149	0.90325
14	6	1200	2.2654	2.5689	2.0382	0.34471	1.8505	1.3069
14	7	1200	2.3937	2.7497	2.1236	0.36441	1.9126	1.4395
14	8	1200	3.2324	3.6559	2.9163	0.49206	2.6246	1.8872
14	4	1501	48.8653	48.8322	48.8985	7.3946	48.8343	1.7413
14	5	1501	19.4212	19.3954	19.447	2.9388	19.3969	0.97207
14	6	1501	2.2395	2.2748	2.2036	0.34072	1.854	1.2564
14	7	1501	2.1311	2.3581	1.8766	0.32459	1.6207	1.3841
14	8	1501	2.8692	3.2333	2.4514	0.43712	2.1245	1.9287
14	4	1801	6.0451	6.0353	6.0596	0.91473	6.0326	0.38871

Table 6.6: **Summary of RBF-NARX model construction for identification of compressor temperature**

# neurons	# delays	# training samples	RMSE$^{T_C}_{total}$	RMSE$^{T_C}_{training}$	RMSE$^{T_C}_{test}$	%RMSE$^{T_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
14	5	1801	3.1537	3.8191	1.727	0.48133	1.8656	2.5431
14	6	1801	1.8494	1.9689	1.6539	0.28119	1.5682	0.98046
14	7	1801	1.2996	1.4351	1.0643	0.19772	1.0333	0.78824
14	8	1801	1.5496	1.7265	1.2374	0.23581	1.2356	0.93538
15	4	1200	18.9485	18.8963	18.9832	2.8673	18.9164	1.1019
15	5	1200	21.2376	21.2017	21.2615	3.2137	21.216	0.95905
15	6	1200	3.2826	4.1553	2.5398	0.50036	2.3466	2.2958
15	7	1200	2.0564	2.6026	1.5917	0.31343	1.4511	1.4574
15	8	1200	1.7945	2.0812	1.5748	0.27316	1.422	1.0948
15	4	1501	1.7864	1.8355	1.736	0.2718	1.4534	1.0389
15	5	1501	26.0661	26.0411	26.0911	3.9444	26.0444	1.0626
15	6	1501	1.3764	1.4353	1.3149	0.20945	1.0882	0.84293
15	7	1501	2.4402	2.5931	2.2769	0.37148	1.9353	1.4865
15	8	1501	1.9282	2.1107	1.7264	0.29365	1.4635	1.2557
15	4	1801	30.871	30.8435	30.9122	4.6715	30.8314	1.5638
15	5	1801	2.3518	2.5143	2.0844	0.35771	1.9896	1.2543
15	6	1801	4.2938	5.2265	2.2605	0.65558	2.5007	3.491
15	7	1801	2.7786	3.0661	2.2801	0.42293	2.2594	1.6177
15	8	1801	1.7323	1.945	1.3515	0.26372	1.3381	1.1003
16	4	1200	2.1754	2.4794	1.9466	0.33101	1.7633	1.2742
16	5	1200	18.0577	18.0157	18.0856	2.7325	18.0321	0.96152
16	6	1200	11.7985	11.7584	11.8251	1.7853	11.7737	0.76453
16	7	1200	1.8597	2.0211	1.7439	0.28278	1.5705	0.99617

Table 6.6: Summary of RBF-NARX model construction for identification of compressor temperature

# neurons	# delays	# training samples	RMSE$^{T_C}_{total}$	RMSE$^{T_C}_{training}$	RMSE$^{T_C}_{test}$	%RMSE$^{T_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
16	8	1200	2.1562	2.3735	1.9983	0.32806	1.7965	1.1926
16	4	1501	15.8265	15.8058	15.8472	2.3949	15.8062	0.80131
16	5	1501	3.9168	4.8465	2.6812	0.59771	2.4167	3.0829
16	6	1501	2.7457	2.9554	2.5185	0.41813	2.1451	1.7142
16	7	1501	3.604	3.896	3.2861	0.54882	2.8319	2.2296
16	8	1501	1.7144	1.9144	1.4877	0.26108	1.2854	1.1347
16	4	1801	22.61	22.5835	22.6496	3.4214	22.5735	1.2844
16	5	1801	15.1746	15.1563	15.202	2.2962	15.1486	0.88899
16	6	1801	36.7634	36.747	36.788	5.5632	36.7401	1.3084
16	7	1801	2.4541	2.6683	2.0918	0.37337	2.0319	1.3764
16	8	1801	2.2422	2.5104	1.7647	0.34137	1.7684	1.3787
17	4	1200	71.9109	71.8401	71.958	10.8819	71.868	2.4826
17	5	1200	10.7752	10.7387	10.7994	1.6305	10.7532	0.68717
17	6	1200	3.9621	5.3719	2.6327	0.60448	2.5027	3.0722
17	7	1200	21.353	21.3231	21.3729	3.2312	21.3354	0.86683
17	8	1200	1.8598	2.0995	1.6813	0.28298	1.5119	1.0833
17	4	1501	40.674	40.6467	40.7012	6.1549	40.648	1.4517
17	5	1501	13.2463	13.22	13.2725	2.0044	13.2217	0.80749
17	6	1501	35.8034	35.7777	35.8292	5.4178	35.7803	1.287
17	7	1501	2.0474	2.2249	1.8528	0.31173	1.5885	1.2919
17	8	1501	2.5649	2.6771	2.4475	0.39036	2.0774	1.5047
17	4	1801	2.943	3.4926	1.8313	0.44902	1.8595	2.2815
17	5	1801	2.8404	3.3139	1.9222	0.43307	1.9016	2.1102

Table 6.6: Summary of RBF-NARX model construction for identification of compressor temperature

# neurons	# delays	# training samples	RMSE$^{T_C}_{total}$	RMSE$^{T_C}_{training}$	RMSE$^{T_C}_{test}$	%RMSE$^{T_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
17	6	1801	9.941	9.9167	9.9774	1.5042	9.9064	0.82979
17	7	1801	2.305	2.6162	1.7361	0.35108	1.7495	1.501
17	8	1801	2.3534	2.5865	1.9519	0.35815	1.9142	1.3692
18	4	1200	67.0134	66.9513	67.0547	10.1407	66.9778	2.183
18	5	1200	22.0574	22.0224	22.0808	3.3378	22.0365	0.96074
18	6	1200	1.4406	1.5852	1.3356	0.21912	1.1819	0.82385
18	7	1200	3.7845	4.5955	3.1296	0.57653	2.8723	2.4646
18	8	1200	10.6259	10.5965	10.6455	1.6079	10.6079	0.61859
18	4	1501	28.8231	28.7996	28.8466	4.3616	28.801	1.1283
18	5	1501	20.8712	20.8465	20.8959	3.1582	20.8486	0.9717
18	6	1501	16.1921	16.1714	16.2127	2.4502	16.174	0.76541
18	7	1501	51.1541	51.1223	51.186	7.7409	51.1275	1.6502
18	8	1501	2.2437	2.3539	2.1277	0.34145	1.8061	1.3315
18	4	1801	57.8469	57.8173	57.8914	8.7537	57.8082	2.1158
18	5	1801	14.2412	14.2228	14.2687	2.155	14.215	0.86248
18	6	1801	3.0729	3.8268	1.2792	0.46953	1.3844	2.7439
18	7	1801	6.139	7.617	2.678	0.93797	3.1563	5.2663
18	8	1801	2.3159	2.6464	1.7035	0.35277	1.7307	1.5392
19	4	1200	6.7305	6.71	6.7441	1.0184	6.7186	0.40023
19	5	1200	3.9568	4.7748	3.3011	0.60287	2.9916	2.5902
19	6	1200	32.5637	32.5258	32.589	4.9277	32.5416	1.1995
19	7	1200	2.2081	2.5027	1.9876	0.33608	1.7755	1.3128
19	8	1200	2.5378	2.6721	2.4441	0.38594	2.1649	1.3244

Table 6.6: Summary of RBF-NARX model construction for identification of compressor temperature

# neurons	# delays	# training samples	RMSE$^{T_C}_{total}$	RMSE$^{T_C}_{training}$	RMSE$^{T_C}_{test}$	%RMSE$^{T_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
19	4	1501	15.5102	15.4996	15.5207	2.3471	15.4947	0.69243
19	5	1501	36.3256	36.2687	36.3825	5.4967	36.2721	1.9713
19	6	1501	2.248	2.2595	2.2364	0.34198	1.8758	1.2392
19	7	1501	3.2822	3.4332	3.1238	0.49962	2.6627	1.9194
19	8	1501	2.9195	3.1035	2.723	0.44441	2.3401	1.746
19	4	1801	17.8715	17.8598	17.8892	2.7044	17.8513	0.84988
19	5	1801	28.1041	28.0855	28.1319	4.2527	28.0775	1.222
19	6	1801	2.1404	2.3484	1.7833	0.32573	1.7603	1.2178
19	7	1801	1.264	1.3644	1.0961	0.19216	1.0465	0.70905
19	8	1801	2.0942	2.3248	1.69	0.31873	1.6732	1.2596
20	4	1200	3.3749	3.9728	2.9091	0.51418	2.5874	2.1672
20	5	1200	2.3095	2.8049	1.9094	0.35182	1.7325	1.5275
20	6	1200	2.3741	3.1574	1.6583	0.36206	1.5538	1.7953
20	7	1200	2.9552	3.4639	2.5607	0.44983	2.3282	1.8203
20	8	1200	14.3277	14.2983	14.3473	2.1681	14.3102	0.70793
20	4	1501	10.2453	10.2357	10.2549	1.5503	10.2366	0.42231
20	5	1501	49.5614	49.5368	49.586	7.5	49.5377	1.5332
20	6	1501	22.4266	22.4046	22.4486	3.3937	22.4076	0.92383
20	7	1501	12.6035	12.5793	12.6277	1.9071	12.5812	0.75018
20	8	1501	13.7666	13.749	13.7842	2.0832	13.7506	0.66428
20	4	1801	5.8958	5.8939	5.8986	0.8922	5.8921	0.20937
20	5	1801	24.1078	24.0954	24.1265	3.6481	24.0912	0.89588
20	6	1801	1.5806	1.6944	1.3924	0.24032	1.3323	0.85053

Table 6.6: Summary of RBF-NARX model construction for identification of compressor temperature

# neurons	**# delays**	**# training samples**	**RMSE**$^{T_C}_{total}$	**RMSE**$^{T_C}_{training}$	**RMSE**$^{T_C}_{test}$	**%RMSE**$^{T_C}_{total}$	**mean** (μ_{ae})	**Std** (σ_{ae})
20	7	1801	2.3228	2.484	2.0573	0.35322	1.9812	1.2128
20	8	1801	2.1765	2.4505	1.6835	0.33138	1.6635	1.4038

Table 6.7: Summary of RBF-NARX model construction for identification of compressor pressure

# neurons	**# delays**	**# training samples**	**RMSE**$^{P_C}_{total}$	**RMSE**$^{P_C}_{training}$	**RMSE**$^{P_C}_{test}$	**%RMSE**$^{P_C}_{total}$	**mean** (μ_{ae})	**Std** (σ_{ae})
10	4	1200	0.27729	0.27057	0.28167	2.6835	0.18022	0.21077
10	5	1200	0.2495	0.22431	0.26496	2.4478	0.17078	0.18193
10	6	1200	0.36428	0.32529	0.38809	3.5942	0.25576	0.25944
10	7	1200	0.49642	0.45142	0.52425	4.8411	0.3804	0.31899
10	8	1200	0.49292	0.44786	0.52079	4.804	0.37688	0.31776
10	4	1501	0.26541	0.22748	0.29858	2.6596	0.16943	0.20432
10	5	1501	0.23573	0.20105	0.26595	2.3341	0.16305	0.17028
10	6	1501	0.39076	0.35537	0.42322	3.8445	0.28424	0.26819
10	7	1501	0.45941	0.4237	0.49255	4.4883	0.34727	0.30081
10	8	1501	0.44724	0.41178	0.48011	4.3721	0.33566	0.29561
10	4	1801	1.1516	1.1506	1.1531	10.0516	1.1505	0.050379
10	5	1801	0.24287	0.22155	0.27175	2.3507	0.16949	0.17398
10	6	1801	0.21905	0.19276	0.25344	2.117	0.17276	0.13469
10	7	1801	0.45406	0.41265	0.50993	4.4376	0.34153	0.29926
10	8	1801	0.46189	0.41917	0.51945	4.5183	0.34831	0.3034
11	4	1200	3.3114	3.2965	3.3213	28.9083	3.3024	0.24376
11	5	1200	0.43436	0.38927	0.46197	4.2643	0.32616	0.28691

Table 6.7: Summary of RBF-NARX model construction for identification of compressor pressure

# neurons	# delays	# training samples	RMSE$^{P_C}_{total}$	RMSE$^{P_C}_{training}$	RMSE$^{P_C}_{test}$	%RMSE$^{P_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
11	6	1200	0.25316	0.22564	0.26995	2.5006	0.16227	0.19436
11	7	1200	0.31217	0.27981	0.33199	3.087	0.20591	0.23467
11	8	1200	0.44639	0.40343	0.47286	4.3703	0.33323	0.29708
11	4	1501	0.25549	0.22948	0.2791	2.5073	0.16807	0.19246
11	5	1501	0.26464	0.24162	0.28583	2.585	0.1746	0.19891
11	6	1501	0.2285	0.19818	0.25526	2.2368	0.16076	0.1624
11	7	1501	0.38733	0.36051	0.41242	3.7817	0.27443	0.27338
11	8	1501	0.50747	0.47154	0.54104	4.9383	0.3905	0.32414
11	4	1801	0.28079	0.24998	0.32153	2.7877	0.17914	0.21625
11	5	1801	0.24886	0.22088	0.28577	2.4524	0.16769	0.18392
11	6	1801	0.20601	0.1795	0.24038	1.9433	0.175	0.10872
11	7	1801	0.61769	0.56468	0.68965	5.9957	0.48668	0.38043
11	8	1801	0.2727	0.2513	0.30199	2.6516	0.17377	0.21021
12	4	1200	0.32781	0.3111	0.33849	3.208	0.20895	0.25262
12	5	1200	0.23131	0.19446	0.2529	2.2988	0.15838	0.16862
12	6	1200	0.27221	0.24732	0.28761	2.6824	0.17416	0.20924
12	7	1200	0.27992	0.26796	0.28762	2.7224	0.17539	0.2182
12	8	1200	0.72836	0.66784	0.76604	7.0486	0.5781	0.44314
12	4	1501	0.40207	0.40601	0.39809	3.8368	0.25615	0.30997
12	5	1501	0.18382	0.15026	0.21215	1.7631	0.14662	0.11088
12	6	1501	0.41938	0.38413	0.4519	4.1141	0.31007	0.28242
12	7	1501	0.24373	0.21003	0.27332	2.4229	0.16106	0.18296
12	8	1501	0.28221	0.25853	0.30407	2.763	0.17764	0.21933

Table 6.7: Summary of RBF-NARX model construction for identification of compressor pressure

# neurons	# delays	# training samples	RMSE$^{P_C}_{total}$	RMSE$^{P_C}_{training}$	RMSE$^{P_C}_{test}$	%RMSE$^{P_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
12	4	1801	3.1086	3.1048	3.1143	27.1382	3.1042	0.16505
12	5	1801	1.4399	1.4388	1.4415	12.5672	1.4386	0.059397
12	6	1801	0.46907	0.42517	0.52816	4.59	0.35626	0.30519
12	7	1801	0.28504	0.25415	0.32595	2.8285	0.18177	0.2196
12	8	1801	0.21429	0.19242	0.24346	2.0043	0.17885	0.11806
13	4	1200	3.2028	3.1951	3.208	27.9563	3.1986	0.16441
13	5	1200	0.21169	0.17245	0.23421	2.1133	0.14349	0.15567
13	6	1200	2.3936	2.3854	2.399	20.8964	2.389	0.14737
13	7	1200	0.20783	0.17069	0.22927	1.9472	0.17839	0.10665
13	8	1200	0.22611	0.20788	0.23748	2.1465	0.1745	0.14382
13	4	1501	0.22129	0.18274	0.25407	2.1845	0.16128	0.15155
13	5	1501	0.25489	0.22415	0.28231	2.5192	0.167	0.19259
13	6	1501	0.28789	0.28713	0.28865	2.7343	0.18022	0.22454
13	7	1501	0.22635	0.19835	0.25127	2.2	0.16481	0.15518
13	8	1501	0.53687	0.49701	0.574	5.2298	0.41463	0.3411
13	4	1801	3.4083	3.405	3.4133	29.753	3.4046	0.15905
13	5	1801	0.22288	0.19118	0.2634	2.2253	0.15171	0.1633
13	6	1801	0.20361	0.17408	0.24124	1.9881	0.15853	0.1278
13	7	1801	0.22131	0.19672	0.25378	2.0797	0.1865	0.11917
13	8	1801	0.21898	0.19559	0.25001	2.0826	0.17688	0.12912
14	4	1200	0.39179	0.4219	0.37038	3.712	0.23663	0.31232
14	5	1200	0.20587	0.17015	0.22656	1.8846	0.18115	0.097821
14	6	1200	0.23809	0.19789	0.26147	2.3855	0.15235	0.183

Table 6.7: Summary of RBF-NARX model construction for identification of compressor pressure

# neurons	# delays	# training samples	RMSE$^{P_C}_{total}$	RMSE$^{P_C}_{training}$	RMSE$^{P_C}_{test}$	%RMSE$^{P_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
14	7	1200	0.2073	0.16992	0.22884	2.0362	0.15616	0.13636
14	8	1200	0.22481	0.19147	0.24452	1.9687	0.20179	0.099123
14	4	1501	0.35318	0.32374	0.38036	3.4851	0.22438	0.27279
14	5	1501	0.41109	0.43278	0.38817	3.8563	0.248	0.32792
14	6	1501	0.27294	0.26606	0.27966	2.6113	0.17579	0.20883
14	7	1501	0.27604	0.25067	0.29928	2.3314	0.23951	0.13726
14	8	1501	0.23	0.2021	0.25488	2.2183	0.17316	0.1514
14	4	1801	3.429	3.4254	3.4344	29.9306	3.4249	0.16716
14	5	1801	2.8881	2.8849	2.8928	25.2102	2.8847	0.1407
14	6	1801	2.3418	2.3378	2.3477	20.4487	2.3372	0.14631
14	7	1801	0.30941	0.29792	0.32588	2.5975	0.25967	0.16826
14	8	1801	0.2217	0.2019	0.24849	2.0147	0.19213	0.11065
15	4	1200	0.43115	0.46429	0.40758	4.0764	0.27641	0.33095
15	5	1200	0.27279	0.26985	0.27473	2.6239	0.17986	0.20513
15	6	1200	0.29513	0.3171	0.27953	2.7865	0.18358	0.23112
15	7	1200	0.2077	0.1777	0.22548	1.8368	0.18385	0.096639
15	8	1200	0.23644	0.22366	0.24457	2.1341	0.19811	0.12907
15	4	1501	0.32802	0.31222	0.3431	3.1905	0.20596	0.25534
15	5	1501	0.25707	0.23293	0.27914	2.5073	0.17132	0.1917
15	6	1501	0.20471	0.16697	0.23652	1.9701	0.16456	0.12178
15	7	1501	0.25635	0.24594	0.26635	2.4437	0.17342	0.18882
15	8	1501	0.33477	0.30639	0.36093	2.8124	0.28841	0.16999
15	4	1801	3.3353	3.3318	3.3406	29.1183	3.3315	0.16064

Table 6.7: Summary of RBF-NARX model construction for identification of compressor pressure

# neurons	# delays	# training samples	RMSE$_{total}^{P_C}$	RMSE$_{training}^{P_C}$	RMSE$_{test}^{P_C}$	%RMSE$_{total}^{P_C}$	mean (μ_{ae})	Std (σ_{ae})
15	5	1801	0.24709	0.22678	0.27477	2.3809	0.17826	0.17114
15	6	1801	1.4641	1.4631	1.4656	12.7794	1.463	0.056395
15	7	1801	1.935	1.9298	1.9429	16.9009	1.9285	0.15864
15	8	1801	0.26447	0.24586	0.29018	2.25	0.23386	0.12353
16	4	1200	0.28487	0.24611	0.30801	2.8389	0.18627	0.21557
16	5	1200	0.35642	0.34882	0.3614	3.4574	0.23157	0.271
16	6	1200	2.3845	2.3774	2.3892	20.8182	2.3807	0.13551
16	7	1200	0.23444	0.21631	0.24577	2.2075	0.18785	0.14028
16	8	1200	0.23581	0.22585	0.24222	2.1883	0.18775	0.1427
16	4	1501	3.5263	3.5236	3.529	30.7675	3.5237	0.1345
16	5	1501	0.41183	0.40234	0.42112	3.9623	0.27065	0.31046
16	6	1501	3.3109	3.3064	3.3154	28.9006	3.3073	0.15503
16	7	1501	0.27366	0.24441	0.30009	2.3227	0.23972	0.13202
16	8	1501	0.22316	0.18936	0.2525	1.968	0.19989	0.099242
16	4	1801	0.31182	0.2779	0.35673	3.0941	0.20236	0.23728
16	5	1801	0.30919	0.2848	0.34255	3.0313	0.19647	0.23878
16	6	1801	2.6603	2.6574	2.6646	23.2261	2.6573	0.12607
16	7	1801	0.21901	0.1956	0.25006	2.0507	0.18329	0.1199
16	8	1801	0.32885	0.34869	0.29657	2.9872	0.19262	0.26657
17	4	1200	0.30438	0.27417	0.32294	2.9991	0.20601	0.2241
17	5	1200	0.43951	0.49983	0.39423	4.0914	0.26638	0.34965
17	6	1200	1.5079	1.5025	1.5115	13.1665	1.5051	0.092196
17	7	1200	0.20107	0.16349	0.22262	1.9271	0.16391	0.11649

Table 6.7: Summary of RBF-NARX model construction for identification of compressor pressure

# neurons	# delays	# training samples	RMSE$^{P_C}_{total}$	RMSE$^{P_C}_{training}$	RMSE$^{P_C}_{test}$	%RMSE$^{P_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
17	8	1200	0.38303	0.3519	0.40245	3.1942	0.32889	0.19637
17	4	1501	0.35668	0.32375	0.38683	3.5287	0.22894	0.27355
17	5	1501	1.4298	1.4249	1.4347	12.4795	1.4249	0.11813
17	6	1501	1.4487	1.4449	1.4525	12.6508	1.4457	0.093066
17	7	1501	0.2465	0.21187	0.27686	2.1676	0.22207	0.10702
17	8	1501	0.30593	0.32726	0.28298	2.8178	0.18793	0.24144
17	4	1801	4.8255	4.8231	4.829	42.1144	4.8224	0.17382
17	5	1801	0.2955	0.26973	0.33042	2.9032	0.18505	0.23042
17	6	1801	1.522	1.5199	1.5253	13.2906	1.52	0.078676
17	7	1801	0.28553	0.27501	0.30064	2.7051	0.18777	0.21514
17	8	1801	0.33944	0.32125	0.36505	2.8409	0.29538	0.16727
18	4	1200	0.50234	0.49635	0.50628	4.8522	0.33745	0.37218
18	5	1200	2.5842	2.578	2.5884	22.5589	2.5809	0.13035
18	6	1200	0.31453	0.29285	0.32819	3.0854	0.19871	0.24386
18	7	1200	0.26602	0.2386	0.28283	2.3636	0.23415	0.12629
18	8	1200	0.20878	0.17355	0.22927	1.9111	0.18352	0.099555
18	4	1501	2.8501	2.849	2.8513	24.8739	2.8481	0.10885
18	5	1501	0.34186	0.3226	0.3601	3.3328	0.21996	0.26174
18	6	1501	0.32467	0.33594	0.31298	3.0653	0.1933	0.2609
18	7	1501	0.28165	0.27079	0.29211	2.6996	0.1805	0.21625
18	8	1501	1.5804	1.5765	1.5842	13.8006	1.5774	0.097441
18	4	1801	4.9155	4.9126	4.9198	42.8959	4.9115	0.19783
18	5	1801	2.061	2.0598	2.0627	17.984	2.0592	0.085708

Table 6.7: **Summary of RBF-NARX model construction for identification of compressor pressure**

# neurons	# delays	# training samples	RMSE$^{P_C}_{total}$	RMSE$^{P_C}_{training}$	RMSE$^{P_C}_{test}$	%RMSE$^{P_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
18	6	1801	1.6421	1.6391	1.6465	14.341	1.6389	0.10142
18	7	1801	1.0644	1.0635	1.0657	9.2906	1.0635	0.044132
18	8	1801	0.2436	0.22755	0.26589	2.2733	0.19039	0.152
19	4	1200	6.5112	6.5031	6.5166	56.8194	6.5065	0.24699
19	5	1200	1.7997	1.797	1.8016	15.7024	1.798	0.080089
19	6	1200	0.31955	0.39161	0.26071	2.8548	0.20116	0.24833
19	7	1200	0.33369	0.37108	0.30625	3.1263	0.1936	0.27183
19	8	1200	0.23499	0.21372	0.24816	2.2736	0.17194	0.16021
19	4	1501	3.6776	3.6744	3.6809	32.091	3.6747	0.14743
19	5	1501	0.37256	0.38189	0.36298	3.5792	0.22176	0.29942
19	6	1501	0.25818	0.22878	0.28458	2.5371	0.17517	0.1897
19	7	1501	0.37538	0.36184	0.38845	3.6213	0.2401	0.28859
19	8	1501	0.23901	0.22374	0.25337	2.2682	0.17435	0.16351
19	4	1801	1.1541	1.1536	1.1549	10.0699	1.1532	0.046035
19	5	1801	0.37758	0.34033	0.42744	3.7399	0.24398	0.28821
19	6	1801	0.38844	0.36992	0.41469	3.7306	0.24333	0.30284
19	7	1801	2.3634	2.3609	2.3671	20.6324	2.3609	0.10726
19	8	1801	2.1844	2.1817	2.1886	19.0726	2.1816	0.11208
20	4	1200	1.1557	1.1542	1.1566	10.0766	1.1544	0.054232
20	5	1200	0.36147	0.33889	0.37576	3.5483	0.22658	0.28169
20	6	1200	0.37161	0.35245	0.38385	3.6222	0.24561	0.27892
20	7	1200	1.3763	1.3684	1.3815	12.0213	1.3719	0.10993
20	8	1200	0.24181	0.20557	0.2632	2.1215	0.21852	0.10356

Table 6.7: Summary of RBF-NARX model construction for identification of compressor pressure

# neurons	# delays	# training samples	RMSE$^{P_C}_{total}$	RMSE$^{P_C}_{training}$	RMSE$^{P_C}_{test}$	%RMSE$^{P_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
20	4	1501	1.8584	1.8564	1.8604	16.215	1.8564	0.085352
20	5	1501	0.3058	0.27388	0.33471	3.0288	0.19746	0.23355
20	6	1501	0.27417	0.24039	0.30424	2.7087	0.18496	0.20242
20	7	1501	1.817	1.8132	1.8207	15.8639	1.8141	0.10262
20	8	1501	0.2428	0.22181	0.26213	2.3224	0.17849	0.16463
20	4	1801	4.1517	4.1496	4.1549	36.2324	4.149	0.15036
20	5	1801	0.40426	0.3787	0.43985	3.9667	0.25692	0.31217
20	6	1801	1.9252	1.9218	1.9304	16.8117	1.9214	0.12207
20	7	1801	1.1892	1.1867	1.1929	10.387	1.1867	0.077042
20	8	1801	2.2976	2.2949	2.3016	20.0598	2.2948	0.113
10	4	1200	0.045255	0.048601	0.04288	0.38168	0.036667	0.026529
10	5	1200	0.049547	0.058181	0.042839	0.41991	0.03709	0.032857
10	6	1200	0.1011	0.11501	0.090647	0.85861	0.080238	0.06151
10	7	1200	0.049792	0.059925	0.041696	0.42267	0.036615	0.033749
10	8	1200	0.063384	0.07224	0.05672	0.53788	0.049643	0.039416
10	4	1501	0.04399	0.046569	0.041247	0.37117	0.034084	0.027814
10	5	1501	0.070668	0.075439	0.065546	0.60009	0.05507	0.044293
10	6	1501	0.098157	0.10246	0.093655	0.83352	0.077801	0.059858
10	7	1501	0.075723	0.084845	0.065333	0.64391	0.055672	0.051337
10	8	1501	0.094174	0.099654	0.088352	0.80033	0.073373	0.059045
10	4	1801	0.047926	0.052365	0.040358	0.40531	0.036169	0.031449
10	5	1801	0.038502	0.043303	0.029882	0.3256	0.026933	0.027519
10	6	1801	0.036631	0.038677	0.033326	0.30776	0.026959	0.024805

Table 6.7: Summary of RBF-NARX model construction for identification of compressor pressure

# neurons	# delays	# training samples	RMSE$_{total}^{P_C}$	RMSE$_{training}^{P_C}$	RMSE$_{test}^{P_C}$	%RMSE$_{total}^{P_C}$	mean (μ_{ae})	Std (σ_{ae})
10	7	1801	0.1107	0.12201	0.091128	0.93991	0.088808	0.0661
10	8	1801	0.11619	0.12879	0.094159	0.98687	0.092656	0.070115
11	4	1200	0.051992	0.072403	0.031802	0.44357	0.027571	0.044087
11	5	1200	0.080207	0.093272	0.070164	0.68116	0.062647	0.050093
11	6	1200	0.040124	0.051533	0.030218	0.34002	0.026533	0.030103
11	7	1200	0.12367	0.14025	0.11126	1.0508	0.098222	0.075162
11	8	1200	0.11721	0.13328	0.10514	0.99584	0.092762	0.071652
11	4	1501	0.027487	0.027771	0.0272	0.22817	0.022586	0.015669
11	5	1501	0.04925	0.051472	0.046922	0.41599	0.039263	0.029736
11	6	1501	0.059555	0.068344	0.049214	0.50653	0.042195	0.042036
11	7	1501	0.04245	0.04814	0.035864	0.35982	0.029964	0.030074
11	8	1501	0.10588	0.11102	0.10047	0.89965	0.083626	0.06495
11	4	1801	0.060627	0.065327	0.052793	0.51309	0.049614	0.03485
11	5	1801	0.050903	0.05566	0.042784	0.43074	0.039926	0.031582
11	6	1801	0.054671	0.061952	0.041406	0.46434	0.03981	0.037477
11	7	1801	0.066057	0.073566	0.05282	0.56075	0.05145	0.041437
11	8	1801	0.035093	0.039023	0.028183	0.2957	0.02369	0.025894
12	4	1200	0.10875	0.13641	0.085501	0.92731	0.07627	0.07754
12	5	1200	0.045572	0.059391	0.033323	0.38684	0.028897	0.035245
12	6	1200	0.083142	0.10497	0.064632	0.70826	0.057959	0.059619
12	7	1200	0.064396	0.07691	0.054486	0.54703	0.048821	0.042
12	8	1200	0.056249	0.067569	0.047225	0.47793	0.04124	0.038259
12	4	1501	1.8216	1.8187	1.8245	15.1073	1.8189	0.10028

Table 6.7: Summary of RBF-NARX model construction for identification of compressor pressure

# neurons	# delays	# training samples	RMSE$^{P_C}_{total}$	RMSE$^{P_C}_{training}$	RMSE$^{P_C}_{test}$	%RMSE$^{P_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
12	5	1501	0.076144	0.079472	0.072662	0.64574	0.0601	0.046762
12	6	1501	0.093508	0.10086	0.08552	0.79507	0.071535	0.06023
12	7	1501	0.041526	0.047645	0.034328	0.35198	0.02924	0.029492
12	8	1501	0.10399	0.10738	0.10049	0.88305	0.083082	0.062552
12	4	1801	0.028344	0.028788	0.027662	0.23564	0.02311	0.016413
12	5	1801	0.051061	0.054583	0.045264	0.43124	0.041898	0.029191
12	6	1801	0.03562	0.040527	0.026609	0.30139	0.023059	0.027154
12	**7**	**1801**	**0.035583**	**0.041004**	**0.025357**	**0.30124**	**0.022888**	**0.027249**
12	8	1801	0.056521	0.064461	0.041869	0.48031	0.04089	0.039028
13	4	1200	1.2741	1.2871	1.2653	10.5876	0.70762	1.0597
13	5	1200	0.033538	0.041606	0.02685	0.28245	0.022606	0.024779
13	6	1200	0.083601	0.09174	0.077706	0.70904	0.068072	0.048539
13	7	1200	0.032045	0.037664	0.027674	0.26878	0.023128	0.022184
13	8	1200	0.050918	0.06243	0.041511	0.43244	0.036921	0.03507
13	4	1501	0.032368	0.035112	0.029368	0.27105	0.02435	0.021329
13	5	1501	0.045542	0.05078	0.039613	0.38571	0.033105	0.03128
13	6	1501	0.072641	0.075026	0.070174	0.61587	0.058254	0.043404
13	7	1501	0.069493	0.07567	0.062705	0.59061	0.052806	0.045183
13	8	1501	0.052283	0.057391	0.046614	0.44387	0.038782	0.03507
13	4	1801	3.2764	3.2721	3.2827	27.1722	3.2704	0.1984
13	5	1801	0.04808	0.050947	0.043422	0.40563	0.039531	0.027372
13	6	1801	0.071332	0.082952	0.048967	0.60806	0.047868	0.052895
13	7	1801	0.054074	0.062127	0.038982	0.45981	0.03667	0.039747

Table 6.7: Summary of RBF-NARX model construction for identification of compressor pressure

# neurons	# delays	# training samples	RMSE$^{P_C}_{total}$	RMSE$^{P_C}_{training}$	RMSE$^{P_C}_{test}$	%RMSE$^{P_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
13	8	1801	0.043556	0.049781	0.032016	0.36956	0.029975	0.031606
14	4	1200	1.0129	1.0111	1.0141	8.4004	1.0118	0.046425
14	5	1200	0.064431	0.069885	0.060525	0.54542	0.052523	0.037325
14	6	1200	0.091408	0.10307	0.082733	0.77585	0.073103	0.054884
14	7	1200	0.054535	0.072042	0.038698	0.4649	0.03396	0.042678
14	8	1200	0.11768	0.13147	0.10751	0.99915	0.094852	0.069666
14	4	1501	0.036899	0.040352	0.033085	0.31077	0.026804	0.025363
14	5	1501	0.073946	0.096419	0.040456	0.63434	0.035644	0.064799
14	6	1501	0.05818	0.05986	0.056447	0.49221	0.047427	0.033703
14	7	1501	0.044687	0.049362	0.03946	0.37684	0.032871	0.030278
14	8	1501	0.045384	0.049522	0.040825	0.38408	0.034028	0.030035
14	4	1801	0.039974	0.042503	0.035846	0.33629	0.03186	0.024147
14	5	1801	0.055296	0.06248	0.042282	0.4693	0.039759	0.038437
14	6	1801	0.14598	0.18156	0.06182	1.2542	0.0664	0.13003
14	7	1801	0.083208	0.09505	0.061281	0.70827	0.060448	0.05719
14	8	1801	0.063606	0.074342	0.042699	0.54247	0.042109	0.047679
15	4	1200	1.9361	1.9329	1.9383	16.0587	1.9341	0.088474
15	5	1200	2.2672	2.2603	2.2718	18.8016	2.263	0.13813
15	6	1200	0.10616	0.15322	0.056008	0.91005	0.052929	0.092037
15	7	1200	0.045481	0.061963	0.02981	0.38714	0.027039	0.036577
15	8	1200	0.040803	0.048874	0.03439	0.34403	0.028927	0.028783
15	4	1501	0.066835	0.074775	0.057809	0.56847	0.047925	0.046593
15	5	1501	0.052253	0.055173	0.049158	0.4425	0.040493	0.033031

Table 6.7: Summary of RBF-NARX model construction for identification of compressor pressure

# neurons	# delays	# training samples	RMSE$^{P_C}_{total}$	RMSE$^{P_C}_{training}$	RMSE$^{P_C}_{test}$	%RMSE$^{P_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
15	6	1501	0.066257	0.074895	0.056302	0.56385	0.047381	0.046322
15	7	1501	0.083473	0.10278	0.058041	0.71321	0.050036	0.066826
15	8	1501	0.035413	0.040005	0.030124	0.29816	0.024823	0.02526
15	4	1801	2.5369	2.5352	2.5394	21.0397	2.5345	0.11054
15	5	1801	0.11075	0.13572	0.055031	0.94981	0.057266	0.094809
15	6	1801	1.0939	1.0932	1.095	9.0726	1.0929	0.046844
15	7	1801	0.04905	0.057733	0.031847	0.41803	0.030275	0.038598
15	8	1801	0.073294	0.086017	0.048268	0.62498	0.048125	0.05529
16	4	1200	0.20233	0.3163	0.039462	1.7441	0.049222	0.19629
16	5	1200	0.13017	0.20044	0.038301	1.1202	0.039069	0.12419
16	6	1200	1.8355	1.8336	1.8368	15.2233	1.8343	0.065504
16	7	1200	0.059929	0.06423	0.056884	0.50723	0.049165	0.034274
16	8	1200	0.044958	0.057679	0.03393	0.38183	0.029459	0.033966
16	4	1501	0.06301	0.080856	0.037433	0.53855	0.031566	0.054542
16	5	1501	0.064637	0.067983	0.061106	0.54798	0.05079	0.039986
16	6	1501	0.041816	0.048603	0.033681	0.35423	0.028583	0.030527
16	7	1501	0.065641	0.081933	0.043621	0.56149	0.037257	0.054052
16	8	1501	0.056182	0.061621	0.050152	0.47715	0.041609	0.037757
16	4	1801	0.087095	0.10617	0.045309	0.74603	0.045412	0.074331
16	5	1801	0.075267	0.082307	0.063247	0.6381	0.060591	0.044661
16	6	1801	2.2419	2.2398	2.2451	18.594	2.239	0.11474
16	7	1801	0.057159	0.063748	0.045514	0.48517	0.043217	0.037415
16	8	1801	0.053814	0.065003	0.03001	0.45964	0.028763	0.04549

Table 6.7: Summary of RBF-NARX model construction for identification of compressor pressure

# neurons	# delays	# training samples	RMSE$^{P_C}_{total}$	RMSE$^{P_C}_{training}$	RMSE$^{P_C}_{test}$	%RMSE$^{P_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
17	4	1200	0.15511	0.23959	0.042926	1.3391	0.043933	0.14878
17	5	1200	1.1879	1.1846	1.1901	9.8511	1.186	0.066559
17	6	1200	0.032872	0.040236	0.026866	0.27645	0.02281	0.023673
17	7	1200	0.058083	0.062424	0.055002	0.49126	0.047694	0.033156
17	8	1200	0.059656	0.083781	0.035401	0.51003	0.032174	0.050245
17	4	1501	2.5692	2.5577	2.5806	21.3007	2.5586	0.23329
17	5	1501	0.1218	0.15872	0.066853	1.0456	0.058771	0.1067
17	6	1501	0.099118	0.10047	0.097746	0.84125	0.079942	0.058606
17	7	1501	2.3399	2.3361	2.3437	19.406	2.3363	0.12902
17	8	1501	0.094925	0.11511	0.069047	0.81088	0.05935	0.074096
17	4	1801	1.964	1.9629	1.9656	16.289	1.9624	0.078848
17	5	1801	0.077919	0.087389	0.061006	0.66198	0.059042	0.050856
17	6	1801	0.048496	0.054455	0.037831	0.41124	0.034659	0.033927
17	7	1801	0.08177	0.091503	0.064462	0.69492	0.063171	0.051929
17	8	1801	0.082432	0.099809	0.045189	0.70564	0.046355	0.068174
18	4	1200	1.5146	1.51	1.5177	12.5608	1.5117	0.093573
18	5	1200	2.3986	2.3895	2.4047	19.8921	2.3931	0.16367
18	6	1200	0.073466	0.10596	0.03889	0.62892	0.035486	0.064338
18	7	1200	1.7734	1.762	1.7809	14.7043	1.7663	0.15872
18	8	1200	0.033752	0.038904	0.029829	0.28434	0.024965	0.022718
18	4	1501	2.1436	2.142	2.1451	17.7791	2.1418	0.087252
18	5	1501	0.092963	0.1156	0.062587	0.79581	0.053528	0.076018
18	6	1501	1.4178	1.4167	1.419	11.7595	1.4169	0.052434

Table 6.7: Summary of RBF-NARX model construction for identification of compressor pressure

# neurons	# delays	# training samples	RMSE$^{P_C}_{total}$	RMSE$^{P_C}_{training}$	RMSE$^{P_C}_{test}$	%RMSE$^{P_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
18	7	1501	0.048777	0.048848	0.048705	0.41184	0.040067	0.027822
18	8	1501	0.26333	0.36648	0.065893	2.2719	0.070491	0.25377
18	4	1801	0.11862	0.14104	0.073029	1.0134	0.072469	0.093927
18	5	1801	0.051237	0.055162	0.044704	0.4332	0.041232	0.030422
18	6	1801	0.1944	0.24529	0.064903	1.6748	0.072147	0.18055
18	7	1801	0.18489	0.23726	0.03174	1.5941	0.047631	0.17868
18	8	1801	0.032492	0.036435	0.025454	0.27376	0.022677	0.023274
19	4	1200	1.3706	1.3679	1.3725	11.3689	1.3684	0.078338
19	5	1200	0.11223	0.12982	0.098791	0.95359	0.087921	0.069768
19	6	1200	2.7996	2.7956	2.8023	23.2182	2.7973	0.11207
19	7	1200	2.6736	2.6684	2.6771	22.1745	2.6706	0.12778
19	8	1200	0.077209	0.10484	0.051089	0.66026	0.046285	0.061808
19	4	1501	1.3338	1.3329	1.3346	11.0615	1.3327	0.052835
19	5	1501	0.076719	0.098805	0.044794	0.65697	0.039529	0.065763
19	6	1501	2.1384	2.1349	2.1419	17.7332	2.1351	0.11808
19	7	1501	0.079964	0.095365	0.060764	0.68281	0.051883	0.060857
19	8	1501	0.1804	0.22911	0.11219	1.5463	0.098717	0.15102
19	4	1801	2.9608	2.9569	2.9667	24.5674	1.4772	2.5665
19	5	1801	0.045958	0.055135	0.026829	0.39134	0.023892	0.039266
19	6	1801	1.7208	1.7191	1.7235	14.2717	1.7187	0.086636
19	7	1801	0.077622	0.093526	0.044044	0.66366	0.044036	0.063933
19	8	1801	0.055913	0.061076	0.047114	0.47354	0.044227	0.034214
20	4	1200	3.992	3.9849	3.9967	33.1073	3.9874	0.19111

Table 6.7: Summary of RBF-NARX model construction for identification of compressor pressure

# neurons	# delays	# training samples	RMSE$_{total}^{P_C}$	RMSE$_{training}^{P_C}$	RMSE$_{test}^{P_C}$	%RMSE$_{total}^{P_C}$	mean (μ_{ae})	Std (σ_{ae})
20	5	1200	2.2234	2.2186	2.2266	18.4401	2.2206	0.11152
20	6	1200	0.18978	0.29207	0.056375	1.6344	0.058052	0.18072
20	7	1200	0.026612	0.027382	0.026087	0.22105	0.021582	0.015573
20	8	1200	1.1471	1.1456	1.148	9.5132	1.1462	0.043905
20	4	1501	2.0735	2.072	2.0749	17.1978	2.0722	0.072711
20	5	1501	3.1189	3.1143	3.1235	25.865	3.1145	0.16564
20	6	1501	1.7084	1.7046	1.7123	14.1676	1.7047	0.11207
20	7	1501	0.07222	0.07616	0.068049	0.61267	0.05719	0.04411
20	8	1501	0.09826	0.13648	0.026036	0.84626	0.027685	0.094295
20	4	1801	2.2607	2.2594	2.2626	18.7494	2.2589	0.089497
20	5	1801	0.29094	0.37354	0.047716	2.5141	0.062761	0.28414
20	6	1801	0.10058	0.11274	0.078897	0.85496	0.077371	0.064281
20	7	1801	0.029222	0.030812	0.026658	0.24434	0.023319	0.017613
20	8	1801	0.071298	0.08182	0.051628	0.60701	0.051156	0.049672

Table 6.8: Summary of RBF-NARX model construction for identification of rotational speed

# neurons	# delays	# training samples	RMSE$_{total}^{N}$	RMSE$_{training}^{N}$	RMSE$_{test}^{N}$	%RMSE$_{total}^{N}$	mean (μ_{ae})	Std (σ_{ae})
10	4	1200	118.567	118.3704	118.6978	0.99667	118.4587	5.0664
10	5	1200	138.6245	138.4739	138.7247	1.1653	138.5356	4.9642
10	6	1200	143.3147	142.3956	143.9238	1.2043	143.046	8.7721
10	7	1200	140.3586	139.4537	140.9584	1.1795	140.0928	8.6367
10	8	1200	112.5416	111.5024	113.2287	0.94555	112.2358	8.2925
10	4	1501	116.8197	131.0573	100.5751	0.99035	95.5964	67.1542

Table 6.8: Summary of RBF-NARX model construction for identification of rotational speed

# neurons	# delays	# training samples	RMSE$^N_{total}$	RMSE$^N_{training}$	RMSE$^N_{test}$	%RMSE$^N_{total}$	mean (μ_{ae})	Std (σ_{ae})
10	5	1501	91.4612	90.4949	92.418	0.77149	74.7798	52.6693
10	6	1501	110.9079	110.3935	111.4202	0.93213	110.5313	9.1329
10	7	1501	82.1303	61.9147	98.2816	0.6843	49.3958	65.6268
10	8	1501	122.442	118.751	126.0272	1.0282	119.5219	26.5855
10	4	1801	148.8705	148.8179	148.9494	1.2514	148.7983	4.6357
10	5	1801	74.2458	77.7809	68.5991	0.62725	62.0095	40.8389
10	6	1801	106.1983	91.911	124.6039	0.88821	79.5694	70.3452
10	7	1801	54.6591	53.9191	55.7513	0.4607	45.8394	29.7768
10	8	1801	74.3076	80.1776	64.5029	0.62913	61.5287	41.6704
11	4	1200	187.566	187.3236	187.7274	1.5767	187.4148	7.5307
11	5	1200	182.0991	181.926	182.2143	1.5308	181.9879	6.364
11	6	1200	125.2834	125.139	125.3795	1.0532	125.1935	4.745
11	7	1200	107.505	105.8796	108.5745	0.90312	106.8773	11.602
11	8	1200	127.3944	126.3958	128.0555	1.0713	102.2268	76.0325
11	4	1501	198.519	191.7188	205.0983	1.6681	192.1677	49.8216
11	5	1501	126.5581	141.7409	109.2735	1.0725	105.8762	69.3453
11	6	1501	148.1275	166.0076	127.7547	1.2546	127.1138	76.0643
11	7	1501	61.8822	52.4594	70.0539	0.51771	43.9866	43.5341
11	8	1501	95.4988	77.2658	110.7797	0.79698	69.6312	65.3678
11	4	1801	104.6924	104.6591	104.7424	0.88007	104.6435	3.2012
11	5	1801	129.6726	144.5396	103.4239	1.0988	109.1068	70.088
11	6	1801	101.1666	112.8487	80.5132	0.85769	81.6375	59.7595
11	7	1801	89.5914	94.371	81.8963	0.75748	75.3896	48.413

Table 6.8: Summary of RBF-NARX model construction for identification of rotational speed

# neurons	# delays	# training samples	RMSE$^N_{total}$	RMSE$^N_{training}$	RMSE$^N_{test}$	%RMSE$^N_{total}$	mean (μ_{ae})	Std (σ_{ae})
11	8	1801	59.0911	47.2497	73.3596	0.49287	36.8988	46.1621
12	4	1200	154.0742	153.961	154.1496	1.2952	154.0045	4.6357
12	5	1200	131.1338	130.6356	131.4648	1.1022	130.8989	7.8474
12	6	1200	145.3746	167.4808	128.552	1.2316	123.9898	75.9093
12	7	1200	140.111	138.7068	141.0389	1.1773	139.5494	12.5348
12	8	1200	185.9278	182.2081	188.3654	1.561	184.7441	20.95
12	4	1501	120.6407	120.5817	120.6997	1.0141	120.586	3.6313
12	5	1501	125.6828	125.507	125.8585	1.0565	125.5421	5.9475
12	6	1501	128.2679	144.1459	110.1109	1.0872	105.7388	72.6204
12	7	1501	110.2457	112.7494	107.682	0.92678	94.5956	56.6291
12	8	1501	161.1686	160.6853	161.6507	1.3547	160.772	11.3007
12	4	1801	102.2751	100.0085	105.5857	0.8594	99.077	25.3802
12	5	1801	138.1374	134.8228	142.968	1.1607	133.6537	34.915
12	6	1801	111.5261	111.2751	111.9018	0.93747	111.1847	8.722
12	7	1801	149.4662	143.3673	158.1789	1.256	142.0304	46.5643
12	8	1801	50.2835	53.4578	45.1021	0.42529	43.4647	25.2877
13	4	1200	163.5469	163.4334	163.6225	1.3748	163.4857	4.475
13	5	1200	143.9317	167.2578	126.0151	1.2197	120.4286	78.8377
13	6	1200	141.5269	165.079	123.3629	1.1993	117.3758	79.0878
13	7	1200	105.7315	104.7936	106.3519	0.88877	105.1323	11.2431
13	8	1200	64.1622	51.0525	71.5764	0.53556	40.7252	49.589
13	4	1501	149.5453	149.4725	149.6181	1.2571	149.4744	4.6043
13	5	1501	125.9023	141.9513	107.4697	1.0676	101.2818	74.8014

Table 6.8: Summary of RBF-NARX model construction for identification of rotational speed

# neurons	# delays	# training samples	RMSE$^N_{total}$	RMSE$^N_{training}$	RMSE$^N_{test}$	%RMSE$^N_{total}$	mean (μ_{ae})	Std (σ_{ae})
13	6	1501	160.6022	160.4928	160.7116	1.3501	160.5099	5.4447
13	7	1501	131.8204	130.8059	132.8278	1.1081	130.8466	15.9961
13	8	1501	101.011	100.2333	101.7832	0.84913	100.3407	11.6192
13	4	1801	103.5608	103.5335	103.6018	0.87056	103.5258	2.6947
13	5	1801	143.9336	139.0946	150.9052	1.2093	90.4553	111.9772
13	6	1801	141.8952	157.3164	114.9308	1.2015	123.197	70.416
13	7	1801	134.5244	149.8718	107.4526	1.1398	113.7847	71.7744
13	8	1801	88.5059	96.9864	73.9756	0.74997	71.1803	52.6079
14	4	1200	193.069	192.9233	193.1661	1.623	192.9812	5.8244
14	5	1200	111.4357	111.3557	111.489	0.93676	111.3844	3.3821
14	6	1200	137.8684	159.2897	121.5169	1.1683	116.0658	74.4195
14	7	1200	147.9648	173.0284	128.5809	1.2538	123.3927	81.6703
14	8	1200	155.3776	181.1303	135.5288	1.3159	132.2648	81.5501
14	4	1501	160.7593	160.6815	160.8372	1.3514	160.6908	4.6951
14	5	1501	129.4571	144.9806	111.7859	1.0969	109.4717	69.1136
14	6	1501	132.6148	148.7176	114.2516	1.124	110.2971	73.6414
14	7	1501	149.7606	166.4367	130.9648	1.2675	131.6671	71.3703
14	8	1501	142.8018	159.3518	124.05	1.2095	123.1979	72.2245
14	4	1801	109.0475	109.0137	109.0982	0.91668	109.0045	3.0609
14	5	1801	124.276	124.2382	124.3326	1.0447	124.2166	3.841
14	6	1801	103.2836	103.2458	103.3404	0.86823	103.224	3.5108
14	7	1801	153.0669	170.3251	122.691	1.296	131.6667	78.0729
14	8	1801	132.3178	131.4718	133.5775	1.1122	131.1579	17.4845

Table 6.8: Summary of RBF-NARX model construction for identification of rotational speed

# neurons	# delays	# training samples	RMSE$^N_{total}$	RMSE$^N_{training}$	RMSE$^N_{test}$	%RMSE$^N_{total}$	mean (μ_{ae})	Std (σ_{ae})
15	4	1200	188.9176	188.7816	189.0082	1.5881	188.8394	5.4344
15	5	1200	189.0883	188.9571	189.1756	1.5895	189.0157	5.2397
15	6	1200	153.0785	143.7442	158.994	1.2863	147.5421	40.8033
15	7	1200	165.4212	165.0233	165.6858	1.3906	165.1674	9.1619
15	8	1200	148.6452	171.7218	131.0323	1.2591	127.143	77.0196
15	4	1501	124.3326	124.2892	124.3759	1.0452	124.2912	3.207
15	5	1501	128.1748	128.1319	128.2177	1.0775	128.1322	3.3027
15	6	1501	141.87	159.3568	121.8856	1.2016	121.4664	73.3131
15	7	1501	145.4795	162.3275	126.3918	1.2319	126.3271	72.163
15	8	1501	149.8521	167.3655	129.9862	1.2688	129.9252	74.6791
15	4	1801	104.9228	104.8979	104.9601	0.88201	104.8878	2.7089
15	5	1801	121.1159	121.0793	121.1707	1.0181	121.0597	3.6879
15	6	1801	146.57	146.4977	146.6784	1.2321	146.4764	5.2381
15	7	1801	136.0246	151.4872	108.768	1.1525	115.5601	71.7654
15	8	1801	165.1245	174.6201	149.7474	1.3954	135.15	94.8871
16	4	1200	160.9517	160.8704	161.0059	1.353	160.8982	4.1493
16	5	1200	124.3016	124.1498	124.4027	1.0449	124.2186	4.5442
16	6	1200	187.3906	187.0991	187.5845	1.5753	187.1932	8.6005
16	7	1200	105.29	13.193	135.4864	0.88014	32.8083	100.0646
16	8	1200	149.669	151.4778	148.4516	1.2663	103.9657	107.6839
16	4	1501	132.7762	132.7301	132.8223	1.1162	132.7321	3.4246
16	5	1501	125.2926	125.2073	125.3779	1.0532	125.2121	4.4915
16	6	1501	99.8341	99.6802	99.9879	0.83925	99.6838	5.4783

Table 6.8: Summary of RBF-NARX model construction for identification of rotational speed

# neurons	# delays	# training samples	RMSE$^N_{total}$	RMSE$^N_{training}$	RMSE$^N_{test}$	%RMSE$^N_{total}$	mean (μ_{ae})	Std (σ_{ae})
16	7	1501	143.8729	160.3881	125.1844	1.2182	125.0647	71.1331
16	8	1501	147.2423	164.4998	127.6589	1.2471	126.7165	75.0006
16	4	1801	119.696	119.6682	119.7376	1.0062	119.6561	3.0877
16	5	1801	137.9771	137.9178	138.066	1.1599	137.9047	4.4695
16	6	1801	193.966	193.8441	194.1489	1.6305	193.8172	7.598
16	7	1801	133.526	133.4788	133.5967	1.1225	133.444	4.6782
16	8	1801	152.8045	168.3582	125.9054	1.2931	134.6588	72.2354
17	4	1200	156.2535	156.1748	156.3059	1.3135	156.2016	4.0279
17	5	1200	100.8808	100.8282	100.9158	0.84803	100.8473	2.6007
17	6	1200	180.7924	180.6297	180.9008	1.5198	180.6853	6.2259
17	7	1200	180.4423	149.5515	198.3721	1.5168	150.3707	99.7566
17	8	1200	155.0478	178.6612	137.0744	1.313	133.688	78.5454
17	4	1501	109.3911	109.3528	109.4294	0.91957	109.3547	2.8215
17	5	1501	166.7579	166.6649	166.851	1.4018	166.6751	5.2566
17	6	1501	159.1486	158.8199	159.4768	1.3378	158.8342	9.9992
17	7	1501	198.3663	202.9424	193.6789	1.6718	109.5602	165.3931
17	8	1501	141.0683	157.6225	122.2799	1.195	120.7233	72.9926
17	4	1801	103.4299	103.4071	103.4642	0.86946	103.3956	2.6661
17	5	1801	131.6153	131.5748	131.676	1.1064	131.5567	3.9279
17	6	1801	105.1297	105.1088	105.1611	0.88377	105.066	3.6597
17	7	1801	117.5298	114.4912	121.9483	0.98767	113.349	31.0739
17	8	1801	162.3177	178.5691	134.2856	1.3731	143.2879	76.2729
18	4	1200	138.5035	138.4339	138.5498	1.1643	138.4575	3.5693

Table 6.8: Summary of RBF-NARX model construction for identification of rotational speed

# neurons	# delays	# training samples	RMSE$^N_{total}$	RMSE$^N_{training}$	RMSE$^N_{test}$	%RMSE$^N_{total}$	mean (μ_{ae})	Std (σ_{ae})
18	5	1200	110.9355	110.8609	110.9851	0.93255	110.8946	3.0099
18	6	1200	127.1022	126.9931	127.1749	1.0685	127.028	4.3434
18	7	1200	109.2968	109.0603	109.4542	0.91875	109.1609	5.4496
18	8	1200	116.3631	110.4611	120.1348	0.97775	112.7685	28.704
18	4	1501	178.5482	178.4888	178.6076	1.5009	178.4889	4.6019
18	5	1501	123.8616	123.8185	123.9048	1.0412	123.8205	3.1937
18	6	1501	140.4227	140.3486	140.4968	1.1804	140.3544	4.379
18	7	1501	180.7813	180.6526	180.9101	1.5197	180.6558	6.7369
18	8	1501	156.2365	174.1055	136.0258	1.3222	136.9175	75.2684
18	4	1801	190.5705	190.5284	190.6336	1.602	190.5072	4.9124
18	5	1801	147.6573	147.6247	147.7061	1.2412	147.6083	3.8046
18	6	1801	149.219	149.1744	149.2858	1.2544	149.1627	4.099
18	7	1801	105.0991	96.7681	116.4899	0.88017	42.1574	96.2895
18	8	1801	123.4976	123.4639	123.5481	1.0381	123.4557	3.2199
19	4	1200	105.706	105.6262	105.7591	0.88859	105.6538	3.3221
19	5	1200	104.7804	104.7263	104.8164	0.88081	104.7456	2.7007
19	6	1200	100.6417	100.5892	100.6767	0.84602	100.6083	2.5936
19	7	1200	175.8074	175.3908	176.0844	1.4778	175.5798	8.9439
19	8	1200	195.6385	178.7287	206.1369	1.6442	186.0336	60.5568
19	4	1501	111.592	111.541	111.643	0.93807	111.5509	3.0275
19	5	1501	162.8035	162.7261	162.8809	1.3686	162.7376	4.6306
19	6	1501	152.9837	152.9045	153.0629	1.286	152.9089	4.7843
19	7	1501	146.5968	146.5215	146.6721	1.2323	146.5255	4.5723

Table 6.8: Summary of RBF-NARX model construction for identification of rotational speed

# neurons	# delays	# training samples	RMSE$^N_{total}$	RMSE$^N_{training}$	RMSE$^N_{test}$	%RMSE$^N_{total}$	mean (μ_{ae})	Std (σ_{ae})
19	8	1501	102.8393	98.9187	106.6182	0.86421	99.2251	27.0284
19	4	1801	187.6882	187.6468	187.7503	1.5778	187.6259	4.8365
19	5	1801	146.9256	146.8932	146.9743	1.2351	146.8769	3.786
19	6	1801	119.515	119.4833	119.5625	1.0047	119.4592	3.6532
19	7	1801	190.7536	190.7013	190.832	1.6035	190.6872	5.0328
19	8	1801	180.7319	175.1315	188.8257	1.5193	173.5615	50.4108
20	4	1200	105.5638	105.4856	105.6159	0.8874	105.5194	3.0629
20	5	1200	157.918	157.8392	157.9705	1.3275	157.8656	4.0693
20	6	1200	118.583	118.2188	118.8251	0.99678	118.3918	6.7323
20	7	1200	138.4243	138.3254	138.4901	1.1636	138.3648	4.0586
20	8	1200	140.565	132.0434	145.967	1.1812	135.5051	37.3812
20	4	1501	165.4456	165.3787	165.5125	1.3908	165.3892	4.3194
20	5	1501	131.6426	131.5986	131.6865	1.1066	131.5989	3.392
20	6	1501	110.3316	110.2939	110.3694	0.92748	110.295	2.8424
20	7	1501	123.1024	123.0548	123.15	1.0348	123.0609	3.1966
20	8	1501	100.3862	100.2958	100.4767	0.84389	100.2986	4.1938
20	4	1801	166.3651	166.349	166.3892	1.3985	166.3033	4.5343
20	5	1801	180.5676	180.5278	180.6274	1.5179	180.5077	4.6532
20	6	1801	127.7655	127.7305	127.8179	1.0741	127.687	4.4788
20	7	1801	118.2387	118.174	118.3356	0.99393	118.1623	4.2489
20	8	1801	132.0718	129.1096	136.3969	1.1096	128.2404	31.5861
10	4	1200	197.8189	197.6827	197.9095	1.6656	197.7329	5.8314
10	5	1200	31.7621	35.6254	28.9027	0.26792	24.2162	20.556

Table 6.8: Summary of RBF-NARX model construction for identification of rotational speed

# neurons	# delays	# training samples	RMSE$^N_{total}$	RMSE$^N_{training}$	RMSE$^N_{test}$	%RMSE$^N_{total}$	mean (μ_{ae})	Std (σ_{ae})
10	6	1200	27.7337	28.9878	26.8656	0.23414	22.6132	16.0589
10	7	1200	24.1489	24.7567	23.7352	0.20394	19.7707	13.8691
10	8	1200	23.7161	26.2896	21.8336	0.2001	17.9005	15.5597
10	4	1501	136.4467	136.3816	136.5119	1.1489	136.3859	4.0737
10	5	1501	158.5063	158.4054	158.6072	1.3346	158.4169	5.3233
10	6	1501	134.2091	134.0978	134.3204	1.13	134.1031	5.3345
10	7	1501	23.5804	25.8368	21.0821	0.1991	19.4593	13.3203
10	8	1501	24.3621	26.6978	21.7756	0.20575	19.8336	14.1496
10	4	1801	192.895	192.8469	192.967	1.6242	192.8006	6.0345
10	5	1801	104.1589	104.126	104.2082	0.87702	104.1184	2.9045
10	6	1801	30.0592	32.1954	26.5323	0.25368	23.8423	18.3088
10	7	1801	25.3203	26.6328	23.2116	0.21381	20.6473	14.6586
10	8	1801	27.0634	28.578	24.6159	0.22848	22.2212	15.4507
11	4	1200	109.3886	109.3176	109.4359	0.92105	109.3499	2.9125
11	5	1200	99.5778	99.4775	99.6446	0.83845	99.5155	3.5241
11	6	1200	99.7219	99.6412	99.7756	0.83966	99.673	3.1213
11	7	1200	19.3312	19.7905	19.019	0.16326	15.5321	11.5105
11	8	1200	24.6523	25.9969	23.7142	0.20813	19.6357	14.908
11	4	1501	147.0733	146.9979	147.1486	1.2384	147.0007	4.6188
11	5	1501	158.848	158.6143	159.0815	1.3375	158.624	8.4339
11	6	1501	48.2913	62.0058	28.6071	0.40593	28.1757	39.2261
11	7	1501	194.8794	194.7818	194.9771	1.6409	194.7849	6.0712
11	8	1501	17.8349	19.7104	15.7359	0.15058	14.2826	10.6831

Table 6.8: **Summary of RBF-NARX model construction for identification of rotational speed**

# neurons	# delays	# training samples	RMSE$^N_{total}$	RMSE$^N_{training}$	RMSE$^N_{test}$	%RMSE$^N_{total}$	mean (μ_{ae})	Std (σ_{ae})
11	4	1801	108.5608	108.5258	108.6134	0.91409	108.512	3.2558
11	5	1801	162.7971	162.7248	162.9056	1.3708	162.7041	5.5039
11	6	1801	101.546	101.5122	101.5968	0.85502	101.5008	3.033
11	7	1801	106.6311	106.5568	106.7425	0.89784	106.5324	4.5859
11	8	1801	31.5214	33.7312	27.878	0.26603	25.1598	18.9921
12	4	1200	134.7526	134.6541	134.8182	1.1346	134.6957	3.9163
12	5	1200	158.8403	158.9074	158.7956	1.3374	158.7434	5.5485
12	6	1200	174.0177	173.679	174.243	1.4652	173.8199	8.2962
12	7	1200	22.3895	22.8313	22.0902	0.18909	18.1601	13.098
12	8	1200	21.4239	21.9979	21.0327	0.18094	17.2234	12.7433
12	4	1501	110.3973	110.3074	110.4871	0.92954	110.3119	4.3402
12	5	1501	187.9859	187.7416	188.23	1.5829	174.8184	69.129
12	6	1501	132.477	132.4127	132.5414	1.1155	132.4211	3.848
12	7	1501	40.6883	52.9485	22.5115	0.34188	21.7558	34.3893
12	8	1501	23.8213	27.2392	19.8196	0.201	17.6634	15.9857
12	4	1801	103.9238	103.8889	103.9763	0.87503	103.8728	3.2563
12	5	1801	103.9092	103.8261	104.0339	0.87492	103.7993	4.7791
12	6	1801	106.6845	106.6496	106.7368	0.89828	106.6329	3.3189
12	7	1801	142.3174	142.1863	142.5141	1.1983	142.1493	6.9164
12	8	1801	28.8442	30.4067	26.3257	0.24355	23.7556	16.363
13	4	1200	121.39	121.3101	121.4432	1.0221	121.3465	3.2497
13	5	1200	156.4517	156.1772	156.6343	1.3173	156.2865	7.1886
13	6	1200	138.099	138.049	138.1323	1.1628	138.035	4.204

Table 6.8: Summary of RBF-NARX model construction for identification of rotational speed

# neurons	# delays	# training samples	RMSE$^N_{total}$	RMSE$^N_{training}$	RMSE$^N_{test}$	%RMSE$^N_{total}$	mean (μ_{ae})	Std (σ_{ae})
13	7	1200	148.9182	148.7316	149.0424	1.2539	148.8092	5.6981
13	8	1200	109.3495	109.194	109.453	0.92073	109.2604	4.4148
13	4	1501	192.9423	192.852	193.0325	1.6246	192.8686	5.3344
13	5	1501	156.4688	156.3694	156.5681	1.3175	156.3932	4.8641
13	6	1501	191.3481	191.1951	191.501	1.6112	191.2101	7.2664
13	7	1501	35.6909	43.1833	26.1244	0.30062	24.4982	25.9596
13	8	1501	27.3397	30.7586	23.4242	0.23076	21.3602	17.0674
13	4	1801	120.1863	120.1469	120.2453	1.012	120.1289	3.7127
13	5	1801	109.3148	109.269	109.3835	0.92043	109.2466	3.8623
13	6	1801	33.626	35.8951	29.8989	0.28381	27.5591	19.2697
13	7	1801	136.2239	136.1643	136.3133	1.147	136.1494	4.506
13	8	1801	130.9234	130.7983	131.1108	1.1024	130.762	6.4999
14	4	1200	108.5498	108.469	108.6036	0.91398	108.4997	3.2991
14	5	1200	122.1959	122.0905	122.266	1.0289	122.1344	3.8757
14	6	1200	125.0431	125.6167	124.6595	1.0526	66.8142	105.7136
14	7	1200	102.228	101.9847	102.3898	0.86077	102.1024	5.0666
14	8	1200	19.3903	19.8797	19.0573	0.16379	15.0864	12.1833
14	4	1501	155.4973	155.4212	155.5735	1.3093	155.4321	4.5054
14	5	1501	114.4408	114.3844	114.4971	0.96359	114.3857	3.55
14	6	1501	112.9992	112.9339	113.0645	0.95146	112.9418	3.6019
14	7	1501	132.1841	132.0747	132.2935	1.113	132.0771	5.3177
14	8	1501	34.1636	43.3203	21.3818	0.28735	20.0418	27.6719
14	4	1801	143.4549	143.3896	143.5528	1.2079	143.3655	5.0645

Table 6.8: Summary of RBF-NARX model construction for identification of rotational speed

# neurons	# delays	# training samples	RMSE$^N_{total}$	RMSE$^N_{training}$	RMSE$^N_{test}$	%RMSE$^N_{total}$	mean (μ_{ae})	Std (σ_{ae})
14	5	1801	133.7946	133.7501	133.8614	1.1265	133.7348	4.0015
14	6	1801	103.4022	103.3651	103.4579	0.87064	103.3425	3.5131
14	7	1801	135.5024	135.4101	135.6406	1.1409	135.3818	5.716
14	8	1801	26.387	27.7557	24.188	0.22281	21.5437	15.2388
15	4	1200	128.0073	127.8447	128.1154	1.0778	127.9096	4.9997
15	5	1200	184.0351	183.8988	184.1258	1.5496	183.9527	5.5085
15	6	1200	158.9563	158.8395	159.0341	1.3384	158.8873	4.6839
15	7	1200	128.8983	128.7768	128.9792	1.0853	59.3397	114.4462
15	8	1200	123.7	195.5603	3.9289	1.0359	16.675	122.5914
15	4	1501	169.7219	169.5944	169.8493	1.4291	169.6078	6.2224
15	5	1501	186.9378	186.8444	187.0312	1.574	186.8535	5.6137
15	6	1501	32.2446	36.7231	27.0302	0.27199	25.5307	19.6983
15	7	1501	185.7616	185.4671	186.0558	1.5641	185.5246	9.3829
15	8	1501	107.7857	107.5937	107.9776	0.90758	107.6049	6.2411
15	4	1801	122.8516	122.8095	122.9146	1.0344	122.7918	3.8329
15	5	1801	120.6032	120.5638	120.6624	1.0155	120.5464	3.7039
15	6	1801	154.7901	154.7438	154.8595	1.3033	154.7332	4.196
15	7	1801	129.9287	132.1046	126.5929	1.0939	118.8526	52.5018
15	8	1801	150.2668	150.2065	150.3572	1.2652	150.1911	4.7678
16	4	1200	142.3425	142.2363	142.4132	1.1985	142.2734	4.4355
16	5	1200	191.3012	191.0476	191.4699	1.6108	191.1466	7.691
16	6	1200	174.9901	174.5932	175.2541	1.4734	174.7426	9.3056
16	7	1200	106.1852	106.1148	106.232	0.89406	47.1922	95.1378

Table 6.8: Summary of RBF-NARX model construction for identification of rotational speed

# neurons	# delays	# training samples	RMSE$^N_{total}$	RMSE$^N_{training}$	RMSE$^N_{test}$	%RMSE$^N_{total}$	mean (μ_{ae})	Std (σ_{ae})
16	8	1200	109.8723	109.7892	109.9276	0.92513	109.8199	3.3931
16	4	1501	134.7526	134.6856	134.8196	1.1346	134.6866	4.217
16	5	1501	184.8367	184.7104	184.9631	1.5563	184.72	6.5678
16	6	1501	143.9517	143.6926	144.2105	1.2121	143.7053	8.4212
16	7	1501	113.7375	113.6621	113.813	0.95766	113.6896	3.3035
16	8	1501	198.4534	198.3125	198.5944	1.671	198.3491	6.4353
16	4	1801	113.4815	113.443	113.5393	0.95552	113.4263	3.5401
16	5	1801	36.0698	38.3197	32.4011	0.30457	28.618	21.959
16	6	1801	194.979	194.9019	195.0945	1.6417	194.8773	6.2968
16	7	1801	35.4897	38.4603	30.493	0.29943	27.7047	22.184
16	8	1801	181.0007	180.9445	181.0849	1.524	180.9306	5.0377
17	4	1200	114.7937	114.6781	114.8707	0.96656	114.725	3.9737
17	5	1200	110.7563	110.6739	110.8112	0.93257	110.7029	3.4406
17	6	1200	140.0575	139.9604	140.1222	1.1793	140.0045	3.8542
17	7	1200	127.626	127.5402	127.6831	1.0746	127.5794	3.4474
17	8	1200	171.4596	171.5034	171.4305	1.4436	171.3664	5.653
17	4	1501	31.2318	34.7716	27.2329	0.26354	25.7931	17.6137
17	5	1501	31.7757	34.7174	28.5299	0.26821	26.9939	16.7666
17	6	1501	137.831	137.7619	137.9002	1.1605	137.7683	4.1597
17	7	1501	180.7312	180.316	181.1457	1.5217	180.3773	11.3064
17	8	1501	114.1124	114.047	114.1777	0.96081	114.0516	3.7236
17	4	1801	108.6896	108.6528	108.745	0.91517	108.6376	3.3633
17	5	1801	106.5698	106.4812	106.7026	0.89733	106.4511	5.0283

Table 6.8: Summary of RBF-NARX model construction for identification of rotational speed

# neurons	# delays	# training samples	RMSE$^N_{total}$	RMSE$^N_{training}$	RMSE$^N_{test}$	%RMSE$^N_{total}$	mean (μ_{ae})	Std (σ_{ae})
17	6	1801	115.2259	115.1704	115.3091	0.9702	115.1513	4.1453
17	7	1801	176.9808	176.9117	177.0844	1.4902	176.8843	5.8453
17	8	1801	161.857	161.8074	161.9314	1.3628	161.7737	5.1925
18	4	1200	109.285	109.1958	109.3444	0.92018	109.2308	3.4437
18	5	1200	102.3664	102.29	102.4172	0.86192	102.3211	3.0438
18	6	1200	169.5445	169.3732	169.6585	1.4276	169.4363	6.0576
18	7	1200	126.7108	126.6152	126.7745	1.0669	126.6515	3.8752
18	**8**	**1200**	**134.4747**	**212.6346**	**2.6076**	**1.1263**	**7.0685**	**134.3112**
18	4	1501	99.7465	99.6975	99.7955	0.83987	99.699	3.0786
18	5	1501	43.4613	52.6548	31.6962	0.36598	30.5798	30.8881
18	6	1501	180.9963	180.9068	181.0859	1.524	180.9185	5.3067
18	7	1501	120.5452	120.4738	120.6166	1.015	120.4874	3.7331
18	8	1501	158.1591	157.9758	158.3422	1.3317	158.0073	6.9275
18	4	1801	31.6943	34.3004	27.3204	0.2673	26.1631	17.8921
18	5	1801	107.6138	107.5772	107.6686	0.9061	107.562	3.3388
18	6	1801	125.7658	125.7153	125.8417	1.0589	125.6994	4.0889
18	7	1801	109.3846	109.3492	109.4377	0.92101	109.336	3.2609
18	8	1801	133.667	133.8736	133.3563	1.1254	94.2531	94.7957
19	4	1200	113.3755	113.2606	113.452	0.95463	113.3044	4.0155
19	5	1200	161.2492	161.0849	161.3585	1.3577	161.1482	5.7073
19	6	1200	101.6249	101.5501	101.6747	0.85568	101.5807	2.9954
19	7	1200	163.5428	162.8733	163.9873	1.3771	163.1303	11.61
19	8	1200	161.2447	161.0644	161.3648	1.3577	161.1397	5.8206

Table 6.8: Summary of RBF-NARX model construction for identification of rotational speed

# neurons	# delays	# training samples	RMSE$^N_{total}$	RMSE$^N_{training}$	RMSE$^N_{test}$	%RMSE$^N_{total}$	mean (μ_{ae})	Std (σ_{ae})
19	4	1501	119.4811	119.4204	119.5419	1.006	119.4251	3.6589
19	5	1501	164.5157	164.2498	164.7814	1.3852	164.2581	9.2054
19	6	1501	33.7852	38.1586	28.7505	0.28505	26.9957	20.3177
19	7	1501	183.9956	183.902	184.0891	1.5492	183.908	5.6785
19	8	1501	100.8118	100.743	100.8807	0.84884	100.7555	3.3711
19	4	1801	138.6046	138.5524	138.6829	1.167	138.5301	4.5441
19	5	1801	111.6155	111.5645	111.6919	0.93981	111.5479	3.8832
19	6	1801	116.7497	116.7018	116.8216	0.98303	116.6861	3.8542
19	7	1801	177.7103	177.6361	177.8216	1.4963	177.6176	5.7405
19	8	1801	147.2936	147.2461	147.3648	1.2402	147.2306	4.3075
20	4	1200	119.0366	118.9487	119.0952	1.0023	118.9833	3.5631
20	5	1200	149.2693	149.1581	149.3434	1.2568	149.1983	4.6048
20	6	1200	30.5354	32.0403	29.4901	0.25768	25.7183	16.4642
20	7	1200	128.9061	128.8108	128.9695	1.0854	128.8481	3.8654
20	8	1200	154.0375	153.8927	154.1339	1.297	153.9489	5.2256
20	4	1501	130.4491	130.3829	130.5154	1.0984	130.365	4.6845
20	5	1501	105.9153	105.8617	105.9689	0.8918	105.8641	3.2937
20	6	1501	111.6484	111.5898	111.7071	0.94008	111.5939	3.4901
20	7	1501	134.4554	134.3937	134.5172	1.1321	134.4055	3.6653
20	8	1501	145.4837	145.3953	145.5721	1.225	145.403	4.8458
20	4	1801	121.9545	121.8886	122.0535	1.0269	121.8579	4.8553
20	5	1801	111.0587	111.0208	111.1156	0.93511	111.005	3.4541
20	6	1801	158.8357	158.7808	158.918	1.3374	158.7583	4.9592

Table 6.8: Summary of RBF-NARX model construction for identification of rotational speed

# neurons	# delays	# training samples	RMSE$^{N}_{total}$	RMSE$^{N}_{training}$	RMSE$^{N}_{test}$	%RMSE$^{N}_{total}$	mean (μ_{ae})	Std (σ_{ae})
20	7	1801	159.4777	159.4262	159.555	1.3428	159.4092	4.6751
20	8	1801	151.8963	151.7926	152.0517	1.279	151.7578	6.484

Table 6.9: Summary of RBF-NARX model construction for identification of turbine temperature

# neurons	# delays	# training samples	RMSE$^{T_T}_{total}$	RMSE$^{T_T}_{training}$	RMSE$^{T_T}_{test}$	%RMSE$^{T_T}_{total}$	mean (μ_{ae})	Std (σ_{ae})
10	4	1200	248.0781	179.3072	284.8268	15.7583	153.0082	195.3048
10	5	1200	207.1761	148.416	238.4197	13.2638	175.6342	109.9026
10	6	1200	116.529	106.6732	122.657	9.5312	77.1716	87.3275
10	7	1200	107.7162	99.2078	113.0304	8.6593	80.7307	71.3234
10	8	1200	172.1691	174.8978	170.3266	13.1648	120.1952	123.2898
10	4	1501	147.9648	147.6238	148.3052	9.397	147.6888	9.0341
10	5	1501	210.1373	171.3942	242.7947	13.4563	143.7779	153.2757
10	6	1501	183.362	149.3543	212.0001	11.7609	159.1257	91.123
10	7	1501	191.311	154.2249	222.3129	12.2994	165.1711	96.5481
10	8	1501	116.2038	109.0299	122.9642	9.4428	77.3203	86.7606
10	**4**	**1801**	**178.8614**	**230.875**	**2.4151**	**9.2798**	**47.9622**	**172.3396**
10	5	1801	131.9345	130.7706	133.6623	7.821	61.2848	116.8565
10	6	1801	180.3875	158.4	209.0921	11.5788	156.4552	89.8003
10	7	1801	152.1321	130.5643	179.7081	9.8619	129.5118	79.8311
10	8	1801	123.4252	113.3757	137.1327	9.9988	84.0548	90.3952
11	4	1200	220.7185	159.9259	253.2489	14.0443	192.6668	107.7034
11	5	1200	211.3474	150.6581	243.5286	13.4975	147.5283	151.363
11	6	1200	181.9475	127.4505	210.5694	11.68	157.7301	90.7129

Table 6.9: Summary of RBF-NARX model construction for identification of turbine temperature

# neurons	# delays	# training samples	RMSE$^{T_T}_{total}$	RMSE$^{T_T}_{training}$	RMSE$^{T_T}_{test}$	%RMSE$^{T_T}_{total}$	mean (μ_{ae})	Std (σ_{ae})
11	7	1200	185.3197	129.6805	214.5254	11.8944	160.6223	92.4485
11	8	1200	116.0043	110.1355	119.7551	9.423	79.1754	84.7977
11	4	1501	242.1402	198.9689	278.7254	15.5464	152.3787	188.2138
11	5	1501	197.3852	161.5336	227.6781	12.5992	143.3319	135.7315
11	6	1501	181.4568	147.828	209.7794	11.6385	157.508	90.1138
11	7	1501	160.4448	129.4396	186.3772	10.3228	138.661	80.7331
11	8	1501	101.7118	92.1387	110.464	8.3056	74.2975	69.4751
11	4	1801	266.1683	235.7966	306.1479	16.9956	232.294	129.9641
11	5	1801	193.558	170.0927	224.2131	12.3949	150.5632	121.6567
11	6	1801	179.4307	157.8617	207.6392	11.5045	155.9445	88.7656
11	7	1801	183.0264	159.5606	213.4572	11.779	157.7368	92.8474
11	8	1801	153.4241	133.9736	178.686	9.8792	132.5832	77.2181
12	4	1200	297.9633	214.5184	342.4553	19.0474	259.9294	145.6911
12	5	1200	198.2247	141.1172	228.4844	12.6965	171.3687	99.6449
12	6	1200	178.3149	123.0885	207.0918	11.4621	153.9954	89.9128
12	7	1200	154.5077	106.734	179.4111	9.9457	133.6015	77.6226
12	8	1200	176.8405	120.0556	206.1693	11.3898	152.1847	90.0837
12	4	1501	313.1933	258.3615	359.792	19.9726	273.2148	153.1388
12	5	1501	243.3416	199.3398	280.5475	15.5136	211.392	120.5546
12	6	1501	182.6072	149.0022	210.9421	11.7089	158.6305	90.4681
12	7	1501	176.7276	143.5864	204.5858	11.3462	153.2398	88.0501
12	8	1501	159.1183	127.7437	185.2695	10.2495	137.1953	80.6116
12	4	1801	177.0031	228.4751	2.566	9.491	51.9466	169.2371

Table 6.9: Summary of RBF-NARX model construction for identification of turbine temperature

# neurons	# delays	# training samples	RMSE$^{T_T}_{total}$	RMSE$^{T_T}_{training}$	RMSE$^{T_T}_{test}$	%RMSE$^{T_T}_{total}$	mean (μ_{ae})	Std (σ_{ae})
12	5	1801	204.3356	180.1991	235.9719	13.0841	176.604	102.7988
12	6	1801	182.011	159.8666	210.9274	11.6798	156.6419	92.7048
12	7	1801	166.2136	145.5467	193.1244	10.6858	143.8962	83.2055
12	8	1801	147.6718	128.8098	172.1447	9.5167	127.4851	74.5409
13	4	1200	160.0443	159.9158	160.1299	10.1498	159.9665	4.9917
13	5	1200	196.8365	140.4223	226.7635	12.5784	136.1104	142.2157
13	6	1200	177.3899	124.3245	205.268	11.3868	153.8168	88.3752
13	7	1200	162.7089	113.247	188.5965	10.4612	140.9099	81.3681
13	8	1200	147.2319	99.5911	171.7909	9.5003	126.6637	75.0695
13	4	1501	180.5525	147.0356	208.7732	11.6277	115.2624	138.9972
13	5	1501	199.6514	163.5347	230.188	12.7807	173.7579	98.3469
13	6	1501	176.2766	143.9626	203.54	11.2734	124.2613	125.0511
13	7	1501	161.4472	130.3833	187.4477	10.3845	139.5647	81.1732
13	8	1501	149.9837	120.5348	174.5476	9.669	129.407	75.8343
13	4	1801	364.305	364.0052	364.7545	23.1158	363.9673	15.6851
13	5	1801	195.5261	172.3409	225.9009	12.5118	165.2882	104.4698
13	6	1801	179.164	157.3983	207.5912	11.498	155.4176	89.1502
13	7	1801	161.3369	141.2276	187.5132	10.3754	139.6646	80.781
13	8	1801	149.2598	130.2251	173.962	9.6177	128.911	75.2483
14	4	1200	143.3894	143.2674	143.4706	9.0918	143.3191	4.4908
14	5	1200	242.6513	84.923	305.4598	15.6643	165.5776	177.4097
14	6	1200	189.9998	133.5594	219.6992	12.1875	164.8188	94.5393
14	7	1200	165.3938	115.4255	191.5845	10.6265	143.333	82.5411

Table 6.9: Summary of RBF-NARX model construction for identification of turbine temperature

# neurons	# delays	# training samples	RMSE$_{total}^{T_T}$	RMSE$_{training}^{T_T}$	RMSE$_{test}^{T_T}$	%RMSE$_{total}^{T_T}$	mean (μ_{ae})	Std (σ_{ae})
14	8	1200	148.7356	101.2245	173.3065	9.5924	128.1657	75.4834
14	4	1501	164.4817	134.1277	190.0639	10.5964	124.8976	107.0451
14	5	1501	145.1324	159.0167	129.7602	7.7193	42.0025	138.9447
14	6	1501	172.2062	139.7434	199.4705	11.0469	148.235	87.6579
14	7	1501	163.9222	132.5066	190.2345	10.5418	141.8263	82.2073
14	8	1501	146.3392	117.9054	170.0989	9.4246	126.455	73.6621
14	4	1801	212.9757	186.4941	247.4579	13.84	185.2406	105.1107
14	5	1801	134.7458	134.6747	134.8524	8.5484	134.6626	4.7339
14	6	1801	192.6953	169.6611	222.8415	12.3514	167.3851	95.4817
14	7	1801	166.3764	145.5187	193.5062	10.706	143.8947	83.5329
14	8	1801	147.1616	128.4627	171.44	9.4809	127.1611	74.084
15	4	1200	228.4541	228.2762	228.5726	14.4822	228.3452	7.0537
15	5	1200	225.476	158.2258	260.8305	14.4524	195.462	112.42
15	6	1200	213.0437	150.3132	246.1196	13.6509	184.9246	105.8027
15	7	1200	160.1158	110.2076	186.0815	10.3081	138.2707	80.7495
15	8	1200	149.775	102.5474	174.2771	9.6547	129.2304	75.7228
15	4	1501	215.7758	277.3815	127.1126	10.084	43.3616	211.4092
15	5	1501	162.9787	133.8174	187.6773	10.3713	141.6756	80.5743
15	6	1501	180.8626	147.4978	208.9839	11.5997	149.3284	102.0578
15	7	1501	265.4489	215.1401	307.6638	17.004	229.5758	133.2816
15	8	1501	153.5239	123.6985	178.4472	9.8818	132.6238	77.3467
15	4	1801	159.2979	159.0704	159.6387	10.1151	159.0675	8.5672
15	5	1801	230.6506	230.3683	231.0735	14.6418	230.3447	11.8755

Table 6.9: Summary of RBF-NARX model construction for identification of turbine temperature

# neurons	# delays	# training samples	RMSE$^{T_T}_{total}$	RMSE$^{T_T}_{training}$	RMSE$^{T_T}_{test}$	%RMSE$^{T_T}_{total}$	mean (μ_{ae})	Std (σ_{ae})
15	6	1801	179.4164	157.5194	207.9981	11.5151	155.5938	89.3502
15	7	1801	160.7468	140.222	187.3785	10.353	138.7222	81.2274
15	8	1801	149.7369	130.6232	174.5386	9.6506	129.294	75.5389
16	4	1200	190.8991	190.7492	190.9989	12.1036	190.8069	5.9337
16	5	1200	111.6185	111.4783	111.7117	7.0788	111.5447	4.0576
16	6	1200	164.0801	115.2647	189.758	10.5506	142.4627	81.4177
16	7	1200	199.3492	138.3108	231.2415	12.8013	172.3719	100.1568
16	8	1200	149.4384	102.7163	173.7285	9.6283	129.0739	75.3236
16	4	1501	251.6964	251.4689	251.9238	15.9672	251.5156	9.5384
16	5	1501	175.5114	142.8542	202.9978	11.258	129.2537	118.7536
16	6	1501	170.8703	138.6451	197.9332	10.9865	148.068	85.2934
16	7	1501	167.8862	135.8406	194.7444	10.797	145.3064	84.1081
16	8	1501	154.9198	124.5044	180.2903	9.9802	133.6493	78.3585
16	4	1801	203.0655	202.9921	203.1755	12.8745	202.9681	6.2887
16	5	1801	160.8186	141.0275	186.6238	10.3616	139.6286	79.8036
16	6	1801	142.1869	141.9502	142.5413	9.033	141.9414	8.3525
16	7	1801	172.2232	150.9866	199.9058	11.0627	149.222	86.0004
16	8	1801	150.639	131.2802	175.7358	9.7103	129.9693	76.171
17	4	1200	143.2665	143.1283	143.3586	9.0806	143.1752	5.1147
17	5	1200	198.9403	198.1171	199.4869	12.6323	198.4646	13.7515
17	6	1200	179.0259	125.0522	207.3299	11.4997	155.1831	89.2813
17	7	1200	109.564	109.4738	109.6241	6.947	109.5113	3.3979
17	8	1200	138.2809	93.1241	161.5056	8.9352	118.9004	70.6113

Table 6.9: Summary of RBF-NARX model construction for identification of turbine temperature

# neurons	# delays	# training samples	RMSE$^{T_T}_{total}$	RMSE$^{T_T}_{training}$	RMSE$^{T_T}_{test}$	%RMSE$^{T_T}_{total}$	mean (μ_{ae})	Std (σ_{ae})
17	4	1501	114.1674	114.0506	114.2842	7.2593	114.075	4.5929
17	5	1501	243.8794	197.8934	282.4998	15.6403	211.2356	121.9084
17	6	1501	172.153	139.9335	199.2451	11.0673	149.0685	86.1261
17	7	1501	174.8686	141.0802	203.1295	11.2458	151.077	88.0758
17	8	1501	151.9932	122.3192	176.7692	9.7944	131.254	76.657
17	4	1801	265.1592	265.0653	265.3002	16.8127	265.0339	8.1533
17	5	1801	144.9792	144.7145	145.3755	9.211	144.6921	9.1199
17	6	1801	201.5678	177.5594	233.0034	12.9157	175.231	99.6343
17	7	1801	140.7777	121.9195	165.0863	9.0396	120.2346	73.2378
17	8	1801	143.714	125.1645	167.7474	9.272	123.9534	72.7394
18	4	1200	120.9347	120.8214	121.0101	7.6649	120.8622	4.185
18	5	1200	201.6752	201.4751	201.8083	12.7875	201.5696	6.5265
18	6	1200	217.3619	152.3322	251.5247	13.8832	188.1734	108.8159
18	7	1200	191.6773	134.7552	221.6321	12.2958	166.3042	95.3212
18	8	1200	151.8592	103.1033	177.042	9.7927	130.7558	77.2406
18	4	1501	106.229	106.1014	106.3567	6.7455	106.1607	3.8103
18	5	1501	143.638	143.5612	143.7147	9.1078	143.5686	4.4649
18	6	1501	154.0754	125.3085	178.274	9.8866	110.2697	107.6276
18	7	1501	166.8522	135.0458	193.5158	10.7219	144.407	83.5979
18	8	1501	131.6182	104.479	154.062	8.5257	112.9845	67.523
18	4	1801	151.417	151.3618	151.4998	9.5846	151.2192	7.738
18	5	1801	244.4467	213.1544	285.0355	15.5842	195.0991	147.3016
18	6	1801	163.518	143.2958	189.8685	10.5107	131.9807	96.5524

Table 6.9: Summary of RBF-NARX model construction for identification of turbine temperature

# neurons	# delays	# training samples	RMSE$^{T_T}_{total}$	RMSE$^{T_T}_{training}$	RMSE$^{T_T}_{test}$	%RMSE$^{T_T}_{total}$	mean (μ_{ae})	Std (σ_{ae})
18	7	1801	164.6883	144.2214	191.3405	10.5909	142.606	82.3896
18	8	1801	144.312	125.587	168.5555	9.3131	124.3424	73.2578
19	4	1200	124.8185	124.7166	124.8864	7.9148	124.7584	3.8747
19	5	1200	116.8087	116.6751	116.8976	7.4073	116.7347	4.1578
19	6	1200	258.6026	180.1182	299.6961	16.5827	223.7498	129.6801
19	7	1200	163.7364	110.4903	191.1506	10.5774	140.7091	83.7431
19	8	1200	153.1936	104.6758	178.3381	9.8849	132.1211	77.5519
19	4	1501	135.7304	135.5991	135.8616	8.6062	135.5985	5.9818
19	5	1501	170.5506	170.4608	170.6404	10.8149	170.4692	5.2713
19	6	1501	345.84	282.746	399.1138	22.0843	300.2976	171.5706
19	7	1501	143.298	115.173	166.7592	9.2705	123.7542	72.256
19	8	1501	144.2192	115.5175	168.1043	9.3055	124.24	73.2489
19	4	1801	141.176	141.1338	141.2394	8.9519	141.1076	4.3971
19	5	1801	158.3819	140.0886	182.4264	10.0795	137.7159	78.2381
19	6	1801	169.7877	148.6794	197.2737	10.9525	147.2371	84.5664
19	7	1801	154.7682	135.2971	180.0826	9.9732	133.9595	77.5245
19	8	1801	152.7537	133.1631	178.1581	9.8506	131.8532	77.139
20	4	1200	106.3589	106.2383	106.4391	6.7416	106.2848	3.9681
20	5	1200	121.3176	121.1253	121.4456	7.7018	121.2445	4.2119
20	6	1200	134.3155	88.9803	157.4347	8.7846	115.1983	69.0768
20	7	1200	101.7451	101.6587	101.8026	6.4505	101.696	3.1625
20	8	1200	133.2454	56.0057	165.8133	8.6894	107.5564	78.6639
20	4	1501	183.0812	182.9638	183.1986	11.6064	182.9635	6.5636

Table 6.9: Summary of RBF-NARX model construction for identification of turbine temperature

# neurons	# delays	# training samples	RMSE$^{T_T}_{total}$	RMSE$^{T_T}_{training}$	RMSE$^{T_T}_{test}$	%RMSE$^{T_T}_{total}$	mean (μ_{ae})	Std (σ_{ae})
20	5	1501	176.9939	176.9007	177.0872	11.2232	176.908	5.5133
20	6	1501	145.9677	116.5589	170.3887	9.4693	124.6972	75.8887
20	7	1501	196.2827	158.9363	227.6003	12.61	169.8009	98.4773
20	8	1501	148.6711	119.7122	172.8594	9.5822	128.4488	74.8724
20	4	1801	175.4014	175.3249	175.5162	11.1216	175.2861	6.3613
20	5	1801	221.6818	221.4134	222.0841	14.072	221.3955	11.2665
20	6	1801	150.7779	150.5315	151.147	9.5784	150.5184	8.844
20	7	1801	162.3079	141.8501	188.8986	10.45	140.252	81.7034
20	8	1801	270.9309	270.6958	271.2833	17.1924	270.6922	11.3718
10	4	1200	75.263	72.913	76.7889	4.1533	69.2869	29.3961
10	5	1200	155.2793	153.5769	156.4032	8.4818	154.207	18.2197
10	6	1200	122.867	122.7376	122.9532	6.7205	122.7884	4.3959
10	7	1200	13.4734	15.2044	12.1843	0.75263	9.8925	9.1487
10	8	1200	22.5846	31.8863	13.1326	1.3285	11.7452	19.2934
10	4	1501	66.0248	65.7916	66.2573	3.6329	59.822	27.9438
10	5	1501	126.1031	123.1248	129.0147	7.1544	108.7094	63.9189
10	6	1501	93.3593	89.6891	96.8929	5.2703	82.2458	44.1842
10	7	1501	86.8293	82.4681	90.9844	4.8991	76.7083	40.6905
10	8	1501	95.8502	92.8938	98.72	5.4188	83.7536	46.619
10	4	1801	85.4081	81.5205	90.9313	4.696	80.3869	28.8577
10	5	1801	115.4479	117.7485	111.9063	6.5295	101.1892	55.5877
10	6	1801	165.0215	168.7851	159.2061	9.3289	144.67	79.4028
10	7	1801	163.6824	168.4355	156.2778	9.2552	143.1648	79.3592

Table 6.9: Summary of RBF-NARX model construction for identification of turbine temperature

# neurons	# delays	# training samples	RMSE$^{T_T}_{total}$	RMSE$^{T_T}_{training}$	RMSE$^{T_T}_{test}$	%RMSE$^{T_T}_{total}$	mean (μ_{ae})	Std (σ_{ae})
10	8	1801	120.0939	123.6459	114.5566	6.7916	104.8588	58.5519
11	4	1200	188.364	187.9744	188.6232	10.3011	188.1229	9.5295
11	5	1200	74.3993	75.1104	73.9218	4.1802	66.3091	33.7452
11	6	1200	78.4977	83.4896	74.9873	4.4516	67.3956	40.2524
11	7	1200	191.3645	200.8411	184.7807	10.8268	166.5947	94.1783
11	8	1200	51.4856	53.4922	50.1041	2.9079	44.8358	25.3127
11	4	1501	163.5491	163.251	163.8469	8.944	163.2847	9.297
11	5	1501	98.1482	96.5899	99.6832	5.5735	84.0678	50.661
11	6	1501	215.7239	215.3266	216.1207	11.7954	215.3484	12.7252
11	7	1501	95.122	91.5659	98.5521	5.371	83.7565	45.0969
11	8	1501	75.4676	72.5398	78.2879	4.2608	66.2887	36.0778
11	4	1801	233.055	232.9305	233.2417	12.747	232.8748	9.1643
11	5	1801	112.4904	109.292	117.127	6.2709	103.0009	45.2282
11	6	1801	161.4779	161.1155	162.0203	8.8276	160.9836	12.6277
11	7	1801	109.1645	113.5457	102.2374	6.1887	94.2014	55.1725
11	8	1801	126.0525	130.1258	119.6794	7.1327	109.8405	61.8512
12	4	1200	140.6624	140.5237	140.7547	7.6935	140.5742	4.9801
12	5	1200	116.0673	111.1493	119.2315	6.3221	113.4825	24.3628
12	6	1200	156.9673	165.4299	151.0658	8.8876	136.159	78.1121
12	7	1200	124.3935	138.5776	113.9669	7.0995	103.0884	69.6286
12	8	1200	100.6909	106.1413	96.8893	5.7065	87.0066	50.6888
12	4	1501	108.6603	108.5932	108.7274	5.9437	108.6038	3.5047
12	5	1501	63.4218	63.5902	63.2529	3.5309	56.0428	29.6955

Table 6.9: Summary of RBF-NARX model construction for identification of turbine temperature

# neurons	# delays	# training samples	RMSE$^{T_T}_{total}$	RMSE$^{T_T}_{training}$	RMSE$^{T_T}_{test}$	%RMSE$^{T_T}_{total}$	mean (μ_{ae})	Std (σ_{ae})
12	6	1501	115.8169	115.5856	116.0479	6.594	97.777	62.0845
12	7	1501	94.2649	91.5878	96.8698	5.3389	81.7837	46.8832
12	8	1501	74.3123	72.41	76.1683	4.2032	64.6406	36.6656
12	4	1801	260.6618	260.576	260.7905	14.2548	260.5391	7.9965
12	5	1801	104.8933	100.8564	110.6759	5.795	98.6375	35.6886
12	6	1801	232.7136	232.4245	233.1467	12.7256	232.326	13.4278
12	7	1801	118.0944	125.4911	106.0292	6.7156	99.9576	62.8974
12	8	1801	129.4815	134.3815	121.7579	7.3284	112.3561	64.3657
13	4	1200	195.3688	195.1844	195.4915	10.6855	195.2624	6.4472
13	5	1200	118.2804	113.2926	121.4901	6.5048	112.3531	36.9796
13	6	1200	71.876	70.1018	73.0342	3.951	65.8688	28.7704
13	7	1200	110.4787	117.0547	105.8707	6.2668	94.9364	56.5127
13	8	1200	85.1731	88.8288	82.6477	4.8231	73.8752	42.3971
13	4	1501	171.962	171.8752	172.0488	9.4062	171.8789	5.348
13	5	1501	103.5105	103.4361	103.5849	5.6611	103.4443	3.701
13	6	1501	109.7544	107.2162	112.2368	6.0001	106.4395	26.7748
13	7	1501	242.2456	241.9009	242.5899	13.2491	241.9263	12.4342
13	8	1501	108.4142	107.6971	109.1272	6.15	93.0013	55.7265
13	4	1801	115.2928	115.2483	115.3596	6.3063	115.2342	3.6769
13	5	1801	83.8456	79.8002	89.5749	4.6037	78.2682	30.0745
13	6	1801	181.9527	187.4275	173.412	10.2644	158.8225	88.7967
13	7	1801	185.9137	185.6961	186.2397	10.1661	185.6267	10.3273
13	8	1801	152.366	159.1441	141.5858	8.6365	131.4585	77.0457

Table 6.9: Summary of RBF-NARX model construction for identification of turbine temperature

# neurons	# delays	# training samples	RMSE$^{T_T}_{total}$	RMSE$^{T_T}_{training}$	RMSE$^{T_T}_{test}$	%RMSE$^{T_T}_{total}$	mean (μ_{ae})	Std (σ_{ae})
14	4	1200	180.7673	180.6325	180.857	9.8877	180.6801	5.6147
14	5	1200	75.4158	80.1993	72.0525	4.1889	66.7299	35.1436
14	6	1200	76.5233	74.5947	77.7818	4.2057	70.6501	29.4053
14	7	1200	143.5512	149.6471	139.3416	8.1097	125.6482	69.434
14	8	1200	67.7342	70.4228	65.882	3.8324	58.9656	33.337
14	4	1501	152.0941	152.0167	152.1716	8.3186	152.0201	4.745
14	5	1501	88.963	85.6909	92.121	4.8852	83.2991	31.2409
14	6	1501	73.258	72.616	73.8949	4.0356	66.8802	29.9012
14	7	1501	135.8022	130.8733	140.5615	7.5949	122.0812	59.4946
14	8	1501	123.7952	121.9249	125.639	7.0145	106.9044	62.4341
14	4	1801	147.9563	147.8927	148.0518	8.0918	147.8656	5.182
14	5	1801	112.0852	112.037	112.1575	6.1295	112.0148	3.9746
14	6	1801	76.3416	73.4146	80.535	4.2159	70.2029	29.9981
14	7	1801	168.4348	167.8746	169.272	9.2046	167.6496	16.2466
14	8	1801	216.1583	225.821	200.7854	12.2265	187.7567	107.1244
15	4	1200	163.7234	163.6005	163.8053	8.9556	163.6436	5.1129
15	5	1200	329.9522	329.5097	330.2466	18.0454	329.6858	13.2581
15	6	1200	82.7747	80.0486	84.5422	4.5716	76.6884	31.1586
15	7	1200	103.154	99.0631	105.792	5.6426	99.487	27.264
15	8	1200	99.9741	105.4351	96.1634	5.6573	86.5782	49.9987
15	4	1501	269.969	269.8335	270.1045	14.7653	269.8422	8.2736
15	5	1501	172.4222	172.3352	172.5091	9.4316	172.3402	5.3179
15	6	1501	76.0029	73.1782	78.728	4.1932	70.1689	29.207

Table 6.9: Summary of RBF-NARX model construction for identification of turbine temperature

# neurons	# delays	# training samples	RMSE$^{T_T}_{total}$	RMSE$^{T_T}_{training}$	RMSE$^{T_T}_{test}$	%RMSE$^{T_T}_{total}$	mean (μ_{ae})	Std (σ_{ae})
15	7	1501	88.2302	85.5833	90.8018	4.8468	82.988	29.9643
15	8	1501	173.5299	170.5947	176.4182	9.8003	151.2111	85.1483
15	4	1801	145.1103	145.047	145.2054	7.9367	145.0176	5.1871
15	5	1801	318.7874	318.5947	319.0765	17.4343	318.5217	13.015
15	6	1801	78.6741	75.0771	83.7832	4.3353	72.9309	29.5126
15	7	1801	84.6347	80.9734	89.8502	4.6604	78.4778	31.6957
15	8	1801	117.8525	117.1005	118.9721	6.4344	116.7983	15.7304
16	4	1200	206.0819	205.8089	206.2636	11.2706	205.9154	8.2844
16	5	1200	145.8192	145.7106	145.8914	7.976	145.7488	4.5302
16	6	1200	89.8905	86.8022	91.8907	4.9569	83.8622	32.3696
16	7	1200	166.2082	166.0699	166.3003	9.0922	166.1264	5.2152
16	8	1200	95.1256	100.3104	91.5081	5.3854	82.2592	47.7814
16	4	1501	327.4169	326.93	327.9033	17.9041	326.9518	17.4477
16	5	1501	167.3758	167.2913	167.4602	9.1564	167.2956	5.1799
16	6	1501	165.3899	165.0985	165.681	9.044	165.1301	9.2694
16	7	1501	80.1312	77.3356	82.8343	4.4538	72.6451	33.8244
16	8	1501	212.9614	212.0284	213.8908	11.6377	212.0636	19.5375
16	4	1801	166.2228	166.1468	166.3368	9.0906	166.1165	5.944
16	5	1801	106.2498	106.2109	106.3083	5.8115	106.1971	3.3461
16	6	1801	67.0757	64.7288	70.4514	3.7051	60.3785	29.2211
16	7	1801	88.667	85.1697	93.6712	4.8851	82.7182	31.9355
16	8	1801	153.7005	151.0812	157.5499	8.5341	141.8058	59.2971
17	4	1200	144.3074	144.1639	144.4029	7.8928	144.2218	4.97

Table 6.9: Summary of RBF-NARX model construction for identification of turbine temperature

# neurons	# delays	# training samples	RMSE$^{T_T}_{total}$	RMSE$^{T_T}_{training}$	RMSE$^{T_T}_{test}$	%RMSE$^{T_T}_{total}$	mean (μ_{ae})	Std (σ_{ae})
17	5	1200	127.2684	127.1435	127.3516	6.9605	127.1902	4.4621
17	6	1200	101.8417	101.7405	101.9091	5.5709	101.7777	3.611
17	7	1200	116.8393	116.0062	117.3911	6.5287	107.0782	46.7592
17	8	1200	97.7279	94.2515	99.977	5.3759	92.3643	31.936
17	4	1501	333.3508	333.0754	333.6263	18.2307	333.0895	13.2003
17	5	1501	121.0593	120.9959	121.1228	6.6214	121.0004	3.7779
17	6	1501	113.6087	111.4546	115.7241	6.1928	110.9945	24.2354
17	7	1501	79.2788	76.0168	82.4139	4.3862	72.9622	31.0157
17	8	1501	78.7968	76.8145	80.7317	4.3407	72.99	29.6933
17	4	1801	198.377	198.2867	198.5123	10.85	198.2478	7.1599
17	5	1801	115.8266	115.7769	115.9011	6.3356	115.7539	4.1042
17	6	1801	312.072	311.8909	312.3436	17.0675	311.8201	12.5391
17	7	1801	81.0421	77.417	86.1971	4.478	74.571	31.7383
17	8	1801	97.3345	94.0969	102.001	5.3652	90.9517	34.6728
18	4	1200	154.5424	154.4256	154.6201	8.4531	154.4695	4.7458
18	5	1200	245.8031	245.5166	245.9938	13.4453	245.6345	9.1043
18	6	1200	319.0891	318.6504	319.3811	17.4524	318.8281	12.9072
18	7	1200	168.8379	168.6988	168.9304	9.2343	168.755	5.2895
18	8	1200	106.8291	102.5336	109.5978	5.8341	103.3303	27.121
18	4	1501	150.9882	144.8087	156.9285	8.5609	131.7746	73.7205
18	5	1501	318.2245	317.8918	318.5571	17.4054	317.9272	13.7543
18	6	1501	152.8787	152.7814	152.9759	8.3619	152.7975	4.9804
18	7	1501	72.334	71.401	73.2556	3.9708	66.3955	28.7074

Table 6.9: Summary of RBF-NARX model construction for identification of turbine temperature

# neurons	# delays	# training samples	RMSE$^{T_T}_{total}$	RMSE$^{T_T}_{training}$	RMSE$^{T_T}_{test}$	%RMSE$^{T_T}_{total}$	mean (μ_{ae})	Std (σ_{ae})
18	8	1501	89.2868	86.0931	92.3722	4.919	83.7634	30.9215
18	4	1801	203.2992	203.2134	203.4279	11.1198	203.0464	10.1367
18	5	1801	305.4033	305.2101	305.6931	16.7038	305.1255	13.0274
18	6	1801	77.0923	77.9437	75.7964	4.2977	67.2325	37.7291
18	7	1801	95.2301	91.4629	100.6196	5.2332	89.9476	31.2814
18	8	1801	82.8859	79.9299	87.1344	4.5651	77.121	30.3764
19	4	1200	203.0169	202.8579	203.1228	11.1043	202.9167	6.3811
19	5	1200	112.4697	112.3793	112.53	6.1516	112.4146	3.5229
19	6	1200	140.2444	140.1401	140.3139	7.6714	140.1774	4.3369
19	7	1200	355.6418	354.6987	356.2687	19.4473	355.0636	20.2748
19	8	1200	91.2141	89.1937	92.5358	5.0432	84.7102	33.8318
19	4	1501	141.6924	141.6043	141.7806	7.7499	141.6162	4.6491
19	5	1501	348.6103	348.2395	348.981	19.0662	348.2722	15.3534
19	6	1501	121.4029	121.3407	121.4651	6.6401	121.3441	3.7787
19	7	1501	171.2182	170.9769	171.4593	9.3648	171.0071	8.4993
19	8	1501	73.87	72.887	74.8407	4.0737	67.6725	29.6226
19	4	1801	168.0132	167.9403	168.1226	9.1893	167.9127	5.8117
19	5	1801	158.6007	158.4484	158.829	8.6744	158.3896	8.1813
19	6	1801	189.0908	189.0221	189.1937	10.3432	188.9975	5.9389
19	7	1801	168.3026	168.1474	168.5352	9.2034	168.0832	8.5924
19	8	1801	103.862	100.3007	108.9887	5.7256	97.9948	34.4199
20	4	1200	113.3846	113.1547	113.5375	6.2	113.2568	5.3826
20	5	1200	129.3069	129.2056	129.3743	7.073	129.2437	4.0445

Table 6.9: Summary of RBF-NARX model construction for identification of turbine temperature

# neurons	# delays	# training samples	RMSE$^{T_T}_{total}$	RMSE$^{T_T}_{training}$	RMSE$^{T_T}_{test}$	%RMSE$^{T_T}_{total}$	mean (μ_{ae})	Std (σ_{ae})
20	6	1200	366.0749	365.6302	366.3709	20.0232	365.8011	14.1584
20	7	1200	120.268	120.1509	120.346	6.5781	120.197	4.1323
20	8	1200	112.288	108.1027	114.9921	6.1543	108.2228	29.945
20	4	1501	118.412	119.2438	117.5739	6.4856	117.7228	12.7593
20	5	1501	121.2861	121.203	121.3692	6.6342	121.2084	4.3429
20	6	1501	174.5369	174.4326	174.6412	9.5466	174.4489	5.5426
20	7	1501	163.5183	163.4196	163.6171	8.9438	163.4355	5.2048
20	8	1501	149.2317	148.9954	149.4677	8.1611	149.0276	7.8036
20	4	1801	269.0185	268.9439	269.1305	14.7141	268.9265	7.0397
20	5	1801	191.0713	190.9919	191.1903	10.4501	190.9667	6.3226
20	6	1801	186.1256	186.0559	186.2301	10.1793	186.0326	5.8823
20	7	1801	118.6237	118.5738	118.6986	6.4887	118.5553	4.0306
20	8	1801	144.507	144.4525	144.5888	7.9045	144.4348	4.5689

Table 6.10: Summary of RBF-NARX model construction for identification of turbine pressure

# neurons	# delays	# training samples	RMSE$^{P_T}_{total}$	RMSE$^{P_T}_{training}$	RMSE$^{P_T}_{test}$	%RMSE$^{P_T}_{total}$	Mean (μ_{ae})	Std(σ_{ae})
10	4	1200	0.17341	0.15102	0.18684	4.0658	0.15852	0.070299
10	5	1200	0.28252	0.25483	0.29956	8.0653	0.21092	0.188
10	6	1200	0.40855	0.37653	0.42856	11.3677	0.3273	0.24456
10	7	1200	0.40959	0.37842	0.42911	11.3902	0.32791	0.24548
10	8	1200	0.41573	0.38502	0.43499	11.5348	0.33392	0.24769
10	4	1501	0.16658	0.14323	0.18706	4.7807	0.12635	0.10858

Table 6.10: Summary of RBF-NARX model construction for identification of turbine pressure

# neurons	# delays	# training samples	RMSE$^{P_T}_{total}$	RMSE$^{P_T}_{training}$	RMSE$^{P_T}_{test}$	%RMSE$^{P_T}_{total}$	Mean (μ_{ae})	Std(σ_{ae})
10	5	1501	0.15222	0.12974	0.1718	4.2867	0.11862	0.095417
10	6	1501	0.41334	0.39033	0.43514	11.4875	0.33179	0.24654
10	7	1501	0.41563	0.3929	0.4372	11.5311	0.3338	0.24769
10	8	1501	0.40801	0.38742	0.42761	11.2828	0.32686	0.24424
10	4	1801	0.15822	0.14295	0.17871	4.0204	0.13973	0.074252
10	5	1801	0.15052	0.13281	0.17374	4.0655	0.12921	0.077221
10	6	1801	0.35217	0.32356	0.39121	9.8711	0.27698	0.21754
10	7	1801	0.41857	0.38637	0.46271	11.6143	0.33577	0.24997
10	8	1801	0.39683	0.36582	0.43928	11.0358	0.31658	0.23931
11	4	1200	0.16607	0.13992	0.18142	4.7667	0.12687	0.10719
11	5	1200	0.14777	0.12949	0.15879	4.0298	0.11894	0.087706
11	6	1200	0.40936	0.37682	0.42968	11.39	0.32858	0.24419
11	7	1200	0.2243	0.19842	0.23999	6.5191	0.15357	0.16351
11	8	1200	0.41328	0.38318	0.43218	11.4588	0.33078	0.2478
11	4	1501	0.1613	0.14108	0.17927	3.8098	0.14582	0.068969
11	5	1501	0.29467	0.27299	0.31488	8.3818	0.22182	0.19401
11	6	1501	0.29503	0.27377	0.31488	8.3781	0.22268	0.19357
11	7	1501	0.40803	0.3859	0.42903	11.3181	0.32712	0.24392
11	8	1501	0.42192	0.39865	0.44399	11.72	0.33909	0.25112
11	4	1801	0.1602	0.14263	0.18345	4.2136	0.14096	0.07615
11	5	1801	0.15634	0.13913	0.17908	4.1112	0.13741	0.074573
11	6	1801	0.42399	0.39093	0.46925	11.7806	0.34006	0.25327
11	7	1801	0.41666	0.38565	0.45928	11.5441	0.33409	0.24901

Table 6.10: Summary of RBF-NARX model construction for identification of turbine pressure

# neurons	# delays	# training samples	RMSE$^{P_T}_{total}$	RMSE$^{P_T}_{training}$	RMSE$^{P_T}_{test}$	%RMSE$^{P_T}_{total}$	Mean (μ_{ae})	Std(σ_{ae})
11	8	1801	0.41358	0.38228	0.45654	11.4675	0.33205	0.24659
12	4	1200	0.17937	0.15471	0.19407	5.2697	0.11871	0.13449
12	5	1200	0.15691	0.13247	0.17127	4.1181	0.13841	0.07393
12	6	1200	0.1572	0.13413	0.17085	4.183	0.136	0.078853
12	7	1200	0.35646	0.32787	0.3743	9.9654	0.28065	0.21982
12	8	1200	0.15295	0.12988	0.16655	3.9215	0.13616	0.06968
12	4	1501	0.20898	0.18956	0.22675	6.0481	0.13986	0.1553
12	5	1501	0.20107	0.18363	0.21712	4.3573	0.17846	0.092633
12	6	1501	0.15412	0.13133	0.17396	4.1141	0.13277	0.078268
12	7	1501	0.1664	0.14522	0.18518	4.1044	0.15103	0.069867
12	8	1501	0.40551	0.38233	0.42745	11.266	0.32475	0.24289
12	4	1801	0.29504	0.28998	0.30247	8.034	0.19086	0.22503
12	5	1801	0.16163	0.14548	0.18321	4.109	0.14538	0.070651
12	6	1801	0.2453	0.23398	0.26138	5.1353	0.21535	0.11749
12	7	1801	0.36721	0.33827	0.4068	10.2438	0.29112	0.22385
12	8	1801	0.4098	0.37871	0.45247	11.3674	0.32884	0.24459
13	4	1200	0.27449	0.30972	0.24827	7.2799	0.16384	0.22027
13	5	1200	0.17281	0.15093	0.18597	4.0403	0.15764	0.070827
13	6	1200	0.21746	0.1964	0.23043	4.6786	0.19538	0.0955
13	7	1200	0.1722	0.14998	0.18554	4.037	0.15744	0.069779
13	8	1200	0.41879	0.38723	0.43856	11.6118	0.33589	0.25016
13	4	1501	0.18842	0.16625	0.20825	5.4767	0.13224	0.13423
13	5	1501	0.20297	0.18336	0.22085	5.8225	0.14432	0.14274

Table 6.10: Summary of RBF-NARX model construction for identification of turbine pressure

# neurons	# delays	# training samples	RMSE$^{P_T}_{total}$	RMSE$^{P_T}_{training}$	RMSE$^{P_T}_{test}$	%RMSE$^{P_T}_{total}$	Mean (μ_{ae})	Std(σ_{ae})
13	6	1501	0.20229	0.18269	0.22016	4.4097	0.18341	0.085346
13	7	1501	0.21975	0.2134	0.22593	4.6072	0.18175	0.12354
13	8	1501	0.41912	0.3968	0.44032	11.6104	0.33579	0.25085
13	4	1801	0.1805	0.16068	0.2067	5.091	0.14107	0.11261
13	5	1801	0.2038	0.1821	0.2326	5.9328	0.13829	0.14973
13	6	1801	0.32965	0.30192	0.36737	9.2896	0.25564	0.20816
13	7	1801	0.22104	0.20887	0.23815	4.7029	0.19758	0.099132
13	8	1801	0.40903	0.37806	0.45153	11.3617	0.32803	0.24438
14	4	1200	0.19045	0.16646	0.20488	5.5674	0.12782	0.14121
14	5	1200	0.1736	0.14828	0.18858	5.0556	0.12448	0.12102
14	6	1200	0.15847	0.13322	0.17327	4.4575	0.12731	0.094392
14	7	1200	0.17537	0.16215	0.18365	3.9319	0.15535	0.081388
14	8	1200	0.16292	0.13831	0.17743	4.7045	0.12093	0.10919
14	4	1501	0.15872	0.1343	0.17986	4.4875	0.12614	0.096356
14	5	1501	0.16651	0.14225	0.18767	4.4331	0.14598	0.080102
14	6	1501	0.16312	0.14041	0.18305	3.9469	0.14849	0.067539
14	7	1501	0.15565	0.13251	0.17578	4.1328	0.13565	0.076332
14	8	1501	0.39992	0.3771	0.42153	11.1172	0.31894	0.24132
14	4	1801	0.20918	0.18922	0.23599	5.9837	0.1489	0.14694
14	5	1801	0.15576	0.13962	0.17725	3.9694	0.13917	0.069958
14	6	1801	0.19116	0.17782	0.20959	4.3266	0.1744	0.078288
14	7	1801	0.1634	0.14864	0.18333	3.9848	0.14819	0.06886
14	8	1801	0.41607	0.38513	0.45861	11.526	0.33453	0.24743

Table 6.10: Summary of RBF-NARX model construction for identification of turbine pressure

# neurons	# delays	# training samples	RMSE$_{total}^{P_T}$	RMSE$_{training}^{P_T}$	RMSE$_{test}^{P_T}$	%RMSE$_{total}^{P_T}$	Mean (μ_{ae})	Std(σ_{ae})
15	4	1200	0.20997	0.19955	0.21663	5.9195	0.14395	0.15288
15	5	1200	0.16738	0.14225	0.18222	4.4499	0.14664	0.08073
15	6	1200	0.15993	0.13413	0.17502	4.1837	0.1417	0.074164
15	7	1200	0.15588	0.13756	0.16698	3.9527	0.13725	0.073911
15	8	1200	0.19112	0.16974	0.20413	4.2836	0.17481	0.077283
15	4	1501	0.16466	0.1406	0.18565	4.1592	0.14918	0.069729
15	5	1501	0.15414	0.12993	0.17505	4.1733	0.1319	0.079782
15	6	1501	0.18321	0.16135	0.20274	4.1509	0.16798	0.073153
15	7	1501	0.20674	0.18737	0.22445	4.4743	0.18681	0.088576
15	8	1501	0.23053	0.21386	0.24609	4.8671	0.20426	0.10689
15	4	1801	0.2583	0.23681	0.28756	7.4048	0.17101	0.19361
15	5	1801	0.20657	0.18812	0.23153	5.9167	0.14177	0.15027
15	6	1801	0.2291	0.22894	0.22934	6.0948	0.1549	0.16882
15	7	1801	0.18331	0.16895	0.20296	4.2106	0.1682	0.072884
15	8	1801	0.24153	0.23153	0.2558	5.0505	0.2118	0.11611
16	4	1200	0.17345	0.14945	0.18774	4.8011	0.14361	0.097287
16	5	1200	0.17208	0.14872	0.18602	4.7858	0.14067	0.099132
16	6	1200	0.15316	0.129	0.16734	4.2878	0.12412	0.089753
16	7	1200	0.16309	0.13763	0.17804	4.7181	0.12046	0.10996
16	8	1200	0.33429	0.31564	0.34616	6.8605	0.28845	0.16898
16	4	1501	0.17346	0.15012	0.19402	5.0495	0.12527	0.12
16	5	1501	0.25953	0.25131	0.2675	7.2733	0.16892	0.19706
16	6	1501	0.1759	0.15082	0.19785	4.2955	0.1618	0.069027

Table 6.10: Summary of RBF-NARX model construction for identification of turbine pressure

# neurons	# delays	# training samples	RMSE$_{total}^{P_T}$	RMSE$_{training}^{P_T}$	RMSE$_{test}^{P_T}$	%RMSE$_{total}^{P_T}$	Mean (μ_{ae})	Std(σ_{ae})
16	7	1501	0.23676	0.21796	0.2542	4.9863	0.21003	0.10931
16	8	1501	0.18498	0.16706	0.20132	4.1458	0.16786	0.077743
16	4	1801	0.18945	0.17408	0.21042	5.3738	0.1326	0.13533
16	5	1801	1.4345	1.4319	1.4384	33.3405	1.4314	0.094621
16	6	1801	0.16698	0.14913	0.19066	4.3179	0.14977	0.073855
16	7	1801	0.18564	0.17258	0.20367	4.2731	0.16948	0.075752
16	8	1801	0.31558	0.30654	0.32869	6.4683	0.27113	0.16151
17	4	1200	0.21012	0.18757	0.22389	6.095	0.14106	0.15576
17	5	1200	0.18371	0.16095	0.19743	5.2533	0.13849	0.12073
17	6	1200	0.19627	0.18173	0.20539	5.5872	0.13569	0.14183
17	7	1200	0.20198	0.18787	0.21086	5.7554	0.14015	0.14548
17	8	1200	0.17304	0.15081	0.18639	4.0441	0.15856	0.069309
17	4	1501	1.4664	1.4651	1.4678	34.0343	1.465	0.064273
17	5	1501	1.413	1.4096	1.4165	32.8381	1.4101	0.090598
17	6	1501	0.16192	0.13719	0.18337	4.3214	0.14101	0.079615
17	7	1501	0.16695	0.1442	0.18696	4.0278	0.15258	0.067764
17	8	1501	0.26526	0.24838	0.28114	5.5073	0.2317	0.12917
17	4	1801	0.18678	0.16756	0.21237	5.2777	0.14369	0.11935
17	5	1801	0.17127	0.15133	0.19746	4.7039	0.14399	0.092763
17	6	1801	0.18288	0.16335	0.20879	5.2071	0.13924	0.11858
17	7	1801	0.18968	0.17778	0.20626	4.2043	0.17126	0.081565
17	8	1801	0.19754	0.18497	0.21504	4.3971	0.18106	0.079008
18	4	1200	0.24963	0.25246	0.24773	6.9408	0.15765	0.19358

Table 6.10: Summary of RBF-NARX model construction for identification of turbine pressure

# neurons	# delays	# training samples	RMSE$_{total}^{P_T}$	RMSE$_{training}^{P_T}$	RMSE$_{test}^{P_T}$	%RMSE$_{total}^{P_T}$	Mean (μ_{ae})	Std(σ_{ae})
18	5	1200	0.19701	0.17252	0.21176	5.726	0.13885	0.13978
18	6	1200	0.2047	0.18515	0.21675	4.452	0.1851	0.087419
18	7	1200	0.16083	0.13579	0.17554	4.4744	0.132	0.091903
18	8	1200	0.19766	0.18159	0.20768	4.2985	0.17671	0.088573
18	4	1501	1.4694	1.4668	1.472	34.1376	1.4673	0.078111
18	5	1501	0.67461	0.60113	0.74088	19.6023	0.3926	0.54869
18	6	1501	0.21023	0.18781	0.23049	4.5629	0.19169	0.086345
18	7	1501	0.15741	0.13293	0.17857	4.325	0.13192	0.085876
18	8	1501	0.16257	0.14042	0.18206	4.0461	0.14689	0.069685
18	4	1801	1.4733	1.4716	1.4759	34.1807	1.4701	0.098081
18	5	1801	0.22554	0.20514	0.2531	6.4925	0.14753	0.17063
18	6	1801	0.16887	0.14847	0.19553	4.7782	0.13343	0.10352
18	7	1801	0.16357	0.14457	0.18852	4.4953	0.13752	0.088572
18	8	1801	0.1747	0.16187	0.19235	4.0297	0.15923	0.0719
19	4	1200	1.4287	1.4235	1.4321	33.219	1.4258	0.089758
19	5	1200	0.1927	0.1714	0.20567	5.5264	0.14045	0.13195
19	6	1200	0.25968	0.27367	0.24992	7.0684	0.16272	0.20241
19	7	1200	0.16022	0.13563	0.17469	4.538	0.12688	0.097847
19	8	1200	0.20522	0.18778	0.21606	4.4081	0.18294	0.093013
19	4	1501	1.443	1.4416	1.4444	33.4975	1.4417	0.061913
19	5	1501	0.14435	0.12168	0.16393	3.9774	0.11955	0.08092
19	6	1501	0.19762	0.17263	0.2198	4.4995	0.18327	0.073943
19	7	1501	0.17227	0.15038	0.19169	4.0776	0.15847	0.067564

Table 6.10: Summary of RBF-NARX model construction for identification of turbine pressure

# neurons	# delays	# training samples	RMSE$^{P_T}_{total}$	RMSE$^{P_T}_{training}$	RMSE$^{P_T}_{test}$	%RMSE$^{P_T}_{total}$	Mean (μ_{ae})	Std(σ_{ae})
19	8	1501	0.27264	0.25564	0.28864	5.6491	0.23612	0.13633
19	4	1801	1.4764	1.4751	1.4784	34.2772	1.4745	0.076201
19	5	1801	0.18392	0.1672	0.20649	4.5022	0.16967	0.070988
19	6	1801	0.18663	0.16721	0.21247	5.3202	0.13986	0.12359
19	7	1801	0.18095	0.16398	0.20379	4.4519	0.16669	0.070418
19	8	1801	0.17096	0.1539	0.19377	4.1979	0.15691	0.067896
20	4	1200	0.18305	0.1669	0.19306	4.0771	0.15857	0.09146
20	5	1200	0.18895	0.16622	0.20268	5.4662	0.13378	0.13345
20	6	1200	0.18663	0.16441	0.20007	5.1916	0.15096	0.10975
20	7	1200	0.30882	0.29634	0.31687	6.3757	0.26047	0.16594
20	8	1200	0.19195	0.18209	0.19824	5.3289	0.14366	0.12733
20	4	1501	1.447	1.4453	1.4486	33.5804	1.4453	0.069737
20	5	1501	0.17864	0.15907	0.19629	4.9635	0.14234	0.10796
20	6	1501	0.18077	0.15644	0.2022	4.3436	0.16741	0.068216
20	7	1501	0.19983	0.19119	0.20811	5.4617	0.14902	0.13315
20	8	1501	0.23403	0.21624	0.25057	4.9404	0.20818	0.10694
20	4	1801	1.4135	1.4126	1.4149	32.8144	1.4121	0.063044
20	5	1801	0.22907	0.20709	0.25858	6.6167	0.15354	0.17002
20	6	1801	0.18464	0.16298	0.21305	5.3652	0.13155	0.12958
20	7	1801	0.16689	0.14681	0.19314	4.6352	0.13725	0.094949
20	8	1801	0.20249	0.1898	0.22017	4.4921	0.18499	0.082353
10	4	1200	0.04206	0.041372	0.042512	0.88469	0.036605	0.020719
10	5	1200	0.028388	0.038265	0.019162	0.61667	0.016816	0.022874

Table 6.10: Summary of RBF-NARX model construction for identification of turbine pressure

# neurons	# delays	# training samples	RMSE$^{P_T}_{total}$	RMSE$^{P_T}_{training}$	RMSE$^{P_T}_{test}$	%RMSE$^{P_T}_{total}$	Mean (μ_{ae})	Std(σ_{ae})
10	6	1200	0.11497	0.13096	0.10294	2.4936	0.091399	0.069754
10	7	1200	0.073515	0.089074	0.060983	1.6013	0.054738	0.049082
10	8	1200	0.094749	0.1096	0.083398	2.0586	0.073986	0.059199
10	4	1501	0.03267	0.030641	0.034581	0.68331	0.027145	0.018182
10	5	1501	0.047137	0.054334	0.038616	1.0301	0.032015	0.034603
10	6	1501	0.047366	0.044367	0.050188	1.0006	0.041462	0.022904
10	7	1501	0.047904	0.054466	0.040282	1.0461	0.033841	0.033912
10	8	1501	0.088548	0.097417	0.078678	1.9289	0.066065	0.058968
10	4	1801	0.030375	0.029691	0.031373	0.63827	0.024336	0.018179
10	5	1801	0.065731	0.066521	0.064526	1.3976	0.057415	0.032005
10	**6**	**1801**	**0.026215**	**0.030958**	**0.01674**	**0.56995**	**0.015856**	**0.02088**
10	7	1801	0.036441	0.04369	0.021358	0.7974	0.022423	0.028731
10	8	1801	0.04342	0.052328	0.024601	0.95268	0.027342	0.033735
11	4	1200	1.4267	1.4251	1.4277	29.5852	1.4257	0.051337
11	5	1200	0.030405	0.029495	0.030997	0.63677	0.025548	0.016489
11	6	1200	0.040449	0.050807	0.031723	0.88388	0.027711	0.02947
11	7	1200	0.12818	0.14652	0.11433	2.7813	0.10118	0.078712
11	8	1200	0.034697	0.046994	0.023121	0.76037	0.021102	0.027548
11	4	1501	0.071209	0.067685	0.074568	1.5146	0.062443	0.034233
11	5	1501	0.072483	0.069072	0.075743	1.5416	0.063722	0.034551
11	6	1501	0.064206	0.063003	0.065387	1.3644	0.055589	0.032133
11	7	1501	0.043092	0.042758	0.043423	0.91498	0.036079	0.023567
11	8	1501	0.11623	0.12267	0.1094	2.5246	0.091222	0.072033

Table 6.10: Summary of RBF-NARX model construction for identification of turbine pressure

# neurons	# delays	# training samples	RMSE$^{P_T}_{total}$	RMSE$^{P_T}_{training}$	RMSE$^{P_T}_{test}$	%RMSE$^{P_T}_{total}$	Mean (μ_{ae})	Std(σ_{ae})
11	4	1801	0.066314	0.066005	0.066776	1.4089	0.058451	0.031326
11	5	1801	0.043722	0.042411	0.04562	0.9219	0.038088	0.021474
11	6	1801	0.020864	0.021939	0.019137	0.44107	0.015397	0.014083
11	7	1801	0.03452	0.041366	0.020296	0.75521	0.021121	0.027309
11	8	1801	0.05938	0.06966	0.039181	1.2987	0.041333	0.042641
12	4	1200	0.032625	0.044754	0.020954	0.70981	0.018091	0.027155
12	5	1200	0.035273	0.036749	0.034255	0.74349	0.028822	0.020338
12	6	1200	0.065915	0.06767	0.06472	1.4006	0.057841	0.031617
12	7	1200	0.05134	0.065687	0.038951	1.1226	0.034657	0.037885
12	8	1200	0.045565	0.057994	0.034907	0.9965	0.030973	0.033424
12	4	1501	0.023214	0.022812	0.023609	0.48382	0.018241	0.014361
12	5	1501	0.02834	0.027496	0.02916	0.59226	0.023181	0.016307
12	6	1501	0.024525	0.026245	0.022673	0.52858	0.018337	0.016288
12	7	1501	0.027698	0.03478	0.018011	0.60314	0.015955	0.022645
12	8	1501	0.052635	0.050503	0.054685	1.1172	0.045958	0.025661
12	4	1801	0.14823	0.17743	0.087753	3.2539	0.094387	0.11432
12	5	1801	0.035068	0.033602	0.037161	0.73626	0.029781	0.01852
12	6	1801	0.059673	0.059894	0.05934	1.2663	0.052263	0.028805
12	7	1801	0.12516	0.13937	0.1001	2.7148	0.099655	0.075729
12	8	1801	0.12388	0.13876	0.097363	2.6886	0.097604	0.076296
13	4	1200	1.474	1.4699	1.4768	30.5629	1.4714	0.087703
13	5	1200	1.4541	1.4518	1.4557	30.1535	1.4527	0.063725
13	6	1200	0.077259	0.081178	0.074534	1.6474	0.067068	0.038358

Table 6.10: Summary of RBF-NARX model construction for identification of turbine pressure

# neurons	**# delays**	**# training samples**	**RMSE**$^{P_T}_{total}$	**RMSE**$^{P_T}_{training}$	**RMSE**$^{P_T}_{test}$	**%RMSE**$^{P_T}_{total}$	**Mean** (μ_{ae})	**Std**(σ_{ae})
13	7	1200	0.037102	0.047673	0.027919	0.81098	0.024424	0.027934
13	8	1200	0.040892	0.054545	0.028353	0.89654	0.025381	0.032067
13	4	1501	0.041596	0.042409	0.040766	0.88009	0.033355	0.024856
13	5	1501	0.1115	0.11365	0.1093	2.395	0.092664	0.062017
13	6	1501	0.066864	0.064291	0.069343	1.4225	0.058569	0.032262
13	7	1501	0.049219	0.053668	0.044323	1.0726	0.036346	0.033194
13	8	1501	0.024709	0.029619	0.018538	0.5354	0.015837	0.01897
13	4	1801	1.4094	1.4076	1.412	29.2249	1.4068	0.084817
13	5	1801	0.050166	0.062046	0.022716	1.1093	0.019466	0.046243
13	6	1801	0.046749	0.045883	0.048019	0.98659	0.040852	0.022731
13	7	1801	0.028914	0.028024	0.030201	0.6052	0.023772	0.016462
13	8	1801	0.035692	0.042236	0.022553	0.78023	0.02378	0.026621
14	4	1200	1.4452	1.4428	1.4468	29.9693	1.4437	0.065887
14	5	1200	1.4614	1.4588	1.4632	30.3062	1.4598	0.068578
14	6	1200	0.07883	0.079598	0.078313	1.6729	0.070046	0.036168
14	7	1200	0.020993	0.025351	0.017496	0.44707	0.015005	0.014683
14	8	1200	0.12506	0.13048	0.12131	2.6714	0.10891	0.061474
14	4	1501	0.069742	0.066965	0.072414	1.4837	0.061019	0.033778
14	5	1501	1.4125	1.4103	1.4147	29.2909	1.4105	0.0762
14	6	1501	0.023566	0.024334	0.022773	0.49333	0.017749	0.015505
14	7	1501	0.021527	0.021479	0.021575	0.44644	0.017339	0.01276
14	8	1501	0.022119	0.025674	0.017866	0.47387	0.014976	0.01628
14	4	1801	1.4465	1.445	1.4488	29.9948	1.4445	0.074752

Table 6.10: Summary of RBF-NARX model construction for identification of turbine pressure

# neurons	# delays	# training samples	RMSE$^{P_T}_{total}$	RMSE$^{P_T}_{training}$	RMSE$^{P_T}_{test}$	%RMSE$^{P_T}_{total}$	Mean (μ_{ae})	Std(σ_{ae})
14	5	1801	0.056737	0.069122	0.029657	1.2494	0.023616	0.051597
14	6	1801	0.079201	0.08438	0.070719	1.7016	0.060981	0.050546
14	7	1801	0.098559	0.10231	0.092643	2.1036	0.085313	0.049361
14	8	1801	0.050611	0.059655	0.032628	1.1072	0.034886	0.036672
15	4	1200	1.4307	1.4272	1.433	29.6659	1.4285	0.078949
15	5	1200	0.10279	0.10923	0.098265	2.2005	0.088002	0.053122
15	6	1200	0.059166	0.061167	0.057795	1.2565	0.05124	0.029588
15	7	1200	0.074292	0.075628	0.073388	1.5814	0.065193	0.035631
15	8	1200	0.046574	0.047446	0.045983	0.98517	0.04066	0.022717
15	4	1501	1.4452	1.4441	1.4464	29.9726	1.4442	0.054294
15	5	1501	0.031998	0.032731	0.031247	0.67227	0.025259	0.019646
15	6	1501	1.4377	1.4342	1.4411	29.8111	1.4346	0.094833
15	7	1501	0.023133	0.024462	0.021722	0.49686	0.017967	0.014574
15	8	1501	0.049894	0.056786	0.041876	1.0901	0.034921	0.035642
15	4	1801	0.050115	0.062658	0.019709	1.1072	0.021415	0.045316
15	5	1801	1.4403	1.4393	1.4418	29.8702	1.4391	0.060429
15	6	1801	0.071329	0.090572	0.020303	1.5851	0.020521	0.068325
15	7	1801	0.055182	0.05401	0.056897	1.1666	0.048793	0.025779
15	8	1801	0.062035	0.061789	0.062402	1.3166	0.05472	0.029228
16	4	1200	0.06567	0.096431	0.031466	1.4551	0.028344	0.059248
16	5	1200	0.16122	0.17734	0.14952	3.4682	0.13466	0.088673
16	6	1200	0.12102	0.15922	0.086683	2.6478	0.081322	0.089646
16	7	1200	0.085805	0.13341	0.020234	1.9188	0.02176	0.083014

Table 6.10: Summary of RBF-NARX model construction for identification of turbine pressure

# neurons	# delays	# training samples	RMSE$^{P_T}_{total}$	RMSE$^{P_T}_{training}$	RMSE$^{P_T}_{test}$	%RMSE$^{P_T}_{total}$	Mean (μ_{ae})	Std(σ_{ae})
16	8	1200	0.031074	0.031003	0.031121	0.65147	0.026185	0.016735
16	4	1501	0.12161	0.16499	0.048471	2.7032	0.040982	0.11452
16	5	1501	0.037034	0.036627	0.037438	0.78281	0.030758	0.020631
16	6	1501	1.4318	1.4306	1.4331	29.6963	1.4307	0.05646
16	7	1501	0.034484	0.032592	0.036278	0.7232	0.02959	0.01771
16	8	1501	0.048363	0.048392	0.048334	1.0251	0.041594	0.02468
16	4	1801	1.4454	1.444	1.4474	29.9748	1.4434	0.075118
16	5	1801	1.4761	1.4748	1.4781	30.6118	1.4745	0.07027
16	6	1801	0.045078	0.043919	0.046762	0.95159	0.039441	0.02183
16	7	1801	0.024813	0.028923	0.016858	0.53562	0.016518	0.018518
16	8	1801	0.037625	0.036508	0.039242	0.79316	0.032311	0.019282
17	4	1200	1.4763	1.4746	1.4775	30.6193	1.4753	0.055544
17	5	1200	1.422	1.413	1.428	29.4785	1.4161	0.12899
17	6	1200	1.4384	1.4345	1.441	29.8238	1.436	0.082249
17	7	1200	0.024307	0.0263	0.022883	0.50998	0.018606	0.015644
17	8	1200	0.053271	0.053439	0.053159	1.1299	0.046792	0.025466
17	4	1501	1.4649	1.464	1.4659	30.3818	1.464	0.051179
17	5	1501	0.048999	0.058781	0.036686	1.0583	0.03019	0.0386
17	6	1501	1.4675	1.4663	1.4687	30.4352	1.4664	0.054846
17	7	1501	0.031027	0.038864	0.020361	0.67887	0.017461	0.025652
17	8	1501	0.018406	0.018638	0.018172	0.38308	0.015086	0.010548
17	4	1801	1.459	1.4581	1.4604	30.2571	1.4581	0.051869
17	5	1801	1.4377	1.4368	1.4391	29.8166	1.4364	0.06115

Table 6.10: Summary of RBF-NARX model construction for identification of turbine pressure

# neurons	# delays	# training samples	RMSE$^{P_T}_{total}$	RMSE$^{P_T}_{training}$	RMSE$^{P_T}_{test}$	%RMSE$^{P_T}_{total}$	Mean (μ_{ae})	Std(σ_{ae})
17	6	1801	0.03815	0.046058	0.021355	0.83621	0.023142	0.030334
17	7	1801	0.074625	0.074948	0.074136	1.5864	0.065705	0.035385
17	8	1801	0.07342	0.073538	0.073243	1.5605	0.064584	0.034927
18	4	1200	1.4521	1.4506	1.453	30.1166	1.4511	0.052419
18	5	1200	1.4292	1.4275	1.4303	29.6421	1.4282	0.052805
18	6	1200	1.422	1.4203	1.4231	29.4901	1.421	0.053529
18	7	1200	1.4375	1.4315	1.4415	29.8009	1.4338	0.1028
18	8	1200	0.059971	0.071567	0.050795	1.2864	0.045512	0.03906
18	4	1501	1.4679	1.4623	1.4735	30.4262	1.4624	0.12696
18	5	1501	0.091542	0.092588	0.090485	1.9623	0.076248	0.050667
18	6	1501	1.4218	1.4203	1.4232	29.4884	1.4205	0.059915
18	7	1501	0.066377	0.068944	0.063706	1.4211	0.054895	0.037322
18	8	1501	0.030275	0.03843	0.018865	0.66069	0.016332	0.025496
18	4	1801	1.4567	1.4557	1.4583	30.2115	1.4553	0.064355
18	5	1801	0.076893	0.0981	0.018513	1.7082	0.02305	0.073369
18	6	1801	1.4312	1.4297	1.4335	29.6812	1.429	0.079396
18	7	1801	0.033566	0.040858	0.017668	0.7371	0.016677	0.029134
18	8	1801	0.041694	0.051019	0.020998	0.91788	0.018033	0.037599
19	4	1200	1.4323	1.4309	1.4333	29.7046	1.4314	0.050839
19	5	1200	0.069068	0.10323	0.029133	1.5263	0.026436	0.06382
19	6	1200	0.063055	0.096234	0.021319	1.3992	0.019558	0.059955
19	7	1200	0.072169	0.07432	0.070699	1.5345	0.063071	0.035083
19	8	1200	0.10115	0.10574	0.097979	2.1595	0.087896	0.050074

Table 6.10: Summary of RBF-NARX model construction for identification of turbine pressure

# neurons	# delays	# training samples	RMSE$^{P_T}_{total}$	RMSE$^{P_T}_{training}$	RMSE$^{P_T}_{test}$	%RMSE$^{P_T}_{total}$	Mean (μ_{ae})	Std(σ_{ae})
19	4	1501	1.4276	1.4252	1.4301	29.601	1.4254	0.080317
19	5	1501	1.4119	1.4095	1.4143	29.2869	1.4101	0.071733
19	6	1501	0.16438	0.23131	0.022781	3.6681	0.027756	0.16204
19	7	1501	0.074149	0.077964	0.070125	1.5917	0.060196	0.043304
19	8	1501	0.11309	0.13408	0.087163	2.4692	0.077395	0.082469
19	4	1801	1.4478	1.4471	1.4488	30.0256	1.4468	0.052634
19	5	1801	1.4554	1.4539	1.4578	30.1799	1.4533	0.078788
19	6	1801	0.072153	0.091119	0.023638	1.5986	0.021015	0.069037
19	7	1801	0.13462	0.15046	0.10651	2.9142	0.10522	0.083982
19	8	1801	0.137	0.16037	0.091331	2.9905	0.095528	0.098221
20	4	1200	1.4434	1.4417	1.4446	29.9351	1.4425	0.052544
20	5	1200	1.4265	1.4245	1.4278	29.5887	1.4252	0.06128
20	6	1200	0.041779	0.060832	0.021043	0.91763	0.018697	0.037368
20	7	1200	0.14213	0.15349	0.13402	3.0466	0.12069	0.075074
20	8	1200	1.4438	1.4395	1.4467	29.9358	1.4413	0.085749
20	4	1501	1.4363	1.4353	1.4372	29.7895	1.4354	0.050693
20	5	1501	1.4546	1.4507	1.4585	30.157	1.4508	0.10543
20	6	1501	0.093735	0.12761	0.035837	2.0828	0.030188	0.088755
20	7	1501	0.038534	0.037442	0.039596	0.81183	0.032927	0.02002
20	8	1501	0.060782	0.060575	0.060988	1.2897	0.051954	0.031552
20	4	1801	1.4622	1.4612	1.4636	30.3222	1.4609	0.060632
20	5	1801	1.4193	1.4179	1.4214	29.4313	1.4175	0.070515
20	6	1801	1.4628	1.462	1.4641	30.3379	1.4617	0.058603

Table 6.10: **Summary of RBF-NARX model construction for identification of turbine pressure**

# neurons	**# delays**	**# training samples**	**RMSE$^{P_T}_{total}$**	**RMSE$^{P_T}_{training}$**	**RMSE$^{P_T}_{test}$**	**%RMSE$^{P_T}_{total}$**	**Mean (μ_{ae})**	**Std(σ_{ae})**
20	7	1801	1.4544	1.4534	1.4557	30.1598	1.4531	0.061056
20	8	1801	1.4652	1.4631	1.4684	30.3769	1.4624	0.090441

Table 6.11: **Summary of SVM-NARX model construction for identification of compressor temperature**

# delays	**# training samples**	**RMSE$^{T_C}_{total}$**	**RMSE$^{T_C}_{training}$**	**RMSE$^{T_C}_{test}$**	**%RMSE$^{T_C}_{total}$**	**mean (μ_{ae})**	**Std (σ_{ae})**
4	1200	5.785	6.1078	5.5595	0.88026	4.7571	3.2922
5	1200	3.2587	3.4575	3.1191	0.49584	2.5706	2.0029
6	1200	6.7519	6.9366	6.626	1.0262	5.8765	3.3252
7	**1200**	**2.6112**	**2.7304**	**2.5287**	**0.39709**	**2.0356**	**1.6357**
8	1200	8.5326	8.646	8.4563	1.2962	7.6082	3.863
4	1501	10.0526	10.2897	9.8096	1.5335	7.1987	7.0172
5	1501	2.7116	2.6177	2.8024	0.41237	2.1957	1.5913
6	1501	8.0053	8.0475	7.9629	1.22	6.1615	5.1112
7	1501	3.1256	3.0234	3.2246	0.47547	2.4852	1.8958
8	1501	10.1568	10.4808	9.8219	1.5495	7.2417	7.1222
4	1801	4.0933	3.9579	4.2883	0.6212	3.2124	2.537
5	1801	5.1256	5.1964	5.0174	0.77866	4.3276	2.7467
6	1801	25.5922	27.3954	22.6185	3.8929	19.709	16.3266
7	1801	12.9035	13.9279	11.192	1.9636	10.4707	7.5416
8	1801	4.8782	5.1378	4.4605	0.74155	3.5143	3.3836

Table 6.12: Summary of SVM-NARX model construction for identification of compressor pressure

# delays	# training samples	RMSE$^{P_C}_{total}$	RMSE$^{P_C}_{training}$	RMSE$^{P_C}_{test}$	%RMSE$^{P_C}_{total}$	mean (μ_{ae})	Std (σ_{ae})
4	**1200**	**0.04434**	**0.044539**	**0.044207**	**0.37271**	**0.023476**	**0.037618**
5	1200	0.06586	0.074462	0.059439	0.56046	0.042214	0.050556
6	1200	0.054944	0.058191	0.052668	0.46561	0.032905	0.044004
7	1200	0.051974	0.051877	0.052039	0.43826	0.022331	0.046936
8	1200	0.070633	0.067559	0.072611	0.59467	0.029654	0.064113
4	1501	0.047934	0.046986	0.048864	0.40478	0.030515	0.03697
5	1501	0.046661	0.046592	0.04673	0.39495	0.029562	0.036105
6	1501	0.059857	0.058357	0.061321	0.50837	0.033006	0.049939
7	1501	0.10016	0.10284	0.097396	0.85469	0.065978	0.075359
8	1501	0.047577	0.047181	0.04797	0.40371	0.025682	0.040053
4	1801	0.041185	0.037001	0.046766	0.34187	0.028081	0.030129
5	1801	0.16758	0.19602	0.11212	1.4287	0.11603	0.12093
6	1801	0.064463	0.04931	0.082111	0.53039	0.022916	0.060257
7	1801	0.073084	0.078509	0.064088	0.61647	0.04068	0.060721
8	1801	0.058766	0.044294	0.07544	0.48336	0.02175	0.054597

Table 6.13: Summary of SVM-NARX model construction for identification of rotational speed

# delays	# training samples	RMSE$^{N}_{total}$	RMSE$^{N}_{training}$	RMSE$^{N}_{test}$	%RMSE$^{N}_{total}$	mean (μ_{ae})	Std (σ_{ae})
4	1200	26.3914	26.3745	26.4026	0.22282	22.6819	13.4932
5	1200	27.3764	27.3992	27.3612	0.23114	23.5228	14.0064
6	1200	27.8955	27.874	27.9099	0.23553	23.9589	14.2887
7	1200	27.4287	27.3904	27.4543	0.23158	23.7126	13.7868

Table 6.13: Summary of SVM-NARX model construction for identification of rotational speed

# delays	# training samples	RMSE$^{N}_{total}$	RMSE$^{N}_{training}$	RMSE$^{N}_{test}$	%RMSE$^{N}_{total}$	mean (μ_{ae})	Std (σ_{ae})
8	1200	26.1403	26.1954	26.1035	0.2207	22.3773	13.513
4	**1501**	**24.6728**	**26.5164**	**22.6791**	**0.20831**	**21.236**	**12.562**
5	1501	26.7274	28.7473	24.5411	0.22566	22.9553	13.6908
6	1501	28.2652	30.4459	25.9008	0.23865	24.18	14.6386
7	1501	26.9222	28.8732	24.8176	0.2273	23.1971	13.665
8	1501	26.0438	27.9951	23.9333	0.21988	22.4468	13.2081
4	1801	26.545	27.0505	25.7681	0.22412	22.7997	13.5956
5	1801	27.4482	27.9494	26.6783	0.23175	23.5607	14.0829
6	1801	25.4999	25.9773	24.7663	0.21528	21.9856	12.9192
7	1801	25.2779	25.7547	24.545	0.21342	21.7604	12.864
8	1801	26.1525	26.6325	25.4153	0.2208	22.5576	13.2339

Table 6.14: Summary of SVM-NARX model construction for identification of turbine temperature

# delays	# training samples	RMSE$^{T_T}_{total}$	RMSE$^{T_T}_{training}$	RMSE$^{T_T}_{test}$	%RMSE$^{T_T}_{total}$	mean (μ_{ae})	Std (σ_{ae})
4	1200	183.6261	189.9563	179.2832	10.4182	160.2343	89.6932
5	1200	166.4114	175.3166	160.2015	9.4955	141.1499	88.1518
6	**1200**	**104.3983**	**107.0395**	**102.6002**	**5.9073**	**90.6818**	**51.7326**
7	1200	136.832	133.9356	138.7288	7.6189	123.4487	59.0255
8	1200	166.3491	161.8485	169.2822	9.2006	154.693	61.1777
4	1501	293.6453	273.8494	312.1944	16.4391	262.022	132.5706
5	1501	282.3101	269.7902	294.3017	15.649	259.5313	111.1059
6	1501	215.896	209.24	222.355	11.9547	202.3673	75.2297

Table 6.14: Summary of SVM-NARX model construction for identification of turbine temperature

# delays	# training samples	RMSE$^{T_T}_{total}$	RMSE$^{T_T}_{training}$	RMSE$^{T_T}_{test}$	%RMSE$^{T_T}_{total}$	mean (μ_{ae})	Std (σ_{ae})
7	1501	249.4072	235.4363	262.6403	13.9182	226.8929	103.5634
8	1501	236.7342	224.6235	248.2584	13.3207	212.1327	105.0933
4	1801	408.2675	398.3448	422.7191	22.527	380.7186	147.4427
5	1801	307.8869	291.5684	330.865	17.1951	274.3306	139.7866
6	1801	202.5877	190.7599	219.1402	11.2666	183.4603	85.938
7	1801	272.9621	258.0228	293.9563	15.1594	248.0573	113.9212
8	1801	205.2928	208.9996	199.6019	11.6387	174.1967	108.6398

Table 6.15: Summary of SVM-NARX model construction for identification of turbine pressure

# delays	# training samples	RMSE$^{P_T}_{total}$	RMSE$^{P_T}_{training}$	RMSE$^{P_T}_{test}$	%RMSE$^{P_T}_{total}$	Mean (μ_{ae})	Std(σ_{ae})
4	1200	0.14436	0.166	0.12793	3.164	0.099075	0.10501
5	1200	0.22767	0.25312	0.209	4.9621	0.16848	0.15314
6	1200	0.13213	0.14305	0.12432	2.858	0.10348	0.08217
7	1200	0.36862	0.40489	0.34232	8.0233	0.27583	0.24456
8	1200	0.26315	0.27079	0.25794	5.6279	0.22447	0.13736
4	1501	0.15671	0.15761	0.1558	3.4091	0.11986	0.10096
5	1501	0.14812	0.14611	0.15011	3.203	0.11887	0.088379
6	1501	0.15141	0.14926	0.15353	3.2731	0.12131	0.090617
7	1501	0.11376	0.10654	0.12055	2.4167	0.097374	0.058823
8	1501	0.11595	0.11134	0.12038	2.4909	0.094515	0.067168
4	**1801**	**0.088902**	**0.084001**	**0.095787**	**1.8751**	**0.075015**	**0.047715**
5	1801	0.14606	0.14218	0.15169	3.0978	0.12598	0.073916

Table 6.15: Summary of SVM-NARX model construction for identification of turbine pressure

# delays	**# training samples**	**RMSE**$^{P_T}_{total}$	**RMSE**$^{P_T}_{training}$	**RMSE**$^{P_T}_{test}$	**%RMSE**$^{P_T}_{total}$	**Mean** (μ_{ae})	**Std**(σ_{ae})
6	1801	0.12097	0.11917	0.12363	2.5699	0.10214	0.064829
7	1801	0.18853	0.20295	0.16453	4.0615	0.14994	0.1143
8	1801	0.27965	0.3162	0.21338	6.0711	0.21025	0.18441