Liverpool John Moores University

School of Engineering

Ph.D. Thesis

ADVANCED NEURAL NETWORK BASED CONTROL FOR AUTOMOTIVE ENGINES

Yujia Zhai

Director of Studies: Prof. Dingli Yu
Prof. Jianfu Zhang

July 2009

THE FOLLOWING HAVE NOT BEEN COPIED ON INSTRUCTION FROM THE UNIVERSITY

Appendices on pages

136 - 179

Abstract

This thesis investigates the application of artificial neural networks (NN) in air/fuel ratio (AFR) control of spark ignition(SI) engines. Three advanced neural network based control schemes are proposed: radial basis function(RBF) neural network based feedforward-feedback control scheme, RBF based model predictive control scheme, and diagonal recurrent neural network (DRNN) - based model predictive control scheme. The major objective of these control schemes is to maintain the air/fuel ratio at the stoichiometric value of 14.7 , under varying disturbance and system uncertainty. All the developed methods have been assessed using an engine simulation model built based on a widely used engine model benchmark, mean value engine model (MVEM). Satisfactory control performance in terms of effective regulation and robustness to disturbance and system component change have been achieved.

In the feedforward-feedback control scheme, a neural network model is used to predict air mass flow from system measurements. Then, the injected fuel is estimated by an inverse NN controller. The simulation results have shown that much improved control performance has been achieved compared with conventional PID control in both transient and steady-state response.

A nonlinear model predictive control is developed for AFR control in this research using RBF model. A one-dimensional optimization method, the secant method is employed to obtain optimal control variable in the MPC scheme, so

that the computation load and consequently the computation time is greatly reduced. This feature significantly enhances the applicability of the MPC to industrial systems with fast dynamics. Moreover, the RBF model is on-line adapted to model engine time-varying dynamics and parameter uncertainty. As such, the developed control scheme is more robust and this is approved in the evaluation.

The MPC strategy is further developed with the RBF model replaced by a DRNN model. The DRNN has structure including a information-storing neurons and is therefore more appropriate for dynamics system modelling than the RBF, a static network. In this research, the dynamic back-propagation algorithm (DBP) is adopted to train the DRNN and is realized by automatic differentiation (AD) technique. This greatly reduces the computation load and time in the model training. The MPC using the DRNN model is found in the simulation evaluation having better control performance than the RBF-based model predictive control.

The main contribution of this research lies in the following aspects. A neural network based feedforward-feedback control scheme is developed for AFR of SI engines, which is performed better than traditional look-up table with PI control method. This new method needs moderate computation and therefore has strong potential to be applied in production engines in automotive industry. Furthermore, two adaptive neural network models, a RBF model and a DRNN model, are developed for engine and incorporated into the MPC scheme. Such developed two MPC schemes are proved by simulations having advanced features of low computation load, better regulation performance in both transient and steady state, and stronger robustness to engine time-varying dynamics and parameter uncertainty. Finally, the developed schemes are considered to suit the limited hardware capacity of engine control and have feasibility and strong potential to be practically implemented in the production engines.

Acknowledgement

I would like to thank my thesis supervisor Prof. Ding-Li Yu for his guidance and for suggesting the present project. His encouragement to publish in conferences and scientific journals is particularly acknowledged. I would also like to thank him for being patience with me during my thesis. I want to thank the school of engineering for awarding bursary and ORSAS for supporting my research.

I would like to thank Dr. Shi-Wei Wang and Dr. Mahavir Sangha for supporting me during my stay at LJMU and providing me lot of courage and encouragement to achieve success in my thesis and in my real life.

I can not express myself how thankful I am to my parents for their support and love provided by them throughout my life. I would like to thank my dearest wife Yuanyuan Qi for her support, love, patience, and for putting up with me for all these four years.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Importance of Air Fuel Ratio Control

Internal combustion engines used in vehicles are a major source of urban air pollution. The spark-ignition (SI) engine exhaust gases contain oxides of nitrogen (nitric oxide, NO, and small amounts of nitrogen dioxide, NO_2 — collectively known as NO_x), carbon monoxide and organic compounds which are unburned or partially burned hydrocarbons (HC) [2]. Governments set strict emission standards for engines to reduce the air pollution. Since 1980, the permitted levels for the concentrations of carbon monoxide CO, HC, NO_x and particulate matter (PM) have been reduced significantly. Table 1.1 shows the development of the limits in Europe [3], from which, we can expect that this trend will continue in foreseeable future.

The three-way catalytic converter (TWC) is the most common pollution abatement system for SI engine. State-of-the-art systems are capable of removing more than 98% of the pollutants. However, this can only be achieved by operating the engine within very narrow air fuel ratio limits for both steady state and transient operation [4]. The variations of greater than 1% below value of AFR 14.7 can result in significant increase of CO and HC emissions. An increase of more

	Euro – I	Euro – II	Euro – III	Euro – IV
SI Engines				
CO	2.72	2.2	2.3	1
HC	–	–	0.2	0.1
NO_x	–	–	0.15	0.08
$HC + NO_x$	0.97	0.5	–	–
PM	0.14	–	–	–
Diesel Engines				
CO	3.16	1	0.64	0.5
HC	–	–	–	–
NO_x	–	–	0.5	0.25
$HC + NO_x$	1.13	0.7	0.56	0.3
PM	0.18	0.08	0.05	0.025

Table 1.1: European emission standards for passenger cars (g/km)

than 1% will produce more NO_x up to 50%. As shown in Figure 1.1 [1], efficient simultaneous conversion of CO , HC and NO_x occurs only in a narrow band about stoichiometric air fuel ratio, which is near 14.7:1 for typical formulations of gasoline. Therefore, air fuel ratio control remains critical to effective emissions control with a TWC. In addition, the best balance between power output and fuel consumption can be obtained only by maintaining an air fuel ratio of 14.7, Figure 1.2 shows SI engine's power output and its fuel consumption varying with air fuel ratio [5].

A substantial improvement of automotive engine performance is therefore required in order to comply with the required characteristics of very low emission levels and high fuel economy. These strong constraints have made the research on air fuel ratio control systems an active area in both research and automotive industry.

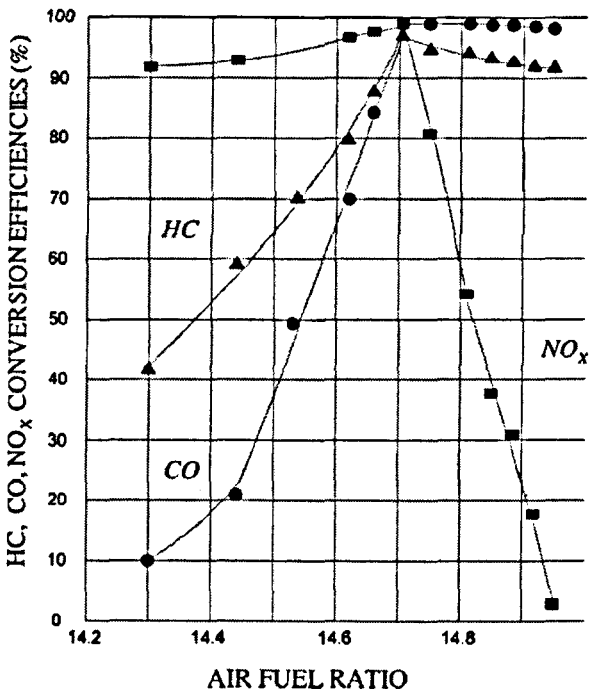

Figure 1.1: TWC Conversion Efficiency Varies with Air Fuel Ratio [1]

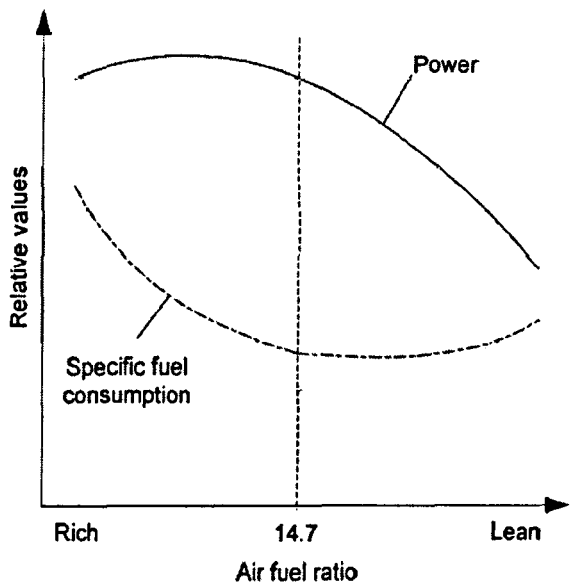

Figure 1.2: Power Output and Fuel Consumption Influenced by Air Fuel Ratio

1.2 Aims and Objectives

In automotive industry, the look-up tables based feed-forward control scheme is widely adopted in current production electronic control unit (ECU) to maintain air fuel ratio of the highly dynamic and nonlinear SI engines. The disadvantage of this method is that the steady state look-up table lacks the capability to compensate transient engine dynamics, and can not guarantee the system robustness against external disturbances and time varying effects, such as engine aging problems. Moreover, it takes a huge effort and labors for engine calibration to obtain control data to fill the tables.

Although the introduction of PID controller later on AFR control improves the steady state performance, the transient response of the AFR is still not satisfactory, due to the variable time-delay in the AFR measurement and the poor compensation for sharp throttle position change. Additionally, due to the severe nonlinearity of engine dynamics, PID controller can not work effectively in the wide range of engine operating condition.

The aim of this project is to develop advanced control schemes for AFR control of automotive SI engines to maintain the AFR within 14.7 $\pm 1\%$ under operational engine working conditions. The control schemes to be developed will be based on neural networks model, neural network controller and model predictive control strategy. Under the guideline of this aim, the following objectives are proposed and will be achieved in the project.

1. Develop a feedforward-feedback control scheme for the AFR control, in which a neural network model and a neural network controller will be used to estimate air mass flow and to generate the control variable - the fuel flow rate.

2. Develop a model predictive control scheme for the AFR control, in which

an RBF model will be used to predict future AFR. The performance of the AFR under the developed MPC should robust with respect to the throttle position change and robust to the plant component change.

3. Develop a DRNN model based predictive control scheme for the AFR control. In addition to achieve all the improved performance, computation load in MPC is expected to be reduced to allow the scheme could be practically implemented for real-time control.

4. The developed new control schemes will be evaluated using a well known engine simulation benchmark - the mean value engine model for regulation against disturbances and robustness against plant component change. Also, comparisons will be done between the schemes.

With these objectives achieved, the accuracy of AFR control will be greatly improved and system complexity is reduced. Additionally, moderate computation are needed to realize such control algorithms.

1.3 Thesis Outline

The thesis proposes three advanced air fuel ratio control schemes for SI engine. It consists of eight chapters and is organized as follows. After an introduction in Chapter 1, working principles of the four-stroke engine and the mean value engine model used in this research is described in details in Chapter 2. To give readers more comprehensive overview, Chapter 3 provides the latest technological advances in the research and industry. The literature survey includes most commonly used modelling and control methods for automotive engine. Chapter 4 presents some common function approximating neural networks and their training algorithms. Automatic differentiation technique is introduced in this chapter as a new method that combines the dynamic back propagation algorithm for training recurrent neural networks. In addition, the detailed steps that are used

to model dynamic systems are explained. In Chapter 5, a radial basis function based feed-forward and feedback control scheme is proposed. It also includes how to model intake manifold dynamics using radial basis function network and how to obtain an inverse controller of fuel injection dynamics. A neural network based model predictive control is developed in Chapter 6, to reduce the computation burden on optimization, the secant method and the reduced hessian method are investigated. Their control performances are compared in term of tracking error and time cost on optimization. Based on the same control principle, Chapter 7 proposes a model predictive control on AFR using diagonal recurrent neural network . Automatic differentiation technique is applied to train such networks efficiently. Since three different methods above are all proposed to control AFR, it is useful to compare them in order to emphasize their relative advantages and disadvantages. The discussions on this issue are given at the end of corresponding chapter. Finally, Chapter 8 concludes the contributions in this project with the research results in previous chapters. Furthermore, an outlook for future research directions and areas is also given in this chapter.

Chapter 2

SI Engine Simulation Models

2.1 Engine Process Description

2.1.1 Engine Operating Cycles

The purpose of internal combustion(IC) engines is the production of mechanical power from the chemical energy contained in the fuel. Figure 2.1 shows the

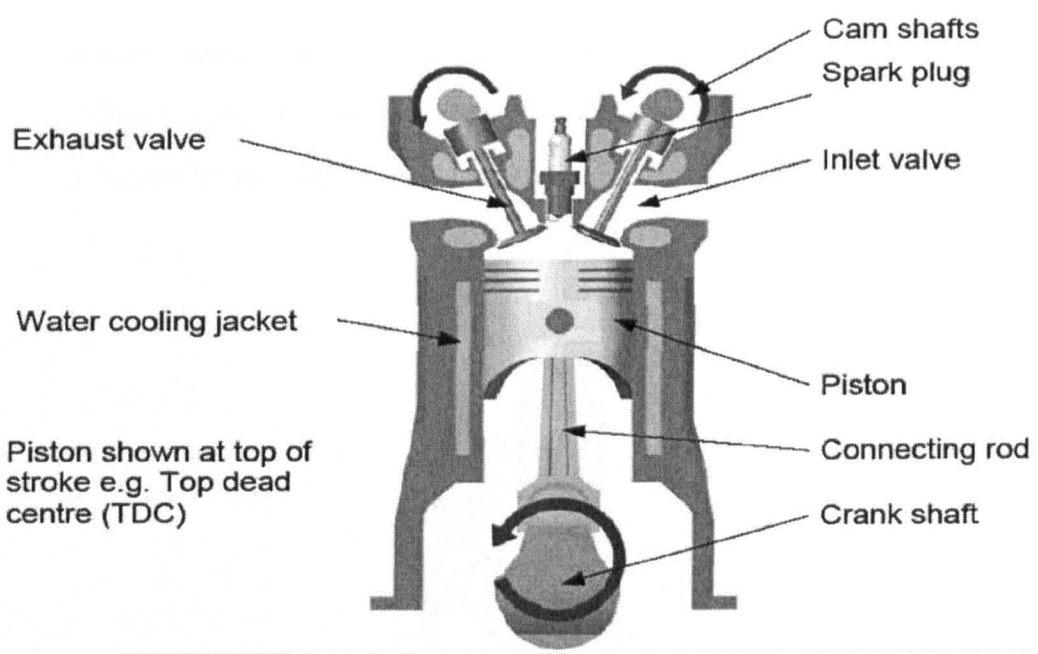

Figure 2.1: Structure of Spark Ignition Engine

structure of a typical spark ignition(SI) engine, a kind of IC engines widely used

in modern automobile industry [6]. Each cylinder of a SI engine requires four strokes of its piston to complete the sequence of events which produces one power stroke. These operations continuously repeated in the following order:

1. ***intake stroke*** The space in the cylinder above the piston is filled with the charge(the mixture of petrol vapor and air). The mixture or air is able to enter the cylinder when the piston moves down. See Figure 2.2 [6].

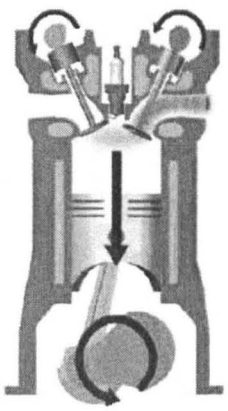

Figure 2.2: Intake Stroke

2. ***compression stroke*** The charge is compressed into the top end of the cylinder (called combustion chamber) thus raising its temperature. This is achieved by the upward movement of the piston. See Figure 2.3 [6].

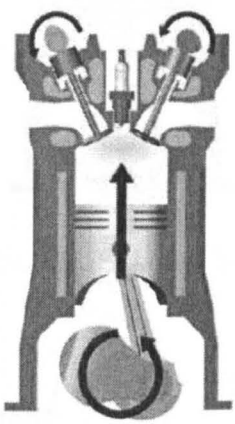

Figure 2.3: Compression Stroke

3. ***power stroke*** The petrol vapour is ignited by an electric spark and burned. The resulting increase in pressure drives the piston down the cylinder. See Figure 2.4 [6].

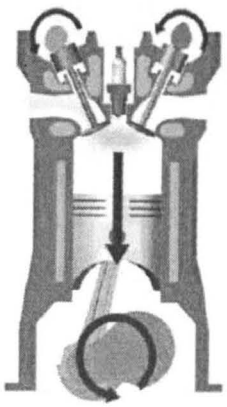

Figure 2.4: Power Stroke

4. ***exhaust stroke*** The exhaust valve is opened and the high pressure exhaust gas remaining in the cylinder after the power stroke is discharged. This is achieved by the upward movement of the piston. See Figure 2.5 [6].

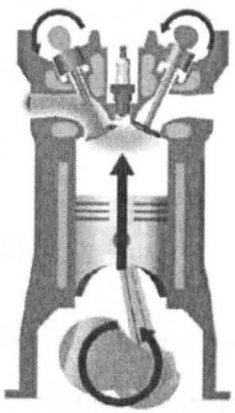

Figure 2.5: Exhaust Stroke

2.1.2 EMS and Its Tasks

An engine control unit (ECU) is an electronic control unit which controls various aspects of an internal combustion engine's operation. The ECU consists of one

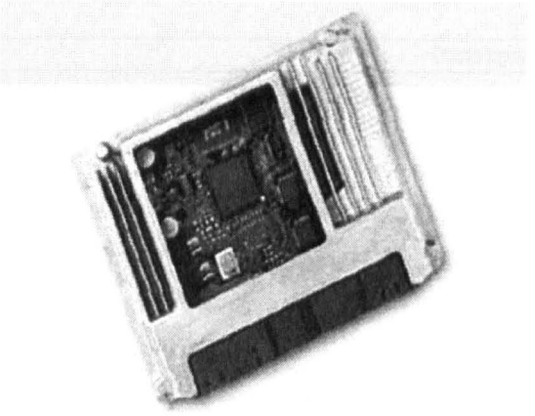

Figure 2.6: DI Motronic Developed by Bosch

or several small powerful micro-computers, which evaluate and set the data for different control tasks. The simplest ECUs control only the quantity of fuel injected into each cylinder each engine cycle, and therefore maintain a AFR of 14.7, so that the highest efficiency can be achieved in the engine operating conditions. More advanced ECUs found on most modern cars also control the ignition timing, variable valve timing (VVT), the level of boost in turbo-charged cars. A modern engine management system (EMS) can integrate all these functions of ECUs and on-board diagnosis of car engine. Figure 2.7 shows typical sensor input signal information necessary for the EMS to calculate the correct output signals used to control the various actuators, the ignition, injectors and idle speed control [4].

2.1.3 Electronic Fuel Injection

Before 1980s, the fuel delivery method for almost all gasoline fuelled engines is usually realized by carburetors. Nowadays, electronic fuel injection systems have replaced carburetors in the marketplace because this kind of systems can provide an accurate, reliable and cost-effective method of metering fuel and providing maximum engine efficiency with clean exhaust emissions. Modern fuel injection systems are controlled electronically because this form of control enables the fuel quantity to be accurately set to suit the engine operating conditions. In term of the practice cost, injection systems are much more expensive than carburettor

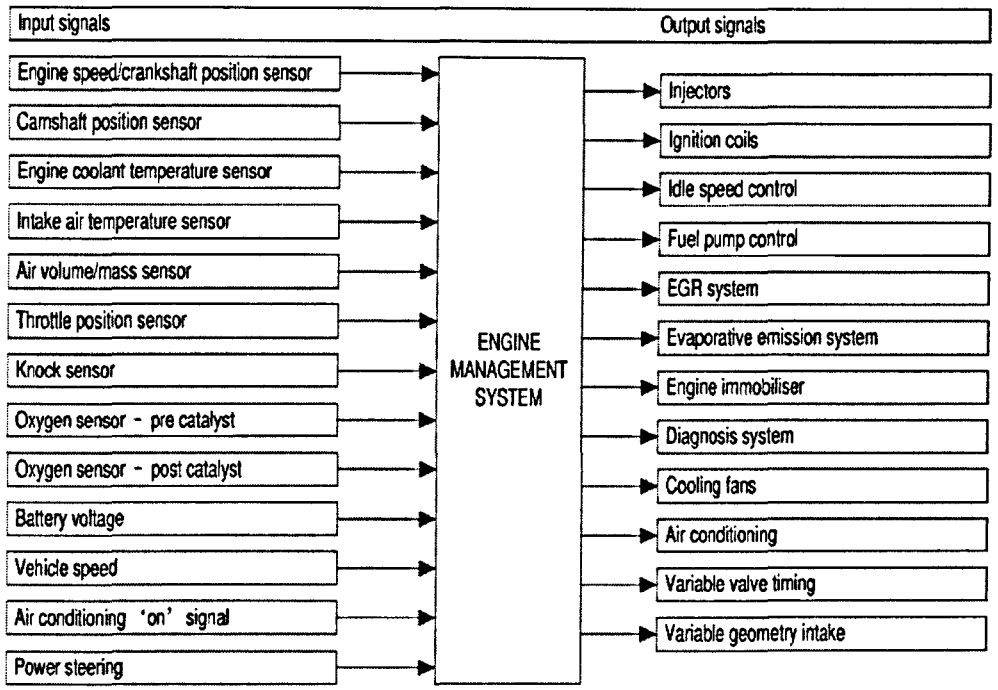

Figure 2.7: A Typical Engine Management System

fuel systems, the strict emission control regulation is one of the most important reasons for their wide implementation in automotive industry.

In operation, the ECU receives data from all of the sensors connected with the engines fuel needs. The ECU compares the input data from the sensors with the data stored in the computer memory. From this comparison of data the ECU provides some output data which appears on the injector cables as an electrical pulse that lasts for a set period. This injector electrical pulse time varies from approximately 2 milliseconds (ms), to around 10 ms. The *duty cycle* concept is based on the percentage of the available time for which the device is energised. For example, if the pulse time varies as Figure 2.8, a single cycle is indicated by 'C', which consists of an ON time 'A', and an OFF time 'B', then, the duty cycle is 25% [7]. There are two main injection systems according to the position of the injectors, one is the manifold injection(MI) and the other is direct injection(DI).

$$C = 60ms(100\%)$$
$$A = 15ms(\frac{1}{4} of\ 60ms)$$
$$B = 45ms(\frac{3}{4} of\ 60ms)$$

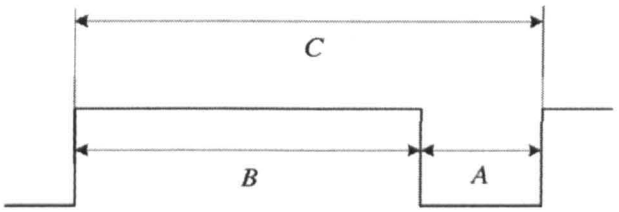

Figure 2.8: Duty Cycle

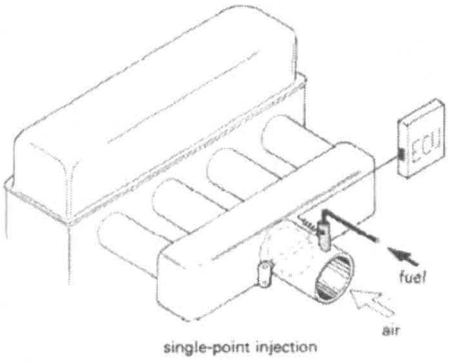

Figure 2.9: Single-Point Fuel Injection

Manifold Injection System

This 'indirect' injection is preferred by automotive industry because the injectors used for such injection system operate at a relatively low fuel pressure. Manifold injection systems can be classified into two main groups, single-point and multi-point.

- *single-point injection*- one injector discharges fuel into the air stream before the throttle butterfly, hence it is also known as throttle-body injection. See Figure 2.9 [6].
- *multi-point injection* - individual injectors are used for each cylinder. An injector is positioned near to the inlet valve of each cylinder. See Figure

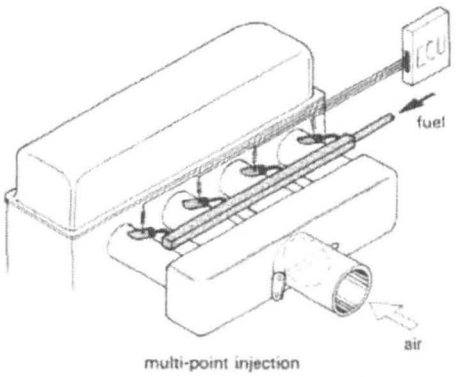

Figure 2.10: Multi-Point Fuel Injection

2.10 [6].

Direct Injection System

Many diesel engines feature direct injection. The injection nozzle is placed inside the combustion chamber and the piston incorporates a depression where initial combustion takes place. See Figure 2.11 [8]. With the development in engine and electronic technology together with stricter emission controls, a number of vehicle manufacturers have developed lean-burn SI engines fitted with gasoline direct injection (DI) system. The displacement volume of such kind of SI engines must be 60% higher than that of a stoichiometric SI engine to obtain the same maximum power output, therefore, a turbo-charger is often required to increase the air supply at a given displacement volume. DI system costs more than indirect injection systems; the injectors are exposed to more heat and pressure, so more costly materials and higher-precision electronic management systems are required.

2.2 Nonlinear Engine Simulation Model

It is economical to design engine control system based on engine simulation model in both industrial practice and scientific research. As described in the section for engine operating cycles, the thermodynamic and kinetic processes are very fast,

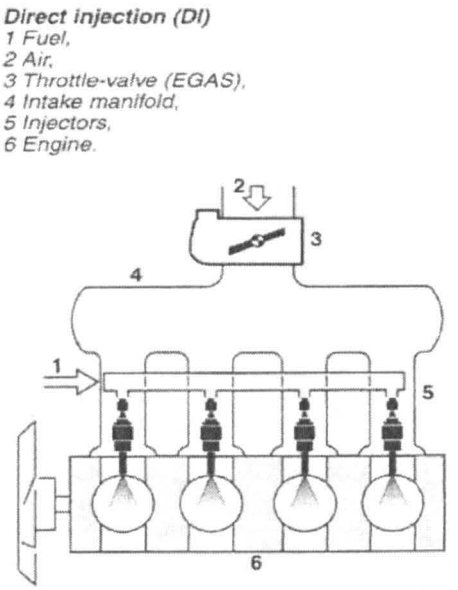

Figure 2.11: Direct Fuel Injection

usually just a few milliseconds for a full operating cycle, which are not accessible for control purposes. Moreover, the models necessary to describe these phenomena are rather complex and not suitable to design real-time control systems. Therefore, in this section, the models that will be introduced are control oriented models (COM), it means the input-output behaviour of the systems are modelled with reasonable precision but low computational complexity, and these models include all relevant dynamic effects explicitly. The experiments need to be carried out to identify key parameters. A typical structure of such a COM for an SI engine is shown in Figure 2.12. There are two types of engine models based on this structure of COM, one is cylinder by cylinder engine model (CCEM); the other is mean value engine model (MVEM). The proper choice of model class depends on the problem to be solved.

2.2.1 Cylinder by Cylinder Engine Model

In CCEM, the fundamental events are triggered by the crank angle degrees and the torque output during the power stroke is approximated by a mathematical function to capture the shape of the torque pulse, one calculation is done for each

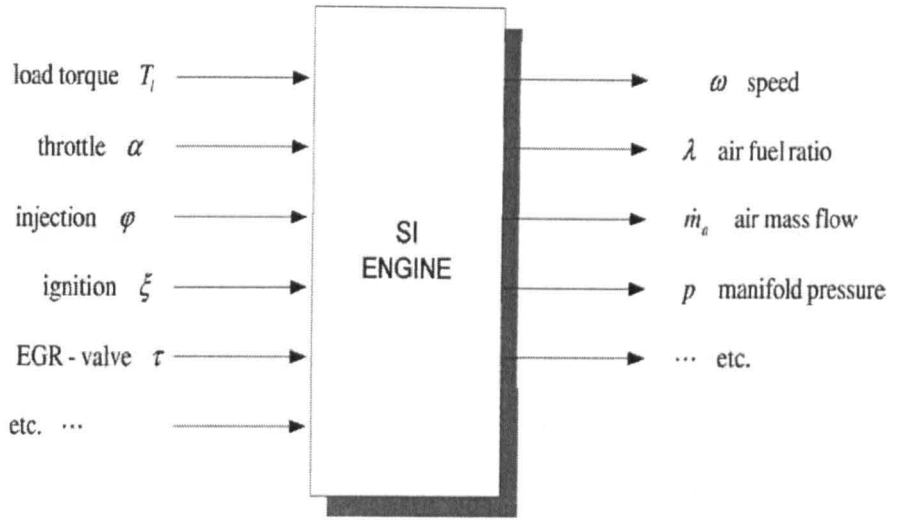

Figure 2.12: Main Input/Output Signals in a COM of an SI Engine

torque pulse. As CCEMs take into account the reciprocating behaviour of the engine, they are usually be used for the research on the misfire detection, the torque production, the gas exchange of the individual cylinders and the ignition processes, etc [4].

2.2.2 Mean Value Engine Model

The time t is the independent variable in MVEM. There are no spatially varying variables for the system descriptions that are represented by ordinary differential equations (ODE). MVEM neglect the discrete cycles of the engine and assume all processes are effects are spread out over the engine cycles. The reciprocating behaviour of engines is captured by introducing delays between cylinder-in and cylinder-out effect. The research done by Jonas Karlsson and Jonas Fredriksson proves that an MVEM is well suited for use in control algorithms or for evaluation of control strategies [9].

2.3 Mean Value Engine Model

The engine model adopted in this research is the mean value engine model developed by Elbert Hendricks [10], which is a widely used benchmark for engine modelling and control. The platform which has been selected for this MVEM is the popular *MATLAB/SIMULINK*. The three distinct systems of this model are the fuel injection, manifold filling and the crankshaft speed dynamic subsystems and those systems are modelled independently. Since this MVEM can achieve a steady state accuracy of about $\pm 2\%$ over the entire operating range of the engine, it is extremely useful for validation of control strategies using simulation. A full description of the MVEM can be found in [10] [11].

2.3.1 Manifold Filling Dynamics

The intake manifold filling dynamics are analysed from the viewpoint of the air mass conservation inside the intake manifold. It includes two nonlinear differential equations, one for the manifold pressure and the other for the manifold temperature. The manifold pressure is mainly a function of the throttle angle α, the engine speed n, the air mass flow past throttle plate $\dot{m}_{at}$, the air mass flow into the intake port $\dot{m}_{ap}$, the exhaust gas re-circulation (EGR) mass flow $\dot{m}_{EGR}$, the EGR temperature T_{EGR} and the manifold temperature T_i. It is described as

$$\begin{aligned} \dot{p}_i &= \frac{\kappa R}{V_i}(-\dot{m}_{ap}T_i + \dot{m}_{at}T_a + \dot{m}_{EGR}T_{EGR}) \\ &= f_p(\alpha, p_i, T_a T_i, n, m_{EGR}, T_{EGR}) \end{aligned} \tag{2.1}$$

where T_a is ambient temperature, κ is the ratio of the specific heats, and V_i is manifold plus port passage volume. The manifold temperature dynamics are described by the following differential equation

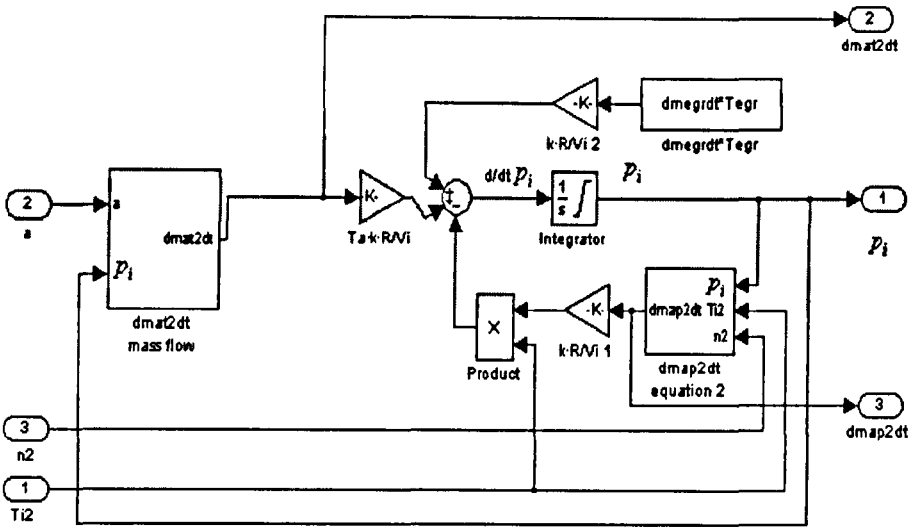

Figure 2.13: Block Diagram of Manifold Pressure Dynamics

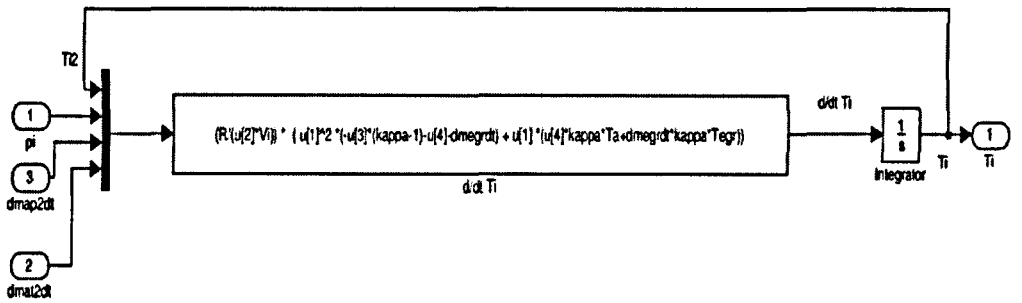

Figure 2.14: Block Diagram of Manifold Temperature Dynamics

$$\begin{aligned}\dot{T}_i &= \frac{RT_i}{p_i V_i}[-\dot{m}_{ap}(\kappa-1)T_i + \dot{m}_{at}(\kappa T_a - T_i) + \dot{m}_{EGR}(\kappa T_{EGR} - T_i)] \\ &= f_T(\alpha, p_i, T_a, T_i, n, m_{EGR}, T_{EGR})\end{aligned} \tag{2.2}$$

where α is the throttle angle, R is gas constant(here 287×10^{-5}). Equation (2.1) and (2.2) can be realised by *MATLAB/SIMULINK* as shown in Figure 2.13 and 2.14, the air mass flow dynamics in the intake manifold can be described as follows. The air mass flow past throttle plate $\dot{m}_{at}$ is related with the throttle position and the manifold pressure. The air mass flow into the intake port $\dot{m}_{ap}$ is represented by a well-known speed-density equation:

$$\dot{m}_{at}(\alpha, p_i) = m_{at1}\frac{p_a}{\sqrt{T_a}}\beta_1(\alpha)\beta_2(p_r) + m_{at0} \tag{2.3}$$

$$\dot{m}_{ap}(n, p_i) = \frac{V_d}{120RT_i}(\eta_i \cdot p_i)n \tag{2.4}$$

where

$$\beta_1(\alpha) = 1 - \cos(\alpha) - \frac{\alpha_0^2}{2!} \tag{2.5}$$

$$\beta_2(p_r) = \begin{cases} \sqrt{1 - (\frac{p_r - p_c}{1 - p_c})^2} & if \quad p_r \geq p_c \\ 1 & if \quad p_r < p_c \end{cases} \tag{2.6}$$

$$p_r = \frac{p_i}{p_a} \tag{2.7}$$

m_{at0}, m_{at1}, α_0 are constants and $p_c \neq 1$. V_d is engine displacement. Additionally, instead of directly model the volumetric efficiency η_i, it is easier to generate the quantity $\eta_i \cdot p_i$ which is called normalised air charge. The normalised air charge can be obtained by the steady state engine test and is approximated with the first-order polynomial Equation (2.8)

$$\eta_i \cdot p_i = s_i(n)p_i + y_i(n) \tag{2.8}$$

where $s_i(n)$ and $y_i(n)$ are positive functions of the crankshaft speed and $y_i \ll s_i$.

2.3.2 Crankshaft Speed Dynamics

The crankshaft speed is derived based on the conservation of the rotational energy on the crankshaft. Its state equation can be written as

$$\begin{aligned} \dot{n} &= -\frac{1}{In}(P_f(p_i, n) + P_p(p_i, n) + P_b(n)) + \frac{1}{In}H_u\eta_i(p_i, n, \lambda)\dot{m}_f(t - \Delta\tau_d) \\ &= f_n(p_i, n, m_f, \lambda) \end{aligned} \tag{2.9}$$

Both the friction power P_f and the pumping power P_p are related with the manifold pressure p_i and the crankshaft speed n. The load power P_b is a function of the crankshaft speed n only. The indicated efficiency η_i is a function of the

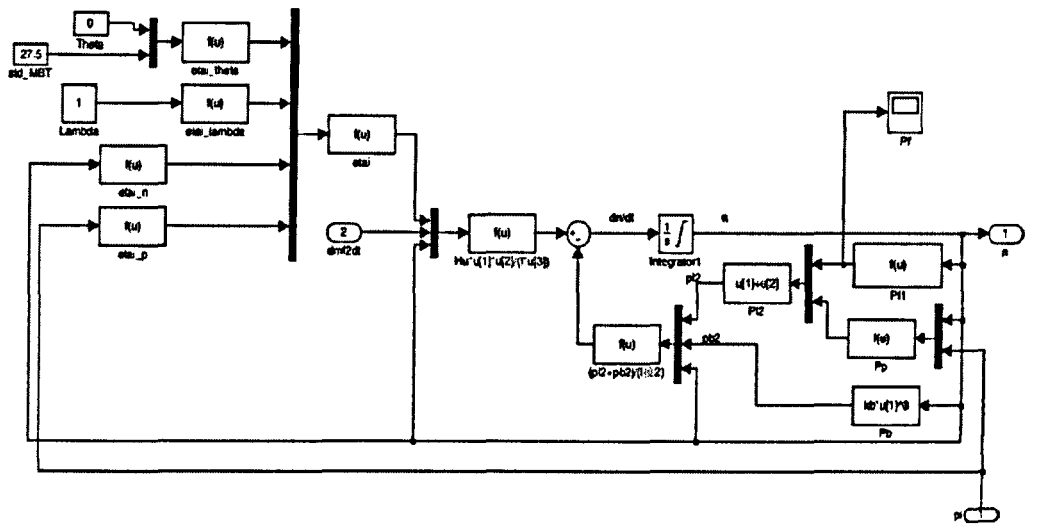

Figure 2.15: Block Diagram of Crankshaft Speed Dynamics

manifold pressure p_i, the crankshaft speed n and the air fuel ratio λ. Here, I is crank shaft load inertia; H_u is fuel lower heating value; $\dot{m}_f$ is engine port fuel mass flow; $\Delta\tau_d$ is the injection torque delay time. The implementation of Equation (2.9) is shown in Figure 2.15

2.3.3 Fuel Injection Dynamics

It has been found that the fuel jet from the injector can be characterised into two portions. One portion mixes with the air stream and enters the cylinder directly; the other portion deposits as fuel film on the surfaces of the intake system components, and mixes with the air stream through the reentrainment/evaporation process during subsequent engine cycles. This is known as wall-wetting.

According to Hendrick's identification experiments with SI engine, the fuel flow dynamics could be described as following equations [10]

$$\ddot{m}_{ff} = \frac{1}{\tau_f}(-\dot{m}_{ff} + X_f\dot{m}_{fi}) \tag{2.10}$$

$$\dot{m}_{fv} = (1 - X_f)\dot{m}_{fi} \tag{2.11}$$

$$\dot{m}_f = \dot{m}_{fv} + \dot{m}_{ff} \tag{2.12}$$

where $\dot{m}_{ff}$ is fuel film mass flow; $\dot{m}_{fi}$ is injected fuel mass flow; $\dot{m}_{fv}$ is fuel vapor mass flow. The model is based on keeping track of the fuel mass flow. The parameters in the model are the time constant for fuel evaporation, τ_f, and the proportion of the fuel which is deposited on the intake manifold or close to the intake valves, X_f. These parameters are operating point dependent and thus the model is nonlinear in spite of its linear form. The MVEM provided by Elbert Hendrick has been validated using the real time data acquired from the engine test bed that equipped with Ford 1.6L engine. The parameters for this model could be approximately expressed in the terms of the states of the model as

$$\tau_f(p_i, n) = 1.35(-0.672n + 1.68)(p_i - 0.825)^2 + (-0.06n + 0.15) + 0.56 \tag{2.13}$$

$$X_f(p_i, n) = -0.277p_i - 0.055n + 0.68 \tag{2.14}$$

2.3.4 MVEM under AFR Measurement Delay

The AFR could be calculated using Equation (2.15)

$$\lambda = \frac{\dot{m}_{ap}}{\dot{m}_f} \tag{2.15}$$

Nowadays, in the practical application of automotive industry, oxygen sensors are used in the fuel injection system. They determine if the air fuel ratio exiting a gas-combustion engine is rich (with unburnt fuel vapour) or lean (with excess oxygen), then, a closed-loop feedback controller,usually a PI controller, adjusts fuel injection rate m_{fi} according to real-time sensor data rather than operating with a open-loop fuel map. Therefore, the time delay of injection systems should also be considered. Manzie's research [12] [13] has shown there are three causes

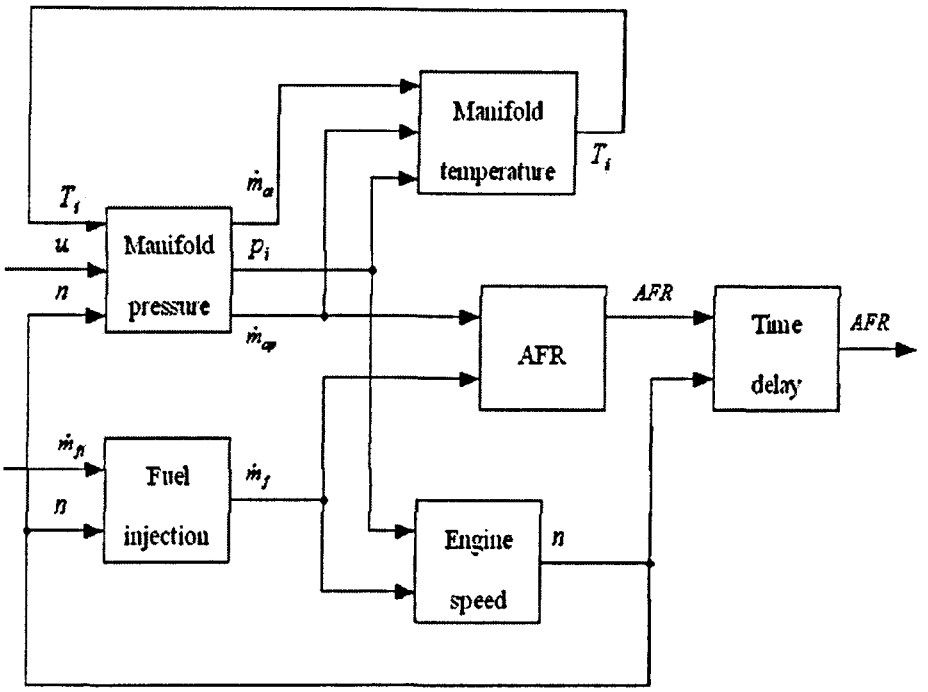

Figure 2.16: Expanded Mean Value Engine Model

of time delay for injection systems: the two engine cycle delay between the injection fuel and the expulsion from the exhaust valves, the propagation delay for the exhaust gases to reach the oxygen sensor and the sensor output delay. It has been found that the engine speed has more influence on these delays than the manifold pressure. Therefore, the following equation can be used to represent the delays of injection systems.

$$t_d = 0.045 + \frac{10\pi}{n} \tag{2.16}$$

The time delay on air fuel ratio measurement has not been considered in original MVEM. A module used for air fuel ratio measurement is added into original MVEM for the research purpose of AFR control, which is based on Equation (2.16). The expanded system is shown in Figure 2.16, it has two inputs - fuel injection rate m_{fi} and throttle angle u; one output - air fuel ratio AFR.

The valid throttle angle for the model depends on what kind of throttle body

is used. If the throttle plate could be turned so that it is perpendicular to the axis of the throttle throat, then, this is a throttle angle of zero degrees. However, this angle cannot of course be attained physically as throttle plates are nearly always elliptical. Usually the throttle plate in the closed position is say 9 to 12 degrees. Using this convention, the maximum opening angle of the throttle (if it is to be effective) is about 70 to 80 degrees. Opening it wider than this will have no influence on the throttle air mass flow. In this project, the throttle angle changes from 20 degrees to 70 degrees in 10 degree steps, with 0.5 random measurement error chosen to simulate the driving dynamics of the simulation engine. It almost covers the whole engine operating conditions.

As described before, an electric pulse triggered by the ECU energies the solenoid in fuel injector, therefore, the fuel injection rate can be controlled by adjusting the duty cycle in each sample time. For the expanded engine model in this project, the fuel injection rate ranges from 6.28×10^{-4} *kg/second* to 3.0×10^{-3} *kg/second.*

The change of throttle angle is considered as external disturbance as it is usually given by the vehicle driver and can not be predicted. However, it is easy to be measured by throttle position sensor. The control methods developed are to control the fuel injection rate to make sure the AFR varying within the $\pm 1\%$ bounds of the stoichiometric value.

2.4 Summary

In this chapter, the operating cycles of a typical four-stroke engine has been described briefly and the engine management system to control the combustion process is introduced. As this research focuses on the fuel injection system for SI engine, the main injection systems in automotive industry are shown and explained. A proper engine simulation model for air fuel ratio control purpose has

been developed based on Elbert Hendricks' well-known mean value engine model. It consists of the modules for the intake manifold dynamics, the crankshaft speed dynamics the fuel injection dynamics. For the control purpose in our research, the MVEM is expanded by adding a module for air fuel ratio measurement delay. With fixed engine load, the operating conditions of MVEM is discussed in term of the range of throttle angle.

Chapter 3

Literature Survey

3.1 Introduction

Performance requirements and environmental concern drive the development of automobiles and engines towards more efficiency and less pollution. Since electronic engine controls was introduced in the 1980's, significant improvements in fuel economy and emissions reductions have been achieved by the development of this area. Look-up table is the dominant method used by ECU in automotive industry. The reasons for this are the simplicity, low computation load, and reliability. However, the difficulty on the implementation of look-up tables is that it will take a huge effort and labors for engine calibration to obtain control data to fill the tables. Besides, the increased number of control and sensor signals, the nonlinear characteristics, time dependents of engine processes, and unavoidable time delays all make the calibration process more complicated. In modern ECUs, around 9000 parameters and more than 600 control and diagnostic functions are implemented [14] [15]. Engine control using look-up tables method can be found in [16] in details.

The ECUs in next generation are expected to have the ability to overcome these difficulties brought by the increasing requirement on engine performance and

the limitation of look-up table method, and improve the control performance by adopting advanced control technologies. Moreover,online adaptation is another attractive characteristic for future engine control system, which means that the control system can alter settings to take account of changes in the condition of components, such as engine wear, components replacement, air leakage in manifold, etc. With the increasing computational powers that are becoming available for engine control units, more advanced methods for modelling and control can be implemented to increase the engine efficiency and reduce the fuel consumption. The purpose of this chapter is to review the recent research that has been done on advanced modelling and control techniques for automotive engines.

3.2 Engine Modelling Methods

First-principles models are chosen in system modelling if a knowledge of underlying physical, chemical and thermodynamic processes is known. However, such first-principles dynamic models for complex industrial processes are very difficult to obtain. The development of such models is always very expensive and time consuming [17]. The engine models by first principles methods, which have been introduced in Chapter 2 are often called analytic engine model. They are important to investigate internal combustion engine dynamics and design the controllers for production engine. They are typically used for hardware in loop (HiL) testing of ECUs or during the design phase of controller algorithms for early validation by offline simulation. In addition to lookup tables, there are many other non-analytic engine models that can be found to describe automotive engine dynamics in both industry practice and university research. These engine modelling methods presented are based on the techniques of system identification for control purpose. The models obtained in this way are called 'grey box' models that represent only the input-output behavior of the plant, and carry a little information about its internal structure. Therefore, high accuracy and good

flexibility can be achieved by these modelling methods.

3.2.1 NARX Models

Auto-regression with extra inputs (ARX) model describes the input-output relationship of dynamic system as a difference equation:

$$\begin{aligned} &y(t) + a_1 y(t-1) + \cdots + a_{na} y(t-n_a) \\ &\quad = b_1 u(t-1) + \cdots + b_{n_b} u(t-n_b) + e(t) \end{aligned} \tag{3.1}$$

$e(t)$ is a direct error in the difference equation, usually white noise. The adjustable parameters are

$$\theta = \begin{bmatrix} a_1 & a_2 & \cdots & a_{n_a} & b_1 & b_2 & \cdots & b_{n_b} \end{bmatrix}^T \tag{3.2}$$

The vector θ of nonlinear ARX (NARX) model should depend on the values of output $y(j) : j = t-n_a, t-n_a+1, \cdots, t$ and input $u(k) : k = t-n_b, t-n_b+1, \cdots, t$ at time t [18].

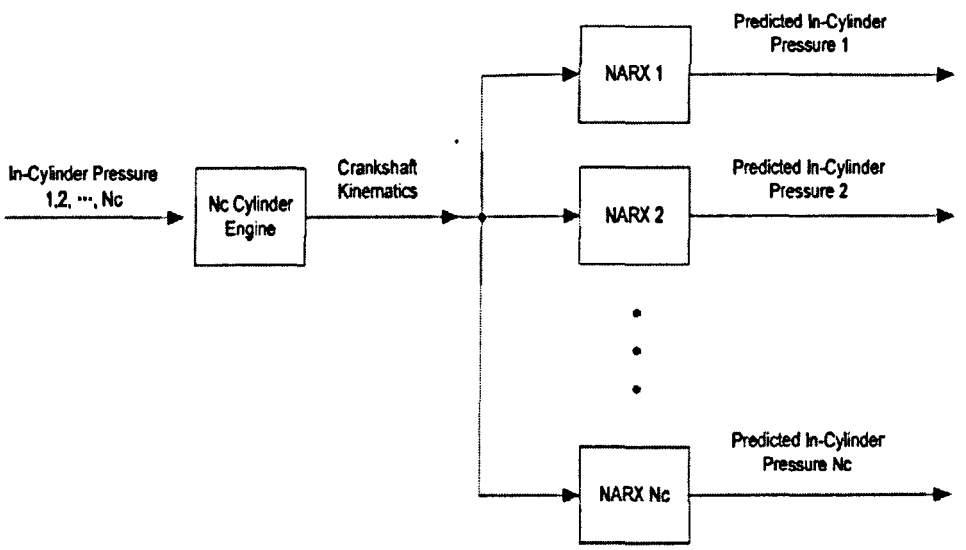

Figure 3.1: In-cylinder pressure reconstruction using NARX models

A recent paper [19] describes such a NARX model which is used for cylinder pres-

sure trace reconstruction on a multi-cylinder engine as shown in Figure 3.1. Four input variables are chosen for the inverse crank dynamics model, to predict the in-cylinder pressure, which are crank replacement, crank acceleration, the delayed crank acceleration, and the delayed in-cylinder pressure. A very good structure of NARX model has been found for multi-cylinder pressure reconstruction in this research. Other practical applications of NARX models in automotive industry have been found in the ignition sweep, exhaust temperature model, engine fault diagnosis, and etc [20] [21] [22]. However, the optimal structure of NARX model is difficult to obtain. More recently, a Takagi-Sugeno fuzzy inference is introduced to NARX model to find the appropriate model network structure, simulation results of the air fuel ratio control based on such NARX model are very attractive [23]. Nevertheless, the main drawback of NARX model remains, which is model estimation and adaptation for multi-variable nonlinear systems require a high computational effort.

3.2.2 Fuzzy Models

Four different fuzzy classes can be identified - relational fuzzy systems, linguistic fuzzy systems, singleton fuzzy systems, and Takagi-Sugeno fuzzy system. The common fuzzy model used in automotive industry is linguistic fuzzy systems that can be divided into three parts as shown in Figure 3.2. Crisp, continuous inputs are transformed into linguistic variables with membership grades between 0 and 1. This process is known as fuzzification. The linguistic inputs are then evaluated

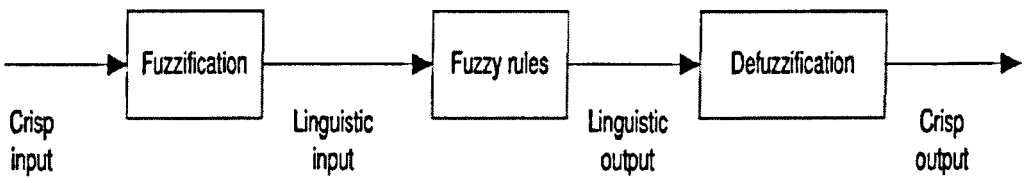

Figure 3.2: Processing Steps of a Fuzzy System

using fuzzy rules and formed into fuzzy outputs. Continuous crisp outputs are

then obtained via the process of defuzzification [5]. Fuzzy modelling enables the integration of different information of diverse sources. It is possible to operate with linguistically formulated rules, physical laws or quantitative information of measurement data.

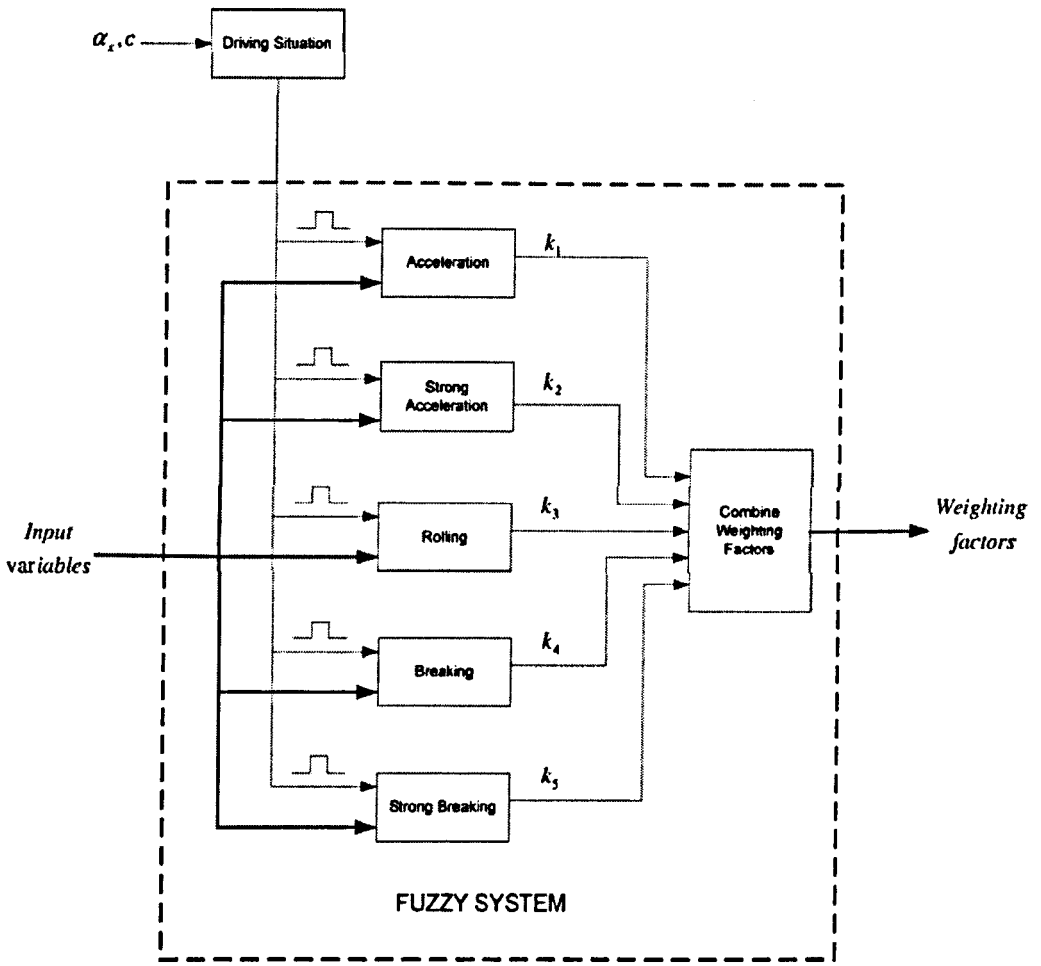

Figure 3.3: Partitioned Fuzzy Estimator

One of the widespread uses of fuzzy models is for fault diagnosis of automotive applications. Paul Frank proposed a fuzzy thresholding method for engine fault diagnosis [24], in his paper, the thresholds are made fuzzy and the fault decision is calculated using fuzzy rules. In this way the fault decision will contain information about the certainty that a fault has occurred. Ahmed Soliman proposed a fuzzy inference for diagnosis of an automotive emission control system [25]. A two-layer

fuzzy logic system is used in his experiment; the lower layer is used to detect the presence of a faulty system, and the higher layer is employed to isolate the faulty system. The enhanced performance of diagnosis system by fuzzy models has been documented. In addition, the fuzzy system can also be used to generates weighting factors in correlation to the accuracy of the wheel speed signals and the acceleration signal [5], see Figure 3.3. According to its input signals the rule base of the fuzzy estimator generates a signal reliability *zero*, *small*, *middle* or *big*. The defuzzified, crisp output values k_i are weighting factors in the range of $[0, 1]$. A comprehensive review on the contribution of fuzzy system to the improvement of modern car performances can be found in [26].

3.2.3 Neural Network Models

Neural networks are inspired by biological brain connectionism. From the past research in this area, neural networks have proved to provide very powerful solutions to a large variety of engineering problems ranging from modelling over prediction up to classification. They can not only provide a simple model structure for nonlinear system, but also capture the nonlinearity and dynamics with satisfactory accuracy. However, one of the major obstacles to engineering application of neural networks is the heavy computational burden for their online training. With the evolution of electronics, reconfigurable device such as field programmable gate arrays (FPGA) have make feasible online training for the parameters of the neural network. Therefore, NN based modelling and control have been a very active research area in recent years.

A lot of researches on engine modelling using neural network has been done over past years, most of them are based on feed-forward model including the multi-layer perception(MLP) network, the radial basis function (RBF) network, the pseudo-linear radial basis function (PLRBF) network [27] [28] [29]. In these researches, neural networks have been used as a modelling method for air path

dynamics, emission gas recirculation (EGR), AFR dynamics. The results shows that NN modelling is relatively easy and inexpensive. As some research have proved that since time series data may have autocorrelation or time dependence, the recurrent neural network models which take advantage of time dependence may be useful. In other words, feedback allows recurrent networks to achieve better predictions than can be made with a feed-forward network with a finite number of inputs. Therefore, recurrent neural network has come to be very popular as an engine modelling method recently, especially for the modelling of AFR dynamics [30] [31]. The recurrent neural network in those paper is trained on-line and off-line by the classical back-propagation algorithm with history stack adaption. The advantage of RNN based engine models is that they can make a relative accurate prediction on air fuel ratio with limited training data set.

The neural networks discussed above are often called discrete-time neural networks (DTNN) because the models based on them usually make predictions and self-updating in every sample time in practice. More recently, the training efficiency of continuous time recurrent neural network (CTRNN) have been improved significantly by the introduction of automatic differentiation (AD) technique [32]. Compared those feedforward and recurrent models under discrete time domain, CTRNN theoretically provides a better modelling performance to approximate the highly nonlinear systems, which makes it a potential method that can be extended for car engine modelling.

3.2.4 Hybrid Models

A first-principle engine model is often based on conventional thermodynamic and gas dynamic relations. The performance of a engine model can be improved by integrating the conventional model with a distributed and synergistic neural network. The structure of such neuro-hybrid model is shown in Figure 3.4 where the demodulator block has the function of providing a set of signals that describe

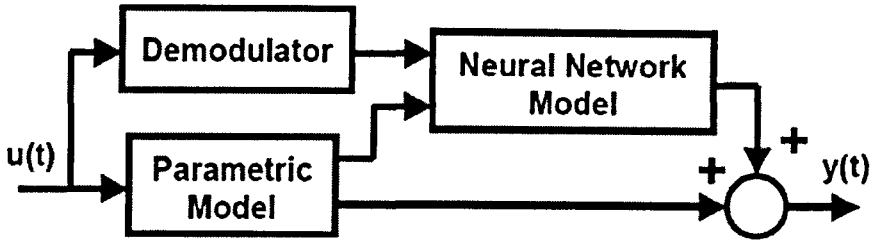

Figure 3.4: Structure of a Neuro-Hybrid Model

in a unique way the frequency content of the input signal of the dynamic system under identification [33]. The neuro-hybrid model provides a flexible structure for system identification because it implements a mechanism where the system dynamics dependent on its dominant behaviour and a priori knowledge can be weighed between the parametric and the neural network models. The experiment result in [34] [35] prove that the hybrid model can follow closely the expected results in predicting the performance of a SI engine. The hybrid model can learn the input output data relation very well and is capable to predict the output in the decided domain.

3.3 Engine Control Methods

3.3.1 LQR Control

Linear quadratic regulator (LQR) design technique is a well established methodology for the design of multi-variable control system. The characteristics of such control loops are well known and understood, and they have robustness properties which appear immediately to make them suitable for engine control. However, as shown in Chapter 2, the engine dynamics are severely nonlinear. This means that linearization process on the engine equations must be carried out before making an LQR controller.

A well-known attempt at making an LQR controller to deal with AFR regu-

lation problem is reported by Christopher Onder *et al* in paper [36]. This work is based on an isothermal model that is linearized close to idle speed and specialized for combined AFR and engine speed control in a limited speed range. Engine mapping was employed in this model. Fairly good AFR and speed control has been achieved in the region $4° \leq \alpha \leq 8°$ and from idle up to 2000rpm on an Federal Test Procedure (FTP) drive cycle using a 3.5L, 6 cylinder engine. Recently, a LQR control on homogeneous charge compression ignition (HCCI) engine has been implemented in Lund University by Maria Karlsson *et al* [37]. LQR control is realized to minimize emissions of NO_x and soot when the engine was operated using direct injection of diesel fuel only. The work focuses on the choice of feedback variables and the control structure to minimize emissions. A weighted sum of emissions is approximated by a quadratic cost function in the measured variables combustion phasing and ignition delay, which is used as a basis for LQR control. Unfortunately, the disadvantage of LQR control on automotive engine is that the linearized model is often of a high order that leads to heavy computational load, and the effective control is usually within a small range of engine operating conditions.

3.3.2 Sliding Mode Control

Variable structure control with a sliding mode is well known for its robustness properties. Such robustness properties make sliding mode control a good candidate for industrial applications. A sliding mode controller for engine speed control has been provided in good agreement with the engine dynamics in the paper [38]. A weighted sum of speed error and integral of speed error are regards as the sliding variables. Another experimental study of a sliding mode controller has been reported by Kaidantzis *et al* on an overall lambda control loop for lambda control in the paper [39]. It has been found that the lambda variations could be held to within $\pm 5\%$ during very large changes of throttle angle and widely varying engine speed. The performance of a PI controller and a sliding mode controller

are compared for the same test conditions. More research on the sliding mode control on AFR can be found in [40], however, the study is based on a simulation model rather than real engine test bed. It is difficult to compare with that been shown in other papers. In practical application, the difficulty using sliding mode control to solve AFR regulation problem is that the inherent measurement time delays exist in internal combustion engines [41], which has been given in Equation (2.16).

3.3.3 Neural Network Control

Neural networks appear to be able to implement many functions essential to control systems with higher degree of autonomy because of their ability to learn, to approximate functions, to classify patterns and because of their potential for massively parallel hardware implementation.

In Wensel's paper [21], A neural network is used to adapt the forgetting factor in an adaptive control algorithm for an SI engine. From the simulation results, the AFR control using the controller is very convincing. However, the stability analysis and robustness test on the developed system are not given in the paper. More recently, Shivaram Kamat *et al* propose a virtual air fuel ratio sensors for engine control in the paper [42]. A fully connected RNN model with inputs as manifold absolute pressure, throttle position, fuel pulse-width, spark advance and engine speed is used as a virtual oxygen sensor for the AFR prediction. As shown in Figure 3.5, the control structure consists the engine feedforward controller and a feedback controller with a NN based virtual sensor for AFR estimation. However, the virtual AFR sensor has not been able to accurately capture the sharp transients in the lambda signal. Experimental results show no significant improvement on AFR control in transient. Nevertheless, the virtual lambda sensor is still useful for its potential in realtime engine diagnostics and for fault-tolerant control development in the event of a relevant system failure.

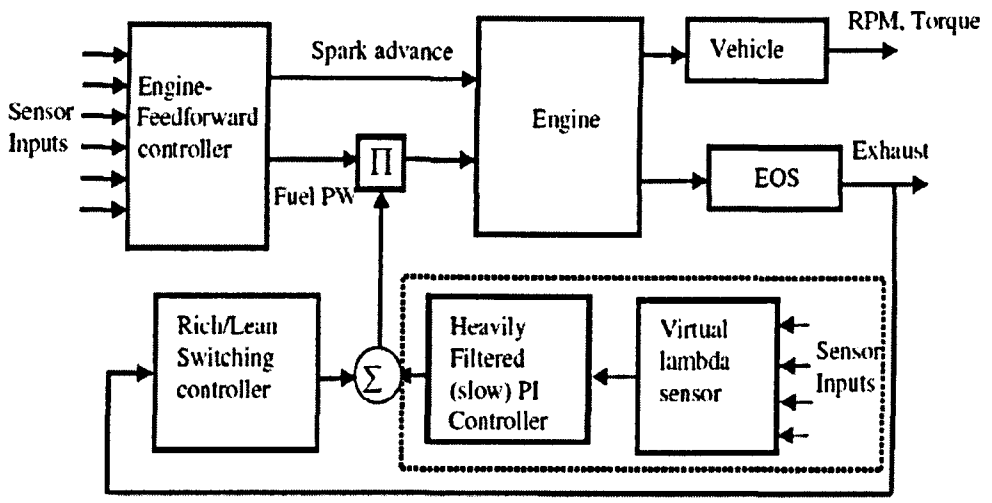

Figure 3.5: Controller design using virtual sensor feedback for more accurate lambda control

The application of neural networks as engine controllers have been found in many other areas as well, such as engine cold start [35], anti-lock braking systems [43], idle speed control [44].

3.3.4 Model Predictive Control

Model predictive control (MPC) is, perhaps, the most general way of posing the process control problem into time domain. It is a collection of optimal control, stochastic control, and control of process with dead time and multi-variable control. It has a significant and widespread impact on industrial process control. It is the only generic control technology which can deal routinely with equipment and safety constraints. The basic structure of a model predictive control is shown in Figure 3.6. From this control structure, It can see that the future evolution of the reference is known a priori, the system can react before the change has effectively been made, thus avoiding the effects of the delay in process response. To obtain

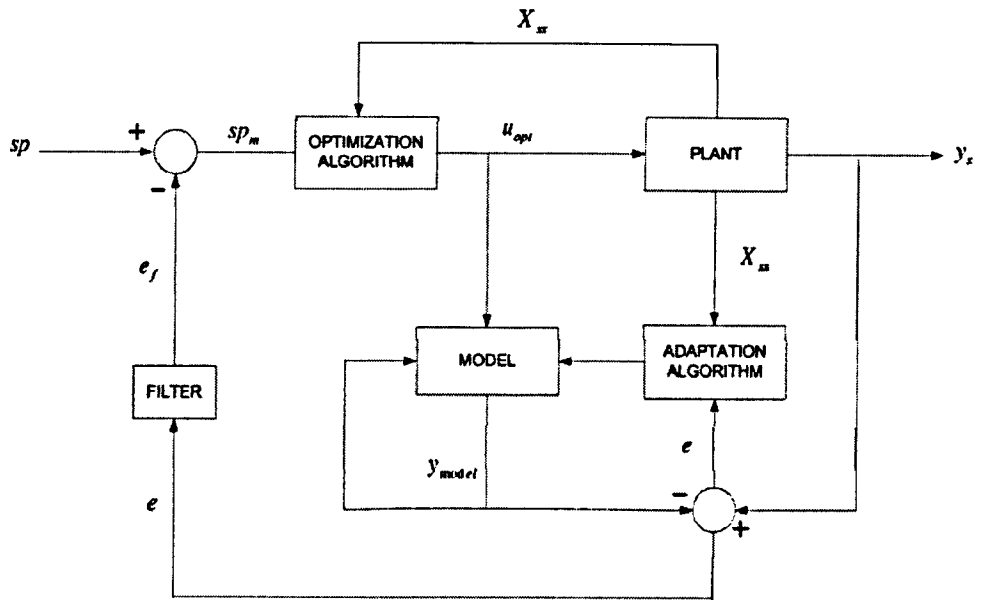

Figure 3.6: Predictive Control Structure

the control law, it is necessary to minimize the cost function J of Equation (3.3).

$$J(N_1, N_2, N_u) = \sum_{j=N_1}^{N_2} \alpha(j) \left[y_{model}(k+j|k) - sp(k+j)\right]^2 + \sum_{i=1}^{N_u} \beta(i) \left[\Delta u(k+i-1)\right]^2 \tag{3.3}$$

N_1, N_2 are the minimum and maximum prediction horizon vectors, with i_{th} element specifying the parameter for the corresponding output, N_u is the control horizon vector, with j_{th} element specifying the parameter for the corresponding input. N_1 and N_2 mark the limits of the instants in which it is desirable for the output to follow the reference. Therefore the predicted model outputs y_{model} are evaluated as past and future values of the presented inputs and outputs of the control signals. A numerical optimization can be obtained for linear models as an analytic solution for example active set method, and for nonlinear models iteratively by for example a Sequential Quadratic Programming method. The detailed description of MPC can be found in [17] [45].

Traditionally, MPC is mainly used to control the process in petro-chemical industry. This is because the slow process in the application, which makes the MPC controller have sufficient time for online optimization. The evolution of electronics has made ECU capable of advanced control algorithms, for example, neural computation and online optimization. Therefore, many researches have been found in the field of engine control using MPC. Compared with the traditional feedback control, one of the good reasons for automotive industry to adopt MPC is that the difficulty brought by the time delay between sampling and control can be easily overcome. Therefore, the control performance of SI engines during transient can be improved without complicate feed-forward compensation. From the consideration of limited computation ability of ECU, linear MPC (LMPC) is more attractive for automotive applications. The successful examples can be found in the paper [46], [47], [48]. The research in these papers includes AFR regualation, cruise control, waste-gate control of diesel engine, and all the experiments are based on LabView or dSPACE prototyping system for real time control. Better control performance has been achieved if compared with PI controlled system. However, the experimental results also show that LMPC is not effective in a wide range of engine operating conditions due to the limitation of linear models used in these application. Replacing the linear internal model in MPC scheme by a nonlinear one can be a potential method to solve this problem. As neural networks as one class of nonlinear model have been used with success in a wide range of application, more recently, lots of research interests have shown on neural networks based nonlinear MPC for AFR [12] [13] [49] [29]. Using these NN based model predictive control, significant fuel consumption savings have been obtained and good control performances have been observed. Nevertheless, the need of fast online calculations to put NMPC into practical use is in imperious demands because these NMPC schemes are only tested on computer simulation models and few reports have been found for real engine control application due

to the expensive computational load on optimization.

3.4 Summary

This chapter reviews the previous research on several non-analytic modelling technologies for automotive engines. These modelling methods are important because the advanced controls are most often model based, for instance, sliding mode control and model predictive control. The models are usually obtained by system identification using the engine input-output data. However, a careful analysis of the a priori knowledge about the engine dynamics is still important to have an effective non-analytic engine model as it involves the selection of the model inputs, the type of excitation signals, the model order, etc. Production ECU employs look-up tables for engine modelling and control. This kind of static models cannot maintain its robustness and lack of the ability to deal with the performance degradation under time-varying conditions. Therefore, more advanced engine control methods need to be developed to improve the performance of ECU. Among the control methods that have been reviewed in this literature, the neural network based control and model predictive control seem more promising for AFR regulation. For example, NN-based feed-forward controller can obtain the input-output mapping by neural computation. The deep understanding on the intake manifold dynamics is essential to build such model. Adaptation algorithm can adjust the neural parameters online, and therefore improve the system robustness against changing conditions such as engine wear, or components replacement. Model predictive control generates the control signal according to the predicted system outputs, which can solve the difficulty caused by the large time delay of oxygen sensors. A good performance of AFR control in both transient and steady state is expected to be obtained by MPC. These two methods can be the potential control schemes to replace the look-up table and traditional feedback control for the production ECU of next generation.

Chapter 4

Neural Modelling Techniques

4.1 Introduction

Neural networks are very powerful tools for modelling nonlinear processes. Most kind of networks are described as black box systems, but that is not completely true. In general neural networks can be stated as grey boxes, caused by the fact that at least some a priori knowledge have to be known. For example, the input variables and the best number of neurons used in a specific application have to be defined and therefore the black box becomes grey. A further advantage is the ability to react in time for time-varying processes by adaptation. This is a very important fact, especially for engines. There are two types of neural network used in this study. One is radial basis function neural network (RBFNN) , and the other is diagonal recurrent neural network (DRNN). Although there are some other types of NN, such as multi-layer perceptrons network (MLPN), that can be used to model dynamic systems, the above two types of NN have been chosen in this study due to their simple structures and that they are easy to train.

4.2 Radial Basis Function Neural Networks

RBFNN views the design of a neural network as a curve-fitting problem in a high-dimensional space. In contrast to the MLPN, RBFNN utilizes a radial con-

struction mechanism. This gives the hidden layer parameters of RBFNN a better interpretation than for the MLPN. Therefore, the algorithm to adjust the output layer weights, due to minimising the squared error between actual and estimated output, is a linear learning rule [50]. The very well developed linear learning algorithms, exhibit much faster convergence than nonlinear algorithms, such as least square (LS). The disadvantage is that, if compared with other feed-forward neural networks, for example MLPN, a larger network is needed to achieve the same modelling performance.

4.2.1 RBFNN Structure

The RBFNN, as shown in Figure 4.1, consists of three layers: input layer, hidden layer and output layer, where $x = [x_1, x_2, \cdots, x_n]^T \in \mathcal{R}^n$ is the input vector,

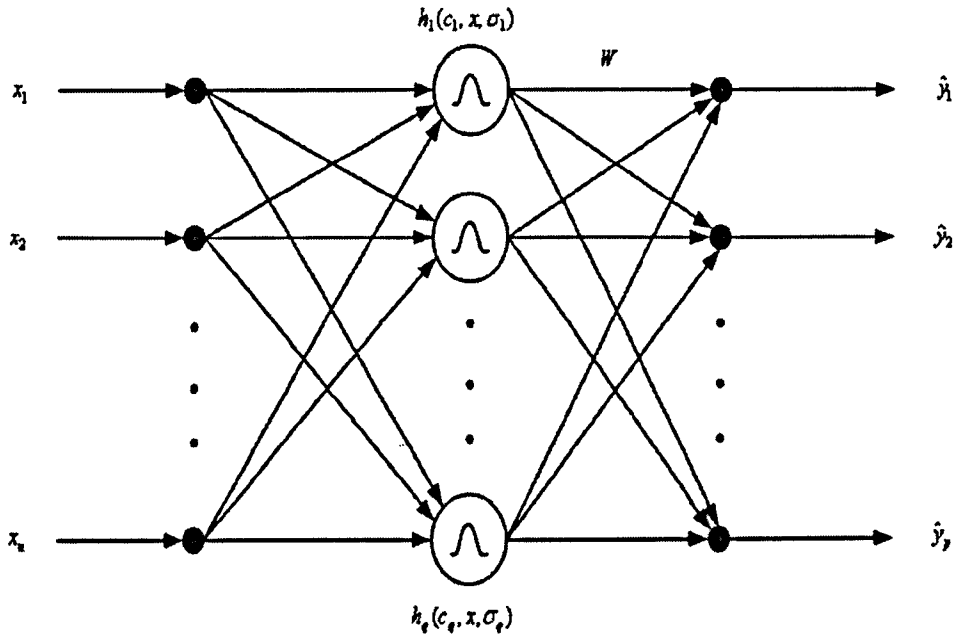

Figure 4.1: RBFNN Structure

$h = [h_1, h_2, \cdots, h_q]^T \in \mathcal{R}^q$ is the hidden layer output vector, $W(k) \in \mathcal{R}^{p \times q}$ is the weight matrix with entry w_{ij}, which is the weight linking the jth node in the hidden layer to the ith node in the output layer, and $\hat{y} = [\hat{y}_1, \hat{y}_2, \cdots, \hat{y}_p] \in \mathcal{R}^p$ is the output vector of the RBFNN.

In mathematical terms, the following equations are used to describe the RBFNN.

$$\hat{y}(k) = W \cdot h(k) \tag{4.1}$$

$$h(k) = f[z(k)] \tag{4.2}$$

$$z_i(k) = \sqrt{[x(k) - c_i]^T [x(k) - c_i]} = \| x(k) - c_i \| \tag{4.3}$$

where $i = 1, 2, \ldots, q$. $c_i \in R^n$ is the i_{th} centre in the input space, and $f[.]$ is the nonlinear activation function in hidden layer. The gaussian basis function given by

$$f[z(k), \sigma] = e^{\frac{-z^2(k)}{\sigma^2}} \tag{4.4}$$

is chosen in this research, where σ is a positive scalar called width, which is a distance scaling parameter to determine over what distance in the input space the unit will have a significant output.

The RBF neural network models are used in this research to predict system outputs. The procedure of RBFNN modelling and prediction is to determine network inputs according to system dynamics; data collection and scaling; network training and validation; using the network to do prediction. The network training includes determining the number of centres, q, appropriate centres and widths, c_i and σ_i, $i = 1, \cdots, q$ from the training data set; obtaining the weights W by training data, and validating the network by the test data. Finally the network model is used to do prediction. In the following three training algorithms are briefly introduced.

4.2.2 Training Algorithms

K-means Algorithm

The K-means clustering algorithm is used to choose the centres of RBFNN from a set of training data in this research. Its objective is to minimise the sum squared distances from each input data to its closest centre so that the data is adequately covered by the activation functions $f[.]$.

Procedure:

Step 1: Choose q initial cluster centres $c_1(1)$, $c_2(1)$, ..., $c_q(1)$.

Step 2: At the k_{th} iteration step, distribute the sample $\{ x \}$ into $S_j(k)$ among the q cluster domains. $S_j(k)$ denotes the set of samples whose cluster is $c_j(k)$

$$x \in S_j(k) \qquad if \parallel x - c_j(k) \parallel < \parallel x - c_i(k) \parallel \tag{4.5}$$

where $j = 1, 2, ..., q$ and $i = 1, 2, ..., j\text{-}1, j\text{+}1, q$.

Step 3: Update the cluster centres.

$$c_j(k+1) = \frac{1}{N_j} \sum_{j}^{N_j} S_j(k) \tag{4.6}$$

where N_j is the number of elements in $S_j(k)$.

Step 4: Repeat *Step 2* to *Step 3* until $c_j(k+1) = c_j(k)$.

ρ-Neighbourhood Method

The RBFNN width σ of each unit is computed by the ρ-Nearest neighbours method. The guideline is that the excitation of each node should overlap with some other nodes(usally closest) so that a smooth interpolation surface between nodes is obtained. To achieve this each hidden node must activate at least one other hidden node to a significant degree. Therefore, the width is selected so that σ is greater than the distance to the nearest unit centre.

$$\sigma_i = [\frac{1}{\rho}\sum_{j=1}^{\rho} \| c_i - c_j \|^2]^{\frac{1}{2}} \tag{4.7}$$

where $i = 1, 2, ..., q$, c_j is the ρ-nearest neighbours of c_i. For nonlinear function approximation ρ depends on the problem and requires to be experimented.

Batch Least Squares Algorithm

The least squares method is one of the most commonly used for linear parameter adaptation. The batch adaptation can be described as following. Assume a system is mathematically described as Equation (4.1), at sample time k, $\hat{Y} \in \mathcal{R}^{k \times p}$ is defined as:

$$\hat{Y}(k) = \begin{bmatrix} \hat{y}^T(1) \\ \vdots \\ \hat{y}^T(k) \end{bmatrix} = \begin{bmatrix} h_1(1) & \cdots & h_q(1) \\ \vdots & \ddots & \vdots \\ h_1(k) & \cdots & h_q(k) \end{bmatrix} \begin{bmatrix} w_{11} & \cdots & w_{p1} \\ \vdots & \cdots & \vdots \\ w_{1q} & \cdots & w_{pq} \end{bmatrix} = H(k)W^T \tag{4.8}$$

where

$$H(k) = \begin{bmatrix} h^T(1) \\ \vdots \\ h^T(k) \end{bmatrix} \tag{4.9}$$

$$h(i) = \begin{bmatrix} h_1(i) \\ \vdots \\ h_q(i) \end{bmatrix}, \quad i = 1, 2, \cdots, k \tag{4.10}$$

Least squares cost function is defined as below:

$$J = \sum_k E^2(k) = \left[Y(k) - \hat{Y}(k)\right]^T \left[Y(k) - \hat{Y}(k)\right] \tag{4.11}$$

where $Y(k) \in \mathcal{R}^{k \times p}$ are the desired output

$$Y(k) = \begin{bmatrix} y^T(1) \\ \vdots \\ y^T(k) \end{bmatrix} \tag{4.12}$$

By matrix vector differentiation rules,

$$\frac{dJ}{dW} = -2H(k)^T Y(k) - 2H(k)^T H(k) W \tag{4.13}$$

The optimum condition of the minimum J is $dJ/dW = 0$, thus the LS algorithm are

$$W = [H^T(k)H(k)]^{-1} H^T(k)Y(k) \tag{4.14}$$

Recursive Least Squares Algorithm

Recursive least squares(RLS) algorithm is a recursive form of the Least Squares(LS) algorithm. It evaluates for each new sample the parameter matrix W newly. The basic idea of RLS algorithm is to compute the new parameter estimate $W(k)$ at discrete time steps k by adding some correction information to the previous parameter estimate $W(k-1)$ at time instant $k-1$. It is used to find the RBF network weights W, which can be summarised as follows [51]:

$$y_p(k) = y_c(k) - W(k-1)h(k) \tag{4.15}$$

$$g_z(k) = \frac{P_z(k-1)h(k)}{\mu + h^T(k)P_z(k-1)h(k)} \tag{4.16}$$

$$P_z(k) = \mu^{-1}[P_z(k-1) - g_z(k)h^T(k)P_z(k-1)] \tag{4.17}$$

$$W(k) = W(k-1) + g_z(k)y_p(k) \tag{4.18}$$

where $W(k)$ and $h(k)$ represent the RBF network weights and activation function outputs at iteration k, $y_c(k)$ is the process output vector, P_z and g_z are middle

terms. μ here is called *forgetting factor* ranging from 0 to 1 and is chosen to be 1 for off-line training. The parameters g_z, w and P_z are updated orderly for each sample with the change of the activation function output $h(k)$.

4.3 Diagonal Recurrent Neural Networks

Recurrent NNs (RNN) have important capabilities that are not found in feed-forward networks, such as attractor dynamics and the ability to store information for later use. Of particular interest is their ability to deal with time varying input or output through their own natural temporal operation. Thus, the RNN is a dynamic mapping and is better suited for dynamic systems modelling than the feed-forward networks. Considering the computation burden of MPC for fast dynamic system, the DRNN, instead of fully connected recurrent neural networks (FRNN), is used in this study. The DRNN has one hidden layer that is comprised of self-recurrent neurons from their own output only. Since there is no inter-links among neurons in the hidden layer, DRNN has considerably fewer weights than FRNN and the network is simplified considerably

4.3.1 DRNN Structure

The DRNN consists of one input layer, one hidden layer and one output layer. The basic DRNN structure is shown in Figure 4.2.

Here, the input vector at sample instant k is $x(k) = [x_1(k) \cdots x_n(k)]^T$ with n the number of input variables, the hidden layer vector $h(k) = [h_1(k) \cdots h_q(k)]^T$ with q the number of hidden layer nodes, and output vector $\hat{y}(k) = [\hat{y}_1(k) \cdots \hat{y}_p(k)]^T$ with p the number of output variables. there are three weighting matrices, $W^h(k) \in \mathcal{R}^{q\times(n+1)}$ connected between input layer and hidden layer, $W^y(k) \in \mathcal{R}^{p\times(q+1)}$ connected between hidden layer and output layers, and $W^d(k) \in \mathcal{R}^{v\times q}$ connected between hidden layer output and hidden layer input as feedback. These matrices

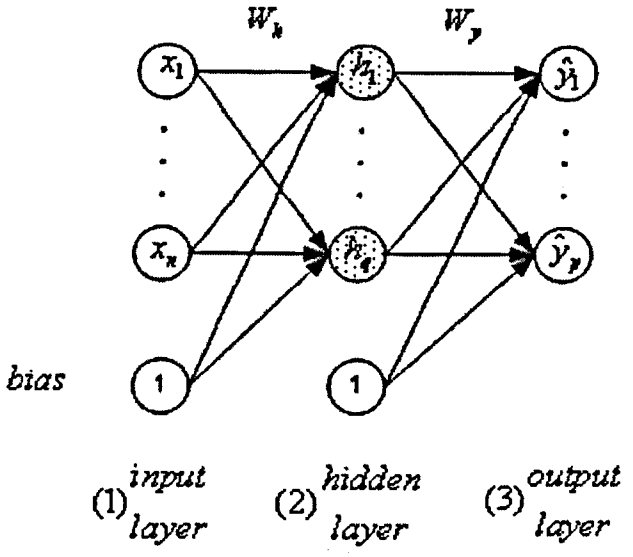

Figure 4.2: DRNN Structure

are defined as follows.

$$W^h(k) = [w^h_{i,j}],\ i = 1, \cdots, q,\ j = 1, \cdots, n+1$$
$$W^y(k) = [w^y_{i,j}],\ i = 1, \cdots, p,\ j = 1, \cdots, q+1$$
$$W^d(k) = [w^d_{i,j}],\ i = 1, \cdots, v,\ j = 1, \cdots, q$$

Here v is the maximum number of delayed sample instants in the feedback as shown in Figure 4.3.

The recurrent structure within each hidden layer node is the feedback from the node output to its input through different orders of time delay, as shown in Figure 4.3. The DRNN output can be calculated from its input and weights as follows:

$$\hat{y}(k) = W^y \begin{bmatrix} h(k) \\ 1 \end{bmatrix} \tag{4.19}$$

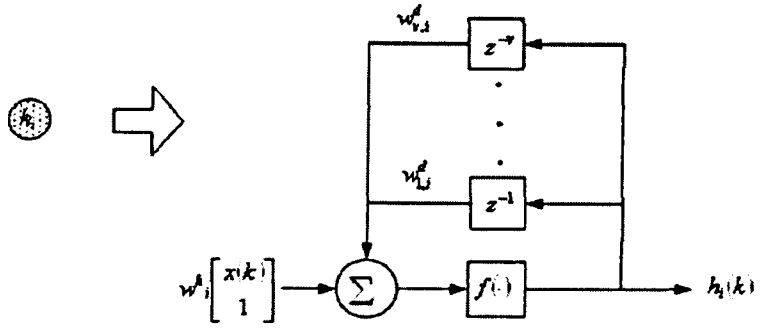

Figure 4.3: The Rcurrent Structure within the i_{th} Hidden Layer Node

$$h(k) = f[z(k)] \tag{4.20}$$

$$z(k) = W^h \begin{bmatrix} x(k) \\ 1 \end{bmatrix} + \left[diag(w_1^d) \cdots diag(w_v^d) \right] \begin{bmatrix} h^d(k-1) \\ \vdots \\ h^d(k-v) \end{bmatrix} \tag{4.21}$$

$$diag(w_i^d) = \begin{bmatrix} w_{i,1}^d & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & w_{i,q}^d \end{bmatrix} \qquad i = 1, \cdots, v \tag{4.22}$$

$$h^d(k) = h(k) \in \mathcal{R}^q, \qquad feedback\ after\ activation\ function \tag{4.23}$$

where $f(\cdot)$ in Equation (4.20) is the nonlinear activation function in hidden layer. The typical hidden layer activation functions used in DRNN are sigmoid and hyperbolic tangent function. In the investigation of process modelling with DRNN, only sigmoid activation function is chosen as the nonlinear transfer function in DRNN.

4.3.2 Training Algorithms

Dynamic Back-Propagation Algorithm

Because the weights in the feedback loop are nonlinearly related to the network output, optimisation algorithms for linear systems cannot be used. Here, training of these weights is achieved using a so-called dynamic back-propagation algorithm. Let $y(k)$ and $\hat{y}(k)$ be the actual responses of the plant and the output of the DRNN model, then an error function for a training cycle for DRNN can be defined as:

$$E_m = \frac{1}{2}[y(k) - \hat{y}(k)]^2 \tag{4.24}$$

The gradient of error simply becomes

$$\frac{\partial E_m}{\partial W} = -e_m(k)\frac{\partial \hat{y}(k)}{\partial W} \tag{4.25}$$

where $e_m(k) = y(k) - \hat{y}(k)$ is the output error between the plant and the DRNN.

Given the DRNN shown in Figure 4.2 and described by Equations (4.19)-(4.23), the output gradients with respect to output, recurrent and input weights, respectively, are given by

$$\frac{\partial \hat{y}(k)}{\partial W^y} = h(k) \tag{4.26}$$

$$\frac{\partial \hat{y}(k)}{\partial W^d} = W^y P(k) \tag{4.27}$$

$$\frac{\partial \hat{y}(k)}{\partial W^h} = W_j^y Q(k) \tag{4.28}$$

where $P(k) \equiv \partial h(k)/\partial W^d$ and $Q(k) \equiv \partial h(k)/\partial W^h$ and satisfy

$$P(k) = f'(z)[h(k-1) + W^d P(k-1)], \quad P(0) = 0 \tag{4.29}$$

$$Q(k) = f'(z)[x(k) + W^d Q(k-1)], \quad Q(0) = 0 \tag{4.30}$$

The weights can now be adjusted following a gradient method, i.e., the update rule of the weights becomes

$$W(k+1) = W(k) + \eta\left(-\frac{\partial E_m}{\partial W}\right) \tag{4.31}$$

where η is the learning rate and $\eta = \eta^h, \eta^d, \eta^y$ respectively for the corresponding weight matrix. Equation (4.24)-(4.31) define the dynamic back-propagation algorithm (DBP) for DRNN.

The update rules call for a proper choice of the learning rate η. If let η^h, η^d, and η^y be the learning rate for DRNN weights W^h, W^d, and W^y respectively, then, the DBP algorithm converges if $0 <\mid W_j^d \mid< 1$, $j = 1, 2, \cdots, v$ and the learning rate are chosen as [52]:

$$0 < \eta^h < \frac{2}{q} \tag{4.32}$$

$$0 < \eta^d < \frac{2}{q}\left[\frac{1}{W^y_{max}}\right]^T \tag{4.33}$$

$$0 < \eta^h < \frac{2}{n+q}\left[\frac{1}{W^y_{max} \cdot x_{max}}\right]^2 \tag{4.34}$$

here, $W^y_{max} := max_k\|W^y(k)\|$, $x_{max} := max_k\|x(k)\|$ and $\|\cdot\|$ is the sup-norm.

4.4 Automatic Differentiation Technique

4.4.1 Introduction

There are many ways to obtain the derivatives of a mathematical function. Straight forward hand calculations is the first to come in mind. This course is usually taken in problems of small size. If the function is not simple, the number of variables is large, or the input and output range of values is large, hand calculation becomes nearly impossible and prone to errors that are difficult to

debug. In the training stage of DRNN, the derivative of error between DRNN and plant against all the weights, is required for DBP algorithm. As shown in equations (4.24)-(4.31), the deduction of this derivative is very complex, due to the internal dynamics of DRNN. Therefore, on the practical application side, it is preferable to obtain this derivative by computational methods. In general, there are three important methods for finding the derivatives. These are, the numerical differentiation method, the symbolic method, and the recently developed method of automatic differentiation. The three methods are introduced in the next sections with special emphasis in the AD tool.

4.4.2 Numerical Differentiation

The most common alternative to hand coding is the numerical approximation of derivatives by Finite Difference (FD) formula. A simple formula is constructed from the expansion of $f(x)$ in Taylor series truncated after the first order term:

$$f(x) = f(x_k) + \Delta x \cdot \left.\frac{\partial f}{\partial x}\right|_{x=x_k} + \mathcal{O}(\Delta x^2) \tag{4.35}$$

where $\Delta x = x - x_k$ and it is some very small positive number. Evaluated at $x = x_k - \Delta x$, then a good approximation of the derivative is computed easily as:

$$\left.\frac{\partial f}{\partial x}\right|_{x=x_k} = \frac{f(x) - f(x_k - \Delta x)}{\Delta x} + \mathcal{O}(\Delta x^2) \tag{4.36}$$

The order of the approximation is controlled by the term at which the series is truncated that is the last term in Equation (4.36). Note that only function evaluations are needed to calculate the derivative. Thus, the coding of the algorithm is very simple and existing codes can be used.

More accurate derivative can be computed using Centered Finite Differences(CFD)

method as:

$$\left.\frac{\partial f}{\partial x}\right|_{x=x_k} = \frac{f(x+\Delta x) - f(x-\Delta x)}{2\Delta x} + \mathcal{O}(\Delta x^3) \tag{4.37}$$

which usually gives better approximation, but cost an additional function evaluation. The main disadvantage of FD and CFD approaches lies with the tradeoff between truncation error and roundoff error. As the truncated Taylor series expansion is only valid in the neighbourhood of the expansion point x_k, small values of Δx tend to reduce the truncation error. Ideally, the exact derivative is the limit of these formula above, when Δx tends to zero. However, very small values of Δx increase the roundoff error. Finding the best Δx requires numerous executions of the program, and even then the computed derivatives are just approximations. Other disadvantages of this approach are the instability of higher order differentiation formula and the computational cost of the techniques, approximately $n+1$ times the computational effort associated with evaluation of the function itself [53].

4.4.3 Symbolic Differentiation

Symbolic differentiation (SD) is a computer aided analog to analytical or hand differentiation employing a graph theoretical approach. The formula representation of a function is transformed into a formula representation for its derivative, that is either interpreted or further transformed into a program in a common programming language. In principle, evaluation of these formula gives exact values of the derivatives of the function. Symbolic differentiation only incurs roundoff error resulting from the individual floatingCpoint operations.

Symbolic differentiation, usually performed in computer algebra packages like Maple and Mathematica [54], is unable to deal with branches, loops and subroutines intrinsic in computer codes. For every binary operator (except + or -), the derivative expression is likely to double in size, leading to a combinatorial

explosion effect. Due to this effect, the resulting derivative code is difficult to manipulate and to be used for practical software applications. The computational cost of SD is almost impossible to predict. It generally grows enormously with function complexity. Instead, Automatic Differentiation is bounded in terms of the number of independent and dependent variables.

4.4.4 Automatic Differentiation

Automatic Differentiation (or AD), which is developed for the automatic computation of derivatives, is a new approach to obtain analytical derivatives of programs (possible containing conditional statement, loops etc.). AD is the numerical computation of exact values of the derivative of a function at a given argument value. It just like FD, requires only the original program f. But instead of executing f on different sets of inputs, it builds a new, augmented, program f', that computes the analytical derivatives along with the original program. This new program is called the differentiated program. Precisely, each time the original program holds some value v, the differentiated program holds an additional value dv, the differential of v. Moreover, each time the original program performs some operations, the differentiated program performs additional operations dealing with the differential values. So, it decomposes the model into a series of elementary functions ($\times$, $/$, $sin(\cdot)$, *etc.*), applies the simple rules of differentiation (product rule, quotient rule, etc.) to evaluate the partial derivatives of the elementary functions, and then accumulates them with the chain rule to obtain the derivatives of the program. The resulted derivative values are obtained without generating a formula for the derivatives, thus avoiding the unnecessary overhead of symbolic differentiation and the truncation error inherent in FD formulas [55].

Table 4.1: Evaluation of f and its derivative

Evaluation of f(x,y)	Evaluation of $f'(x,y)$
$v_1 = y$	$dv_1 = dy$
$v_2 = {v_1}^3$	$dv_2 = 3{v_1}^2 dv_1$
$v_3 = xv_2$	$dv_3 = xdv_2 + dxv_2$
$v_4 = 0.5v_3$	$dv_4 = 0.5dv_3$
$v_5 = x + y$	$dv_5 = dx + dy$
$v_6 = sin(v_5)$	$dv_6 = cos(v_5)dv_5$
$v_7 = exp(v_6)$	$dv_7 = exp(v_6)dv_6$
$f(x,y) = v_4 - v_7$	$df = dv_4 - dv_7$

To explain further, consider the function $f(x,y)$ represented below:

$$f(x,y) = 0.5xy^3 - exp(sin(x+y)) \tag{4.38}$$

The partial derivatives of this function are easily obtained and equal to:

$$\frac{\partial f}{\partial x} = 0.5y^3 - cos(x+y) \cdot exp(sin(x+y)) \tag{4.39}$$

$$\frac{\partial f}{\partial y} = 1.5xy^2 - cos(x+y) \cdot exp(sin(x+y)) \tag{4.40}$$

Using only binary operations, this function would be represented as shown in Table 4.1. Differentiation each line of the code, one would get the code to generate the derivative without the formula of the derivative, shown as equations (4.39) and (4.40). In order to calculate the partial derivative in respect to x the vector $[dx, dy]^T$ is set to $[1,0]^T$, meaning that $\partial y/\partial x = 1$. Analogously, to calculate the partial derivative in respect to y, the vector $[dx, dy]^T$ is set to $[0,1]^T$. In Table it is shown that the evaluation of the formulas on Table would lead to the same expressions of equations (4.39) and (4.40).

Although great advances have been made in symbolic differentiation of formulas, AD generally requires less memory and CPU time, and also applies to functions defined by computer programs or subroutines for which no formula may be available.

Table 4.2: Evaluation of the partial derivative of f

Evaluation of $\partial f/\partial x$	Evaluation of $\partial f/\partial y$
$dx = 1, dy = 0$	$dx = 0, dy = 1$
$dv_1 = dy = 0$	$dv_1 = dy = 1$
$dv_2 = 3(v_1)^2 dv_1 = 0$	$dv_2 = 3(v_1)^2 dv_1 = 3y^2$
$dv_3 = dx v_2 = v_2 = y_3$	$dv_3 = x dv_2 = 3xy^2$
$dv_4 = 0.5 dv_2 = 0.5y^3$	$dv_4 = 0.5 dv_3 = 0.5(3xy^2) = 1.5xy^2$
$dv_5 = dx + dy = 1$	$dv_5 = dx + dy = 1$
$dv_6 = cos(v_5) dv_5 = cos(x + y)$	$dv_6 = cos(v_5) dv_5 = cos(x + y)$
$dv_7 = exp(sin(x + y)) cos(x + y)$	$dv_7 = exp(sin(x + y)) cos(x + y)$
$df = 0.5y^3 - cos(x + y) exp(sin(x + y))$	$df = 1.5xy^2 - cos(x + y) exp(sin(x + y))$

There are two basic modes of operation in automatic differentiation, one is the forward mode and the other is the reverse mode. In the forward mode, the derivatives are propagated through the computation using the chain rule. This is the most classical approach, the differentiation machinery behaves as a human who would augment the code by additional instructions computing the derivatives and reCusing the shared expressions assigned to temporary variables. Consider a function, $y = g(f(x))$ consisting of two operations: $v = f(x)$ and $y = g(v)$. In forward mode, by applying the chain rule, $\hat{y} = dy/dx$ can be evaluated in the sequence: $\dot{x} = 1$, $\dot{v} = f'(x)\dot{x}$ and $\dot{y} = g'(v)\dot{v}$. In forward mode, a function and its derivatives can be evaluated in parallel. Note that the same method is used in Table 4.1. This mode is easy to understand and implement, and requires computational effort proportional to $n \times m$, where n is the number of independent variables and m the dimension of the function component.

In the reverse mode, the intermediate derivatives are computed in the reverse order, from the final results down to the independent variables. The reverse mode evaluation is based on the definition of adjoint, $\bar{v} = dy/dv$. After evaluating the sequence, $v = f(x)$ and $y = g(v)$ with all intermediate results recorded, the adjoints are evaluated in a reverse sequence: $\bar{y} = 1$, $\bar{v} = \bar{y}g'(v)$ and finally, $dy/dx = \bar{x} = \bar{v}f'(x)$. The reverse mode requires saving the entire computa-

tion trace, since the propagation is done backwards through the computation, and hence, the partial derivatives need to be stored for derivative computation. Hence the reverse mode can be prohibitive due to memory requirements [56]. However, the reverse mode is better for computing multiCdimensional gradients of a function, because the computational effort for it is proportional with m, the length of the code list, and independent of n, the number of independent variables. In fact, when the number of m is much less than n such as the objective function of an optimisation problem, evaluation in reverse mode is vastly more efficient than in forward mode. This can result in significant saving in computational time [57].

There are aspects to be considered other than merely the computational cost when discussing AD modes. Reverse mode implementation is quite more sophisticated and may employ complex structures of indirect addressing. That may prevent vectorisation of the final code [58]. Hence, available AD codes employ a combination of both strategies in order to balance complexity and computational cost.

4.4.5 Implementation of AD Using ADOL-C library

The *C++* package ADOL-C (Automatic Differentiation by Overloading in *C++*) proposed by Griewank et al. [59], facilitates the evaluation of first and higher derivatives of vector functions that are defined by computer programs written in *C* or *C++*. The resulting derivative evaluation routines may be called from *C/C++*, Fortran, or any other language that can be linked with *C*.

In the thesis (where the derivatives will be required) ADOL-C is linked with *MATLAB* via *mex* warp for derivatives evaluation. ADOL-C facilitates the simultaneous evaluation of arbitrarily high directional derivatives and the gradients of these Taylor coefficients with respect to all independent variables. Relative to

the cost of evaluating the underlying function, the cost for evaluating any such scalar-vector pair grows as the square of the degree of the derivative but is still completely independent of n, the number of vector functions component, and m, the number of independent variables.

For the reverse propagation of derivatives, the whole execution trace of the original evaluation program must be recorded, unless it is recalculated in piece as advocated in [60]. In ADOL-C, this potentially very large data set is written first into a buffer array and later into a file if the buffer is full or if the user wishes a permanent record of the execution trace. In either case, it refer to the recorded data as the tape. The user may generate several tapes in several named arrays or files. During subsequent derivative evaluations, tapes are always accessed strictly sequentially, so that they can be paged in and out to disk without significant runtime penalties. If written into a file, the tapes are self-contained and can be used by other *Fortran*, *C* or *C++* programs [61][62].

4.5 SI Engine Modelling Using Neural Networks

Neural network methods have proven to be powerful tools in modelling of non-linear dynamic processes. In this research, neural network is used for engine dynamics modelling, because it is a universal tool for system identification, and can be used for a large number of different applications for engine control. Moreover, for production ECU in automotive industry, memory restrictions are still an important issue. By fully utilizing micro-controller for the neural computation, the system memory cost can be reduced to a certain extent. The flow chart in Figure 4.4 shows the procedure for system identification, in which the work is done in steps [50].

The first step in the engine modelling is the generation of a suitable training

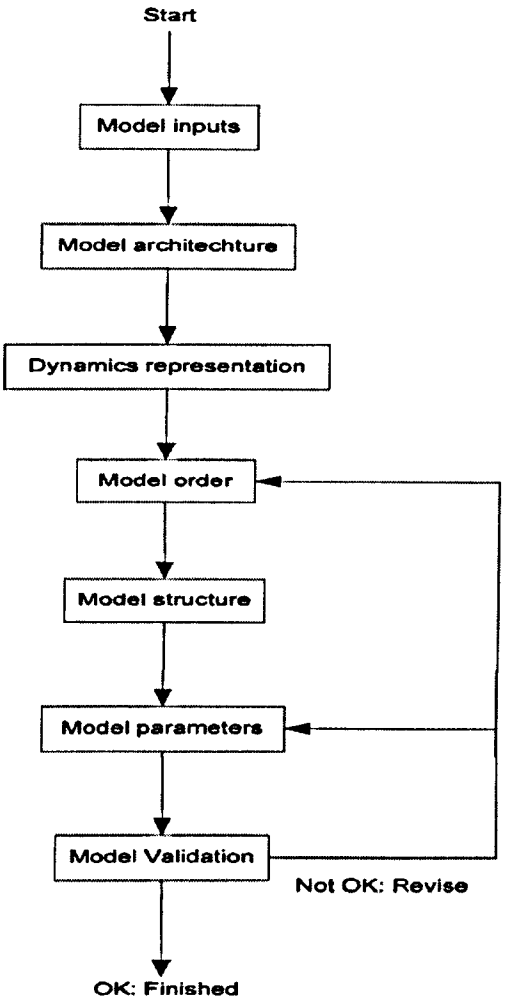

Figure 4.4: The System Identification Loop

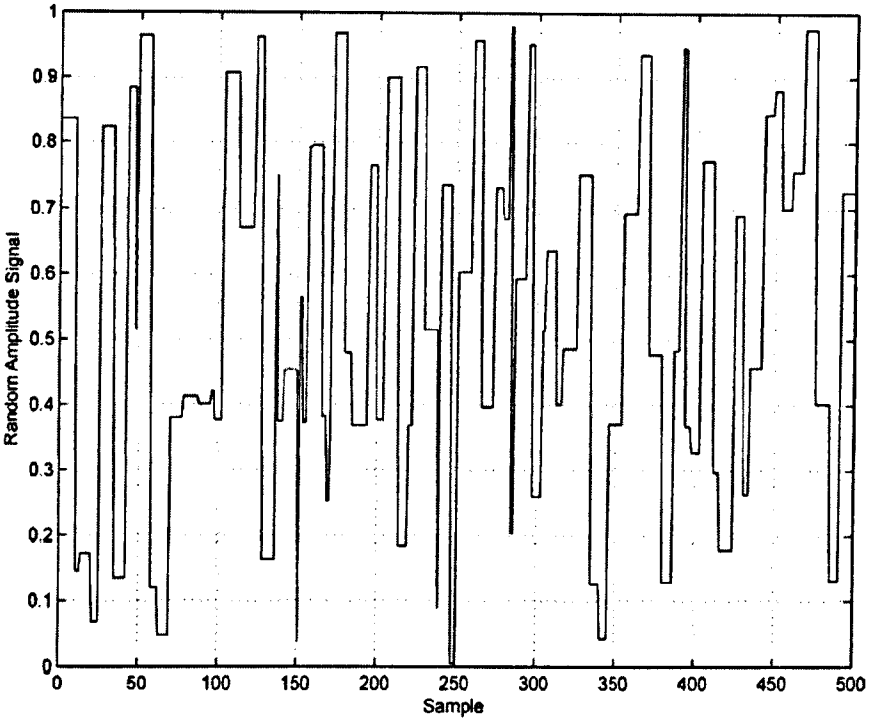

Figure 4.5: Random Amplitude Signal

data set. As the training data will influence the accuracy of the NN modelling performance, the objective of experiment design on training data is to make the measured data become maximally informative, subject to constraints that may be at hand. As such the input signals are required to excite the dynamic modes of the process at different frequencies while also ensuring that the training data adequately cover the specified operating region. A hybrid excitation signal for NN training was proposed by Lightbody and Irwin [63]. A persistently exciting input signal may be sufficient in linear system identification, but this is not the only consideration for the identification of non-linear systems. The process modelling of a NN model consisting of two parts, ***(1)*** the capturing of the dynamics of the process and ***(2)*** the approximation of the underlying non-linear vector function. Random amplitude signal (RAS) is chosen for this research, which is shown in Figure 4.5. Before training or validating the neural network using RAS, all input and output data obtained from MVEM by simulation will be scaled to the range of $[0, 1]$. The linear scale is used as follows:

$$u_s(k) = \frac{u(k) - u_{min}}{u_{max} - u_{min}} \tag{4.41}$$

$$y_s(k) = \frac{y(k) - y_{min}}{y_{max} - y_{min}} \tag{4.42}$$

where u_{min} and u_{max} are the minimum and maximum inputs among the data set, while u_s is the scaled input. The same is for the output.

To testify validity of the NN model, the engine data is usually divided into two parts, the first data set is used for training neural network and the left is used for NN model validation. Generally, the modelling error of the training data set is often smaller than the test data set. In this research, the modelling and control performance are evaluated by mean absolute error (MAE), which is given as the following:

$$MAE = \frac{1}{N}\sum_{k=1}^{N} \|f(k) - y(k)\| = \frac{1}{N}\sum_{k=1}^{N} \|e(k)\| \tag{4.43}$$

As the name suggests, the MAE is an average of the absolute errors $e(k) = f(k) - y(k)$. For modelling performance, $f(k)$ is the prediction by NN model and $y(k)$ the output of SI engine. In term of the evaluation of control performance, $f(k)$ is the engine output and $y(k)$ the corresponding set-point value.

4.6 Summary

In this chapter, two types of neural networks used for SI engine modelling in this research are described, including RBFNN and DRNN. A brief overview of the structures of RBFNN and DRNN, and the corresponding general training algorithms are introduced.

In addition, to improve the performance of dynamic back-propagation algorithm for DRNN, automatic differentiation is implemented, and a short introduction on

this technique is provided.

Finally, the NN modelling procedure and the generation of training and validation data sets is described.

Chapter 5

RBFNN based Feedforward-Feedback Control

5.1 Introduction

Due to the high nonlinearity of automotive engine system dynamics and large disturbance, the traditional feedback control on AFR by itself, or in conjunction of look-up table, is not satisfactory. Significant improvements are expected to achieve by adding feed-forward control. The desired engine port fuel mass flow $\dot{m}_f$ is described by the following equation:

$$\bar{\dot{m}}_f = \frac{\dot{m}_{ap}}{\lambda} \tag{5.1}$$

Here, λ is the air/fuel ratio. When tight control of NO_x, HC and CO emissions is required, operation of the automotive engine with a stoichiometric mixture is advantageous so that a three-way catalyst can be used to clean up the exhaust [2]. In practice, the fuel injection rate m_{fi} can not be obtained in this way because of fuel deposition and transportation mechanisms as shown in Equations (2.10), (2.11), and (2.12). However, during the past decade, methods based on artificial neural network have established effective tools of system modelling, which give us a black-box approach to deal with process mechanisms [64] [65].

In this chapter, a RBFNN model is used to predict the air mass flow into intake port $\dot{m}_{ap}$, then, the expected engine port fuel mass flow $\hat{\dot{m}}_f$ in next sample time can be deduced from Equation (5.1). After that, this expected $\hat{\dot{m}}_f$ is transferred to a RBF-based inverse model of fuel injection dynamics to generate control variable $\dot{m}_{fi}$. Finally, a feedback trim - PI control is used in conjunction with this RBFNN feedforward control to compensate for modelling errors and unmeasured disturbance.

5.2 Control System Configuration

The strategy of RBF-based feedforward-feedback control of AFR is shown in Figure 5.1.

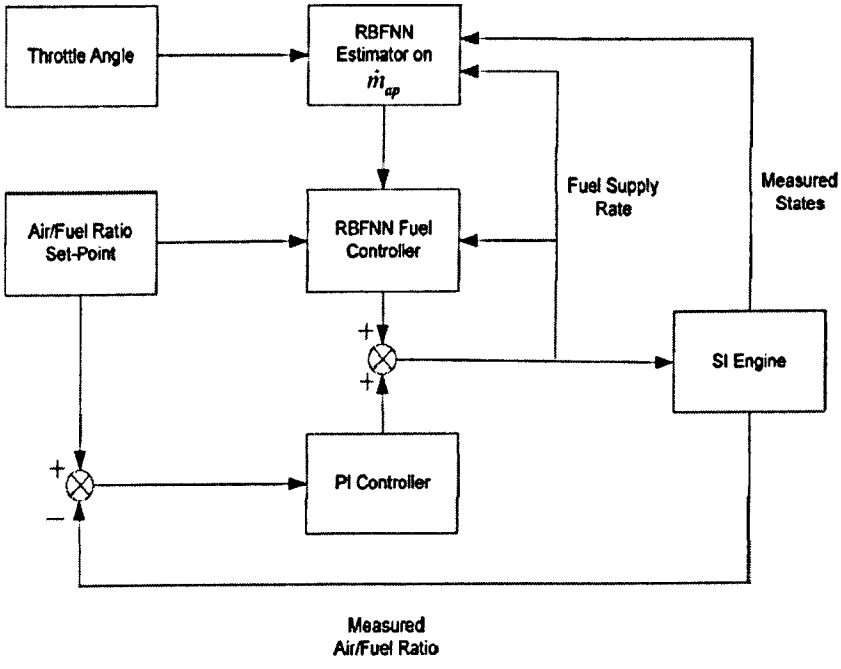

Figure 5.1: RBF-based Feedforward-Feedback Control Structure

External disturbance is usually given by the vehicle driver as the change of throttle angle. It can not be predicted. However, it is easy to be measured by throttle position sensor. In this system structure, the air mass flow rate into the cylinders is estimated by a RBFNN from the throttle angle and some system states including manifold pressure and crankshaft speed. Then, the neural network controller will determine the appropriate fuel flow rate by Equation (5.1) according to the estimated air flow rate. This feed-forward controller produces compensation for the external disturbance. At the same time, the PI controller is used to form the feedback control and to control the uncompensated effects as well as the steady state offset. In the experiment, PI controller is tuned by hand. A flowchart of the control actions in time sequence is given in Figure 5.2. The improved system's transient response is displayed and commented on the section of implementation.

5.3 Neural Modelling of $\dot{m}_{ap}$ and $\dot{m}_{fi}$

5.3.1 Neural Model for $\dot{m}_{ap}$

According to the analysis of $\dot{m}_{ap}$ in Equation (2.4), at sample instant k, the relevant inputs affecting $\dot{m}_{ap}$ can be determined as intake manifold temperature T_i, throttle position u, crankshaft speed n, and $\dot{m}_{ap}$ itself in the last sample time, all inputs are measurable. Different orders and time delays of these variables are tried in the model training, and the following order are found most appropriate giving the minimum modelling error. We have the following equation:

$$\dot{m}_{ap}(k) = g(T_i(k-1), u(k-1), n(k-1), \dot{m}_{ap}(k-1)) + e(k) \qquad (5.2)$$

where $g(\cdot)$ is a nonlinear function and $e(k)$ is a noise with zero mean. Equation (5.2) completely describes the relevant inputs and their time dependencies on

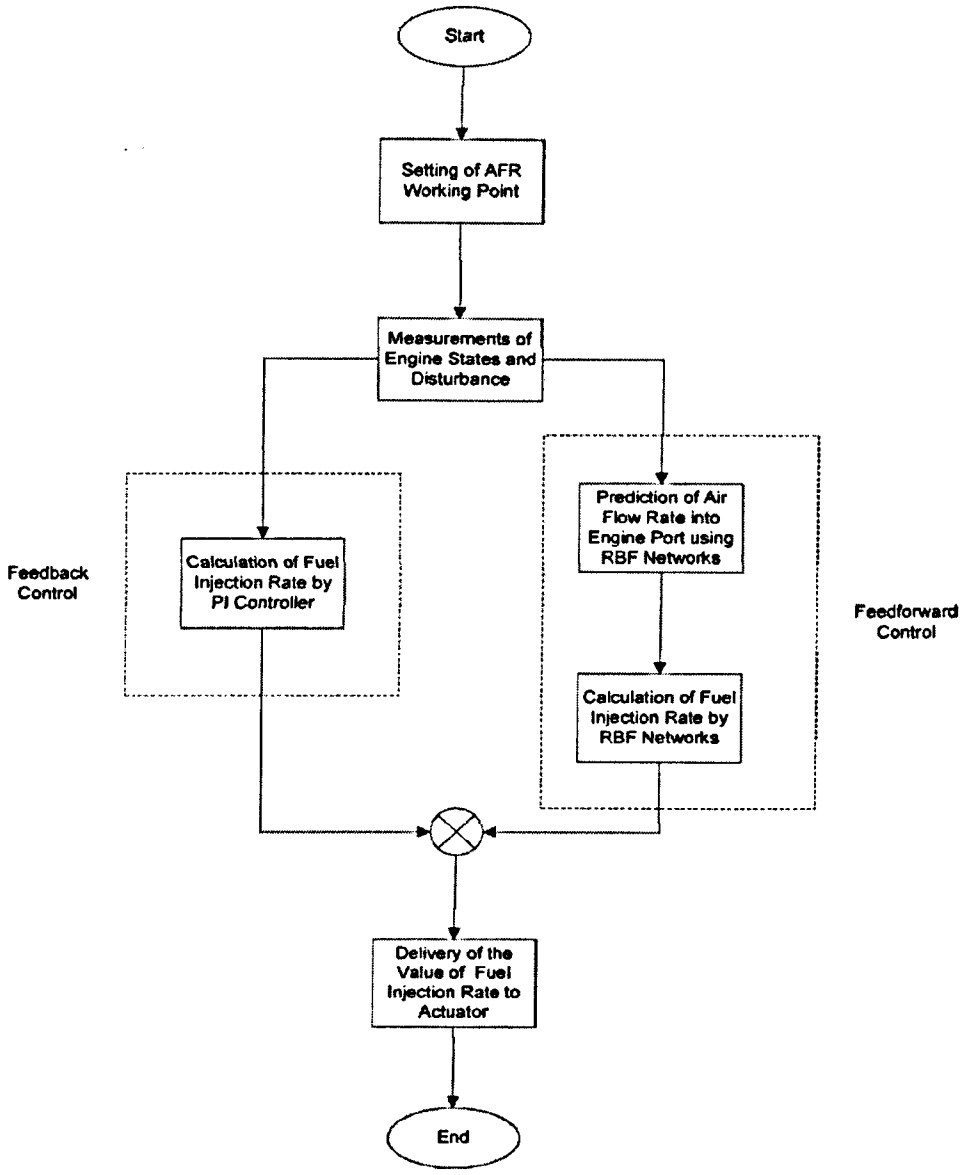

Figure 5.2: RBF-based Feedforward-Feedback Control Flowchart

$\dot{m}_{ap}$. Therefore, the RBF neural network of $\dot{m}_{ap}$ modelling is constructed as Figure 5.3. To train this network model, three sets of RAS were designed for T_i, u and n. The ranges of these excitation signals are listed in Table 5.1. The sample time of simulation system was set to 0.1 s. The simulation was run for 500 seconds with a set of 5000 data samples of $\dot{m}_{ap}$ collected. These data were divided into two groups: the first 4500 data samples were used for training while

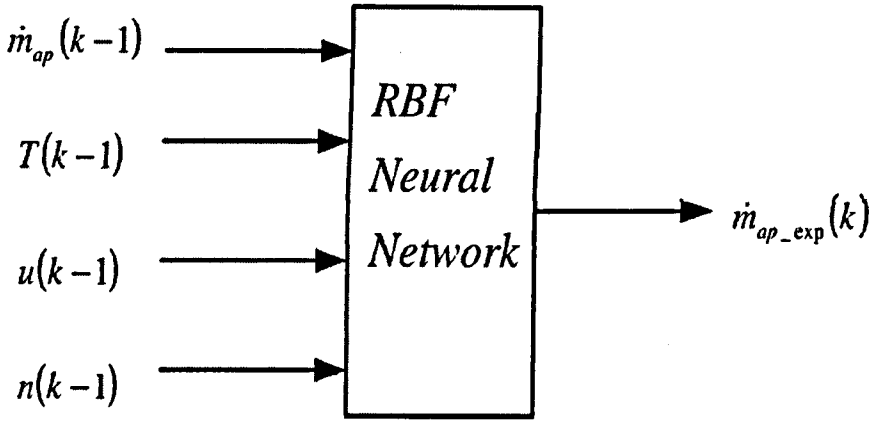

Figure 5.3: The Structure of RBFNN for $\dot{m}_{ap}$

RAS	Min	Max
u	20°	70°
T_i	269 K	311 K
n	1.0×10^3 rpm	5.5×10^3 rpm

Table 5.1: Ranges of Excitation Signals for $\dot{m}_{ap}$ Modelling

the remaining 500 data samples for validation. Before training and test, the raw data were scaled using the following equation

$$x_{scale}(k) = \frac{x(k) - \min\{x(i)\}}{\max\{x(i)\} - \min\{x(i)\}} \quad , \qquad i \in [1, N] \tag{5.3}$$

where N is the number of samples in the data set. In the training, the number of hidden layer nodes were chosen by experiment. Different numbers of nodes from 5 to 15 were tried respectively and the generated MAE of modelling for the 500 test samples were compared. It was found that the neural network with 13 centers gave minimum prediction MAE of 0.0546. Therefore, 13 nodes were used in RBF model for $\dot{m}_{ap}$. The result is shown in Figure 5.4

5.3.2 Inverse Model for $\dot{m}_{fi}$

In this section a RBF network is trained to model the inverse dynamics of fuel injection to generate the control variable $\dot{m}_{fi}$. When $\dot{m}_{ap}$ is predicted by the

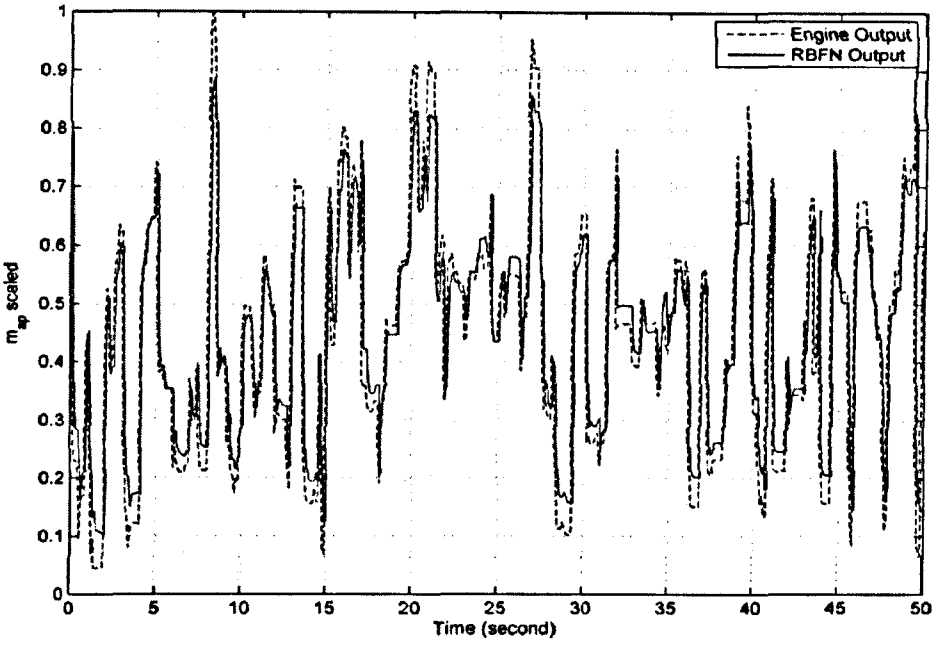

Figure 5.4: $\dot{m}_{ap}$ Modelling with RBFNN

neural model in Figure 5.3, it will be used to calculate the desired fuel flow rate into engine port, $\bar{\dot{m}}_f$, by keeping $\lambda = \dot{m}_{ap}/\bar{\dot{m}}_f = 14.7$. However, such obtained desired $\bar{\dot{m}}_f$ can not be directly used in fuel injection control, as the injected fuel flow rate is not $\dot{m}_f$, but $\dot{m}_{fi}$. There are dynamics between $\dot{m}_{fi}$ and $\dot{m}_f$ as described by Equation (2.10), (2.11), and (2.12). Considering these equations, the dynamics can be represented by

$$\dot{m}_f(t) = f_1[\dot{m}_{fi}(t), n(t)] \tag{5.4}$$

With different orders and time delays tried in model training, the most appropriate order for the discrete equation is as follows

$$\dot{m}_f(k) = f_2[\dot{m}_f(k-1), \dot{m}_f(k-2), \dot{m}_{fi}(k-1), \dot{m}_{fi}(k-2), n(k-1)] \tag{5.5}$$

Therefore, $\dot{m}_{fi}(k-1)$ can be predicted using the above dynamics by

$$\dot{m}_{fi}(k-1) = f_3[\dot{m}_{fi}(k-2), \dot{m}_f(k), \dot{m}_f(k-1), \dot{m}_f(k-2), n(k-1)] \tag{5.6}$$

where f_3 is an inverse nonlinear function of (5.5). If $\dot{m}_{fi}$ is predicted at current sample instant k, the equation is

$$\dot{m}_{fi}(k) = f_3[\dot{m}_{fi}(k-1), \dot{m}_f(k+1), \dot{m}_f(k), \dot{m}_f(k-1), n(k)] \tag{5.7}$$

Since $\dot{m}_f(k)$ and $\dot{m}_f(k+1)$ are not available when $\dot{m}_{fi}(k)$ needs to be predicted, the desired value of fuel flow rate into engine port, $\bar{\dot{m}}_f(k)$, calculated from $\bar{\dot{m}}_f(k) = \hat{\dot{m}}_{ap}(k)/\lambda$, is used to replace $\dot{m}_f(k)$ and $\dot{m}_f(k+1)$. $n(k)$ is approximated by $n(k-1)$. Thus, the above equation becomes

$$\dot{m}_{fi}(k) = f_3[\dot{m}_{fi}(k-1), \bar{\dot{m}}_f(k), \dot{m}_f(k-1), n(k-1)] \tag{5.8}$$

This equation can be approximated by a RBF network in Figure 5.5. The ranges

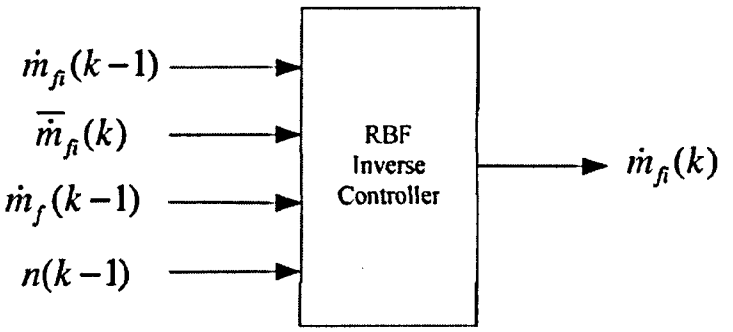

Figure 5.5: The Structure of RBFNN for $\dot{m}_{fi}$

of these excitation signals $\dot{m}_f$, $\dot{m}_{fi}$, and n, are listed in Table 5.2 according the simulation engine model.

RAS	Min	Max
$\dot{m}_f$	1.2×10^{-3} kg/s	2.2×10^{-3} kg/s
$\dot{m}_{fi}$	4.5×10^{-4} kg/s	3.0×10^{-3} kg/s
n	1.0×10^{3} rpm	5.5×10^{3} rpm

Table 5.2: Ranges of Excitation Signals for $\dot{m}_{fi}$ Modelling

Simulation setting is the same as before, a set of 5000 data samples of $\dot{m}_{ap}$ was collected, with the first 4500 data samples for training and the remaining 500 data samples for validation. The number of hidden nodes was selected to be 12 in the experiments following the procedure as described for $\dot{m}_{ap}$. The result is shown in Figure 5.6. The prediction MAE for $\dot{m}_{fi}$ in scaled data is only 0.0012,

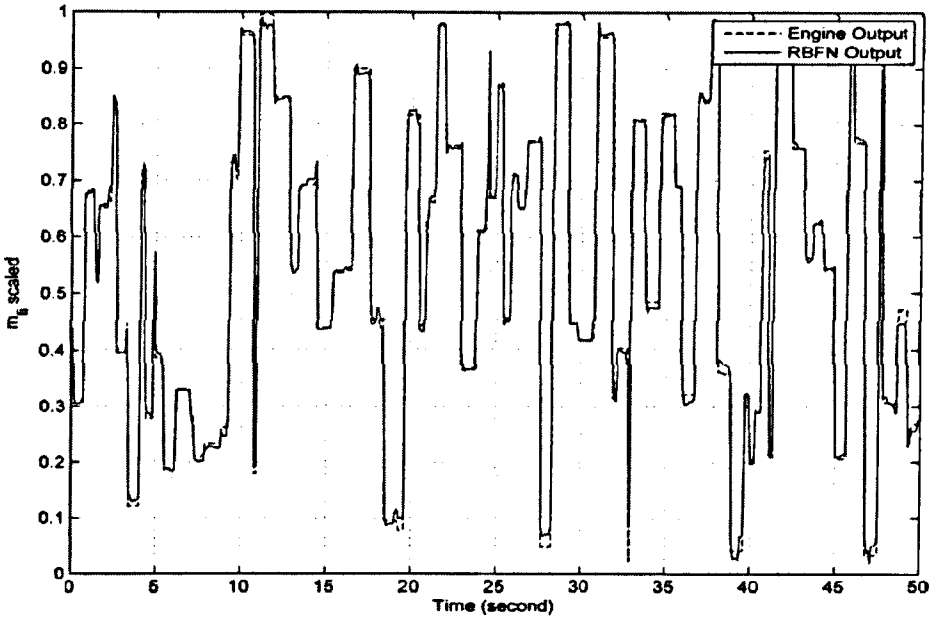

Figure 5.6: $\dot{m}_{fi}$ Modelling with RBFNN

the output of the neural network almost matches the real engine output data perfectly. This is because of $\dot{m}_f(k-1)$, which can be calculated online at sample time k, was used to predict the value of fuel injection rate $\dot{m}_{fi}(k)$. Another reason for constructing the inverse model in this way is that we have created a RBFNN model that can predict the $\dot{m}_{ap}$ for one step ahead. Therefore, $\bar{\dot{m}}_f(k)$ can be calculated through $\hat{\dot{m}}_{ap}(k)$ according to Equation (5.1), then, is used as one of the inputs of inverse model, which is approximated to $\dot{m}_f(k)$.

5.4 PID Control on AFR

We have already seen in last section that the throttle angle u is the signal given by vehicle driver, which can not be predicted and manipulated by control system. Therefore, it is classified as disturbance. $\dot{m}_{fi}$ is the manipulated variable used to control the process. Based on the analysis of Equation (2.10), (2.11), (2.12), and (5.1), we can see that the difficulty to design a PID controller is that the output AFR decreases with $\dot{m}_{fi}$ increasing or the steady state gain is negative, which makes it impossible to tune PID controller by common methods. To solve the problem, two PID control methods are implemented and the configures are shown in Figure 5.7 and 5.8 respectively.

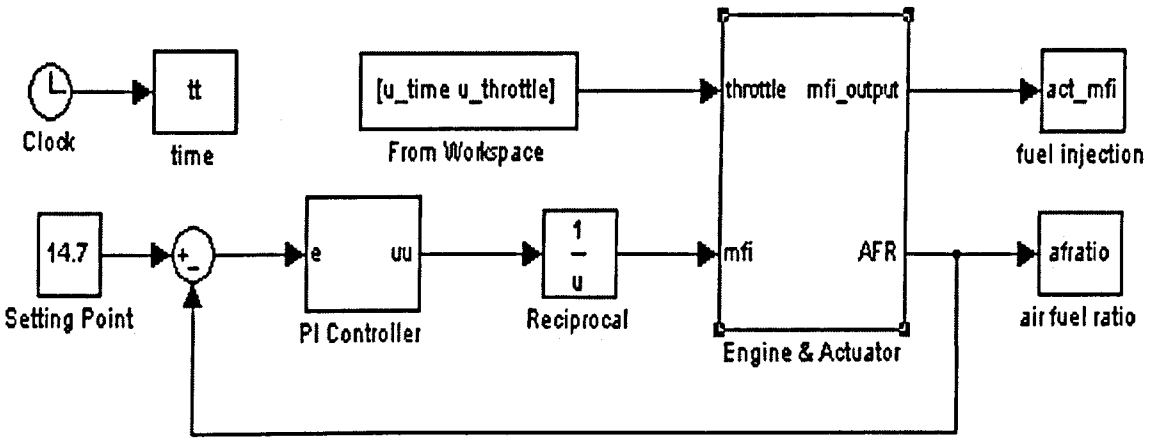

Figure 5.7: PID Control Scheme-1

The idea in the first method is that the design of PID controller will be easier if the output AFR can change in the same direction as the control variable. In Figure 5.7, the actual control variable used by actuator is the reciprocal of PID controller's output uu. When this component is added to the engine dynamics, the AFR will increase when $\dot{m}_{fi}$ increase. A positive static gain is achieved, and the PID controller can be easily realized by standard tuning method, for example, Ziegler-Nichols method.

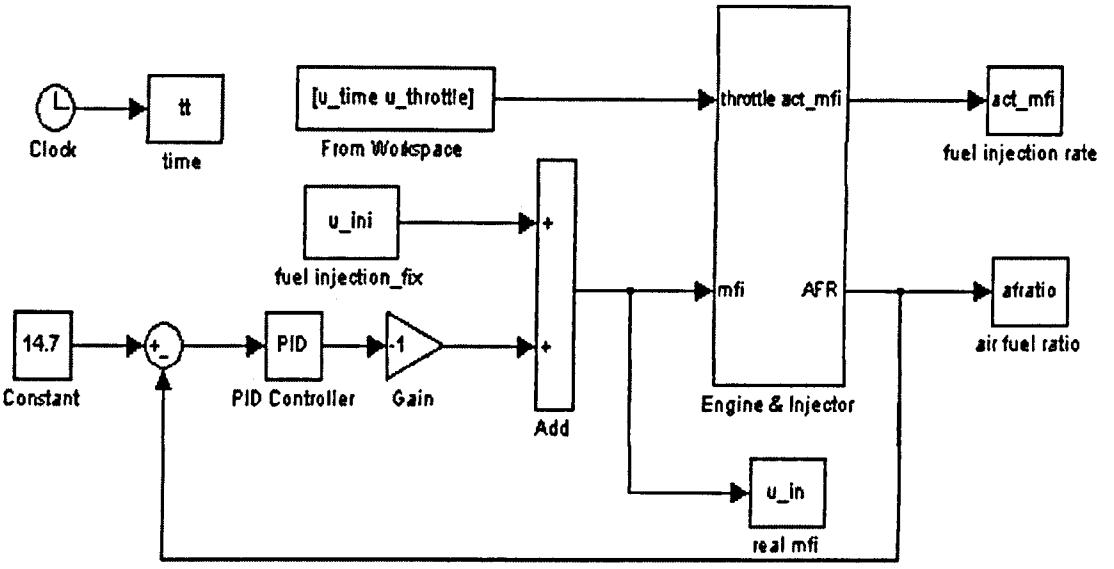

Figure 5.8: PID Control Scheme-2

Inspired by look-up tables method, the optimal fuel injection rate can be found by experiments if given a specific value of throttle angle. Then, the PID controller working at this operating point, as seen in Figure 5.8, is to eliminate the negative effect of throttle angle change. The simulation results show that the better performance can be obtained by this method if compared with the first scheme.

5.5 Feedforward-Feedback Control System Simulation

5.5.1 The Control Structure

The RBF-based feedforward-feedback control system structure in our implementation is shown in Figure 5.9. At the sample time k, the neural model for $\dot{m}_{ap}$ makes one-step prediction on the air flow rate into engine port $\dot{m}_{ap}(k)$ using the

measured states including throttle angle position $u(k-1)$, intake manifold temperature $T_i(k-1)$, and $\dot{m}_{ap}(k-1)$. The predicted air flow rate $\hat{\dot{m}}_{ap}(k)$ divided by 14.7 (stoichiometric value) is the fuel film rate $\bar{\dot{m}}_f(k)$ for an ideal combustion. However, $\bar{\dot{m}}_f(k)$ can not be used by fuel injection directly. In our control scheme, $\bar{\dot{m}}_f(k)$, measured engine speed $n(k-1)$, and fuel injection rate $\dot{m}_{fi}(k-1)$ form a input vector for the inverse model of fuel injection dynamics, which outputs the appropriate fuel injection rate for this sample time. On the other hand, to enhance the performance in steady state, the a PI controller is used to form the feedback control. Therefore, the injection rate for activating fuel injector is the sum of two control variables, one is from RBF based NN controller; the other from PI controller.

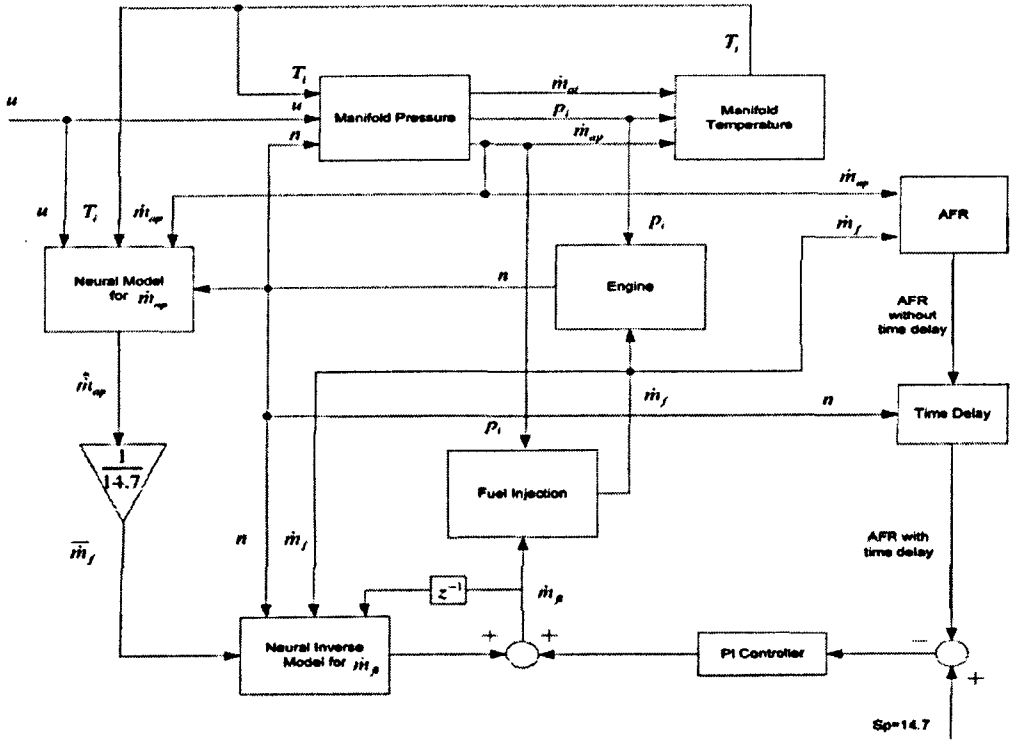

Figure 5.9: RBF-based Feedforward-Feedback Control Structure on AFR

5.5.2 Throttle Angle Change During Control

The change of throttle angle in driving condition is given by the vehicle driver. Usually the throttle plate in the closed position is say 9° to 12° degrees. Using this convention, the maximum opening angle of the throttle, if it is to be effective, is about 70° to 80°. Opening it wider than this will have no influence on the throttle air mass flow. In our simulations, as shown in Figure 5.10, a throttle angle changing gradually from 25° to 65° by 10° each 20 seconds with 0.5° random measurement error is chosen to simulate the driving dynamics of the simulation engine. This almost covers the whole engine operating condition. The AFR is to be controlled between the $\pm 1\,\%$ bounds of the stoichiometric value(14.7), i.e. $99\% \times 14.7 \leq \text{AFR} \leq 101\% \times 14.7$.

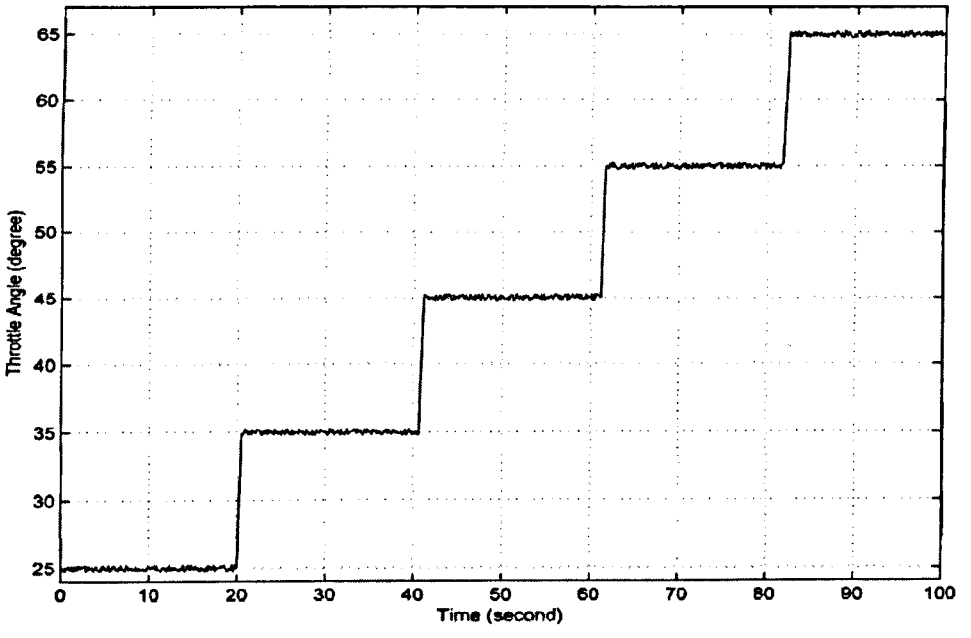

Figure 5.10: Throttle Angle Change During Control

5.5.3 PID Control Parameters Setting

A digital PID controller shown as follows is realized [66].

$$\dot{m}_{fi}(k) = \dot{m}_{fi}(k-1)+K[(1+\frac{T}{T_I}+\frac{T_D}{T})e(k)-(1+\frac{2T_D}{T})e(k-1)+\frac{T_D}{T}e(k-2)] \quad (5.9)$$

Here, the sampling time T is also chosen to be 0.1s. First, the derivative time T_D is set to be 0 and the PI controller was tuned by open-loop method of the Ziegler-Nichols rules [67]. After fine tuning, the PI controller that is use here with RBF based neural network controller for air fuel ratio regulation is

$$\dot{m}_{fi}(k) = \dot{m}_{fi}(k-1) + 2 \times 10^{-6} \cdot e(k) - 1.4 \times 10^{-4} \cdot e(k-1) \quad (5.10)$$

The derivative term T_D can cause large amounts of change in the output given small amounts of measurement or process noise. Further experiments also illustrated that T_D can not make an contribution on the improvement of control performance. In the simulation, the sampling time is chosen to be 0.1 s same as that in the data acquisition.

5.5.4 Simulation Results

From the response of AFR in simulation, it can be seen that the engine process exhibits a higher degree of nonlinearity when the throttle angle operates between 25° and 45°. It means the accurate AFR control in this range is more challenging. The control result and the injected fuel mass are shown in Figure 5.11 and 5.12. It can be seen that the AFR is controlled within the $\pm 1\,\%$ bounds of the stoichiometric value(14.7) in both transient and steady state. The good control performance is because of the good prediction of the RBR based feedforward controller on the air mass flow rate into engine port according the states of manifold dynamics, the fuel injection rate it generates consequently can make appropriate compensation for the influence of the throttle angle. The tracking performance is

evaluated by mean absolute error that has been defined in Equation (4.43). The tracking MAE of AFR control is 0.0014 in this simulation.

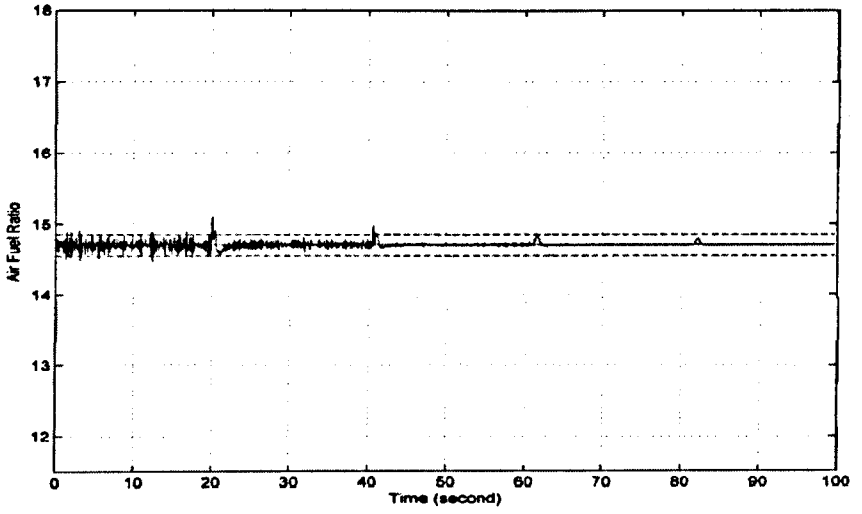

Figure 5.11: Simulation Result of RBF-based Feedforward-Feedback Control on AFR

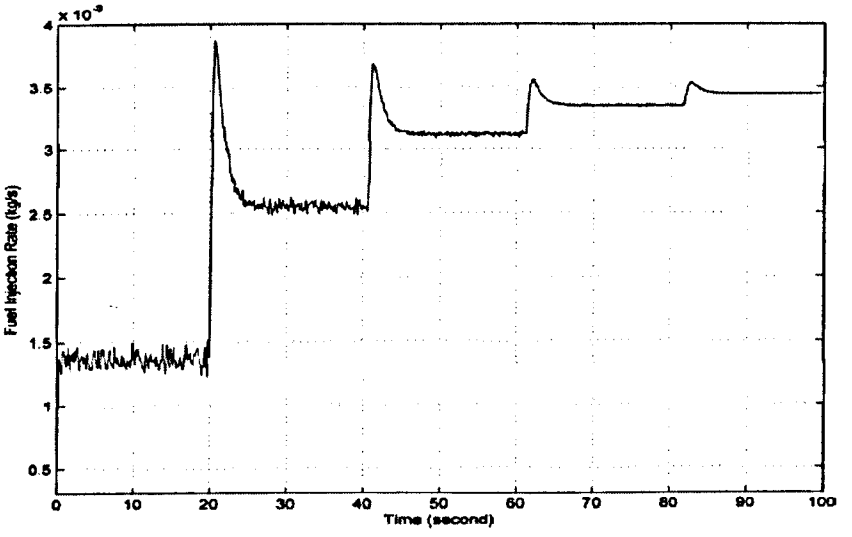

Figure 5.12: $\dot{m}_{fi}$ Produced by RBF-based Feedforward-Feedback Controller

5.6 Control Performance Comparison with PID

In order to show the advantages of RBF-based feedforward-feedback controller, its control result is compared with that of a traditional PID controller, which is used in the production ECU of current automotive industry. In this simulation, the structure of PID control used here is the scheme 2 shown in section 5.4. Because of the severe nonlinearity in the engine dynamics when the throttle angle operates between between 25° and 45°, the PID controller is developed to work at 40° of throttle angle, and initial value for fuel injection is set to 2.895×10^{-3}. By only PID control on AFR, this design is reasonable to guarantee the overall performance in the whole engine operating condition. According to Equation (5.9), the PID controller for air fuel ratio regulation is

$$\dot{m}_{fi}(k) = \dot{m}_{fi}(k-1) + 36.12 \cdot e(k) - 36 \cdot e(k-1) \tag{5.11}$$

The tracking MAE is 0.2499. Figures 5.13 and 5.14 show the control result and the corresponding fuel injection rate, with the tuned PI controller and the same throttle angel changing.

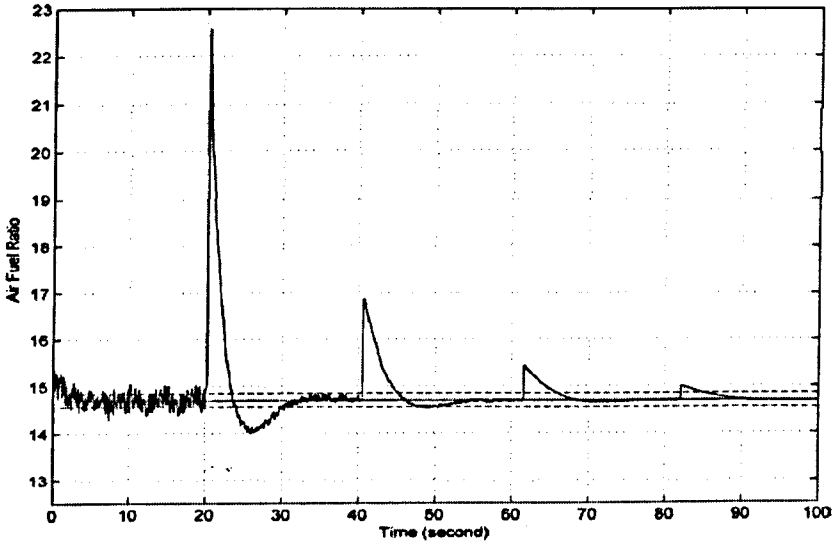

Figure 5.13: Simulation Result of PI Control on AFR

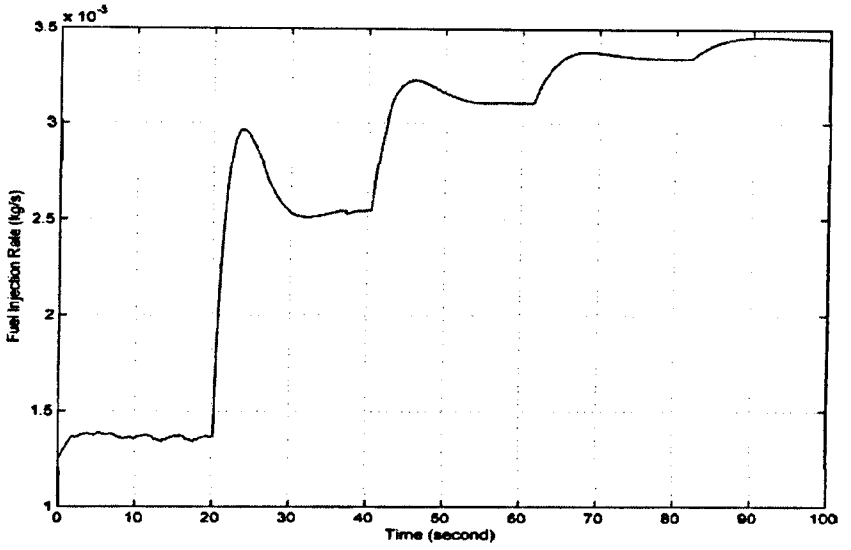

Figure 5.14: $\dot{m}_{fi}$ Produced by PI Controller

Compared with traditional feedback controller, it can be seen that RBF-based feedforward-feedback control significantly reduces the overshoot and settling time. In addition to better transient response, the tracking MAE of RBF-based feedforward-feedback controller is much less than that of PI controller. This means the steady state response is further improved as well. Therefore, a significant improvement in performance is achieved using RBF-based feedforward-feedback controller.

5.7 Robustness Analysis

Air Leakage in the intake manifold is one of the exasperating problems that are influenced by aging effect, which can result from any holes in the manifold or at any of the connection points. Air leakage in intake manifold can ruin engine performance, boost exhaust temperature and raise emissions. In the engine model we are using, the parameter $\dot{m}_{ap}$ stands for the air flow rate into the engine port, which is shown in Equation (2.4). When air leakage happens, the reduced amount of air into the cylinders will result in the decrease of $\dot{m}_{ap}$, this will fool the ECU into over-fuelling, and incomplete combustion will result in high emissions of HC in the exhaust gases and a decreasing effective work. In this section, the

robustness of developed neural network-based control system will be tested to deal with air leakage problem.

5.7.1 Control Performance with 10% Uncertainty

In this simulation, all the parameters settings for engine model are the same as those in previous experiments. The sample time is 0.1 second and throttle angle changes from 25° to 65° in 100 seconds. A model mismatches is introduced on

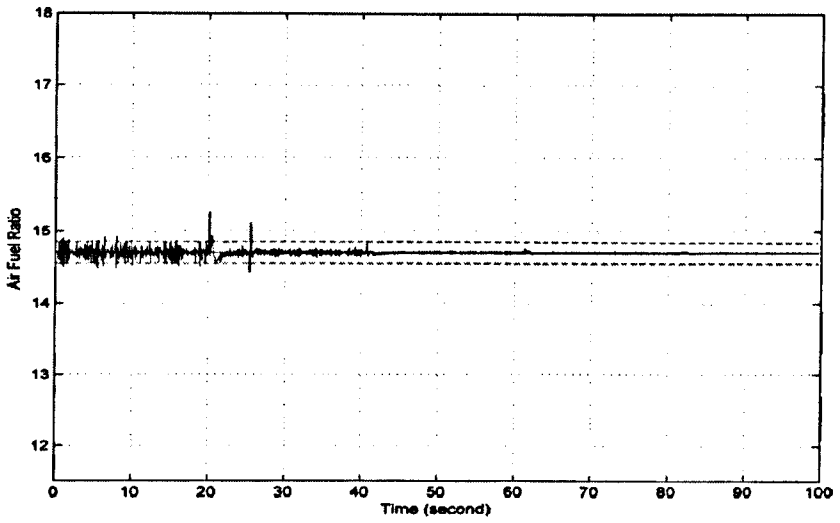

Figure 5.15: Simulation Result of RBF-based Feedforward-Feedback Control on AFR When Engine Model Has 10% Mismatch

$\dot{m}_{ap}$ - the air flow rate into engine port, which is a 10% magnitude decrease of the original $\dot{m}_{ap}$ value after running the simulation for 25 seconds. According to Equation (2.4), if there is an air leakage during engine running, the air fuel ratio suddenly decreased. The look-up table has little capacity to deal with this situation due to the static control map. However, the RBF-based feedforward and feedback controller developed in this project can adjusts the fuel injection rate according the real-time condition of intake manifold, and compensate for the influence of this system mismatch after several sample times and, which can be seen from the comparison on fuel injection rates between Figure 5.12 and Figure 5.16, therefore keep the satisfactory air fuel control performance. The simulation

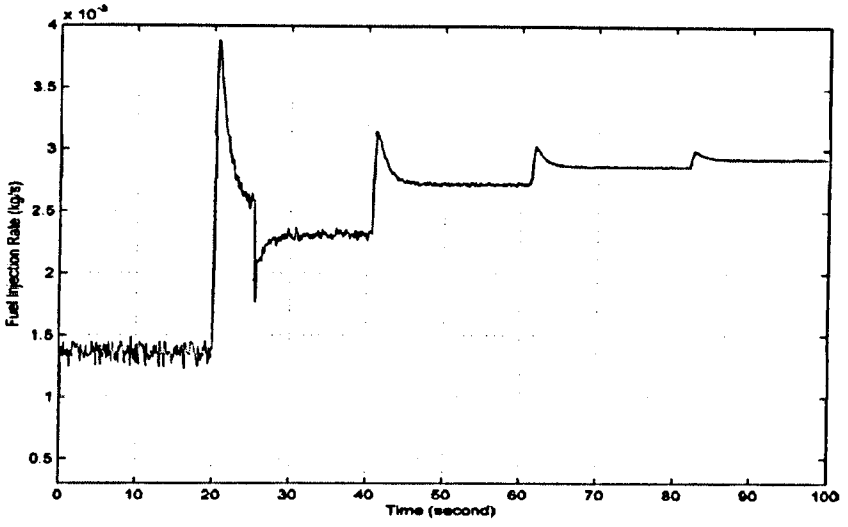

Figure 5.16: $\dot{m}_{fi}$ Produced by RBF-based Feedforward-Feedback Controller When Engine Model Has 10% Mismatch

results are shown in Figure 5.15 and 5.16 and the tracking MAE is 0.0021.

5.7.2 Control Performance with 30% Uncertainty

With the same simulation conditions, more serious air leakage in intake manifold is given to test the method developed. The model mismatch has been increased to 30% this time. From the simulation results shown in Figure 5.15 and 5.16, it can be seen that, the overshoot in this situation has reached 10% due to the increased level of air leakage, the AFR has the oscillation around the stoichiometric value(14.7) for several sampling intervals. But still, a good control performance has been achieved overall. The controller output - m_{fi} still stay in its working range. The tracking MAE is 0.0039. The simulation results show the developed control method is robust to 30% system mismatch.

5.8 Summary

This chapter presents a new development in automotive engine AFR control with a feedforward-feedback control system structure. Two neural networks are used

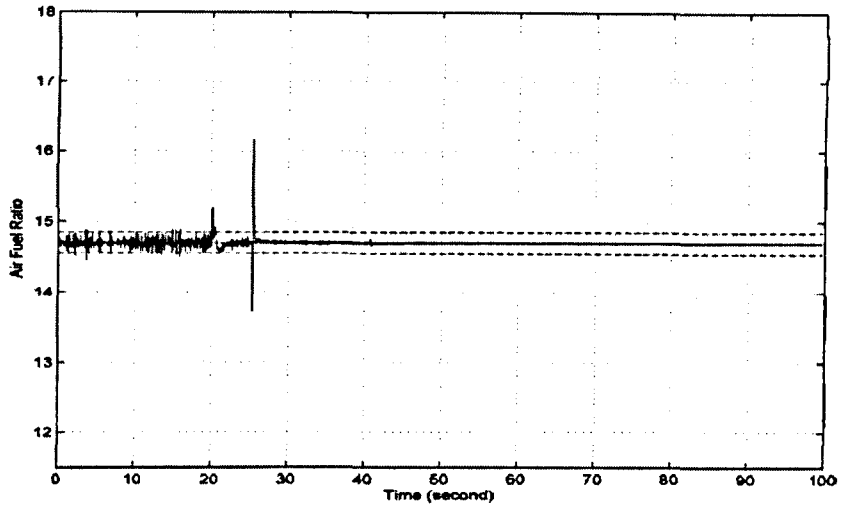

Figure 5.17: Simulation Result of RBF-based Feedforward-Feedback Control on AFR When Engine Model Has 30% Mismatch

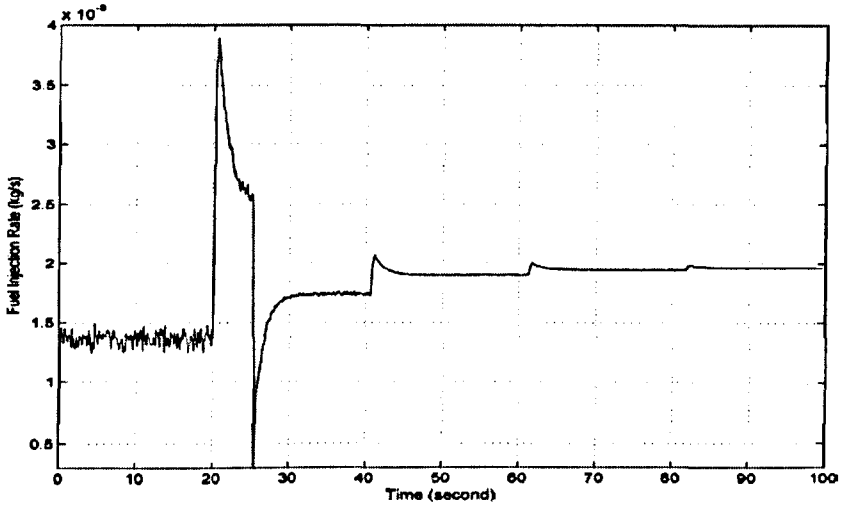

Figure 5.18: $\dot{m}_{fi}$ Produced by RBF-based Feedforward-Feedback Controller When Engine Model Has 30% Mismatch

to estimate air low rate and required fuel injection rate under variations of the throttle angle. The inverse of fuel injection dynamics is employed to obtain a more accurate fuel injection rate. The developed system is evaluated with an engine benchmark of simulation throughout the whole operating space, and satisfactory performance is obtained. Comparing with a PI control the developed system shows a significant improvement in both overshoot reduction and quick

response. Moreover, the system is capable of dealing with the air leakage problem. Therefore, the RBF-based feedforward-feedback control is one of the potential control schemes for the production ECU of the next generation, which can replace the lookup table method and traditional feedback control.

Chapter 6

RBFNN-based Model Predictive Control

6.1 Introduction

Currently the most controllers in ECU are formulated by using PI or PID controllers with gain scheduling along with map based feeforward control functions. Around 9000 parameters are more than 600 control and diagnostic functions are implemented in modern electronic control units [8]. This makes the system highly complex. Adjustments of these controller are mostly done manually by experts after long research and experiments.

Model predictive control is widely adopted in chemical industry, which has been proved an effective control principle to handle the complexities and improve the quality of the control system. For the MPC on automotive engines, previous researches have shown that linear models based MPC are not effective for whole engine working range and can not produce good control during fast throttle angle transients [68]. Recently neural network based models as one class of nonlinear model have attract the attention of researchers for the nonlinear model predictive control (NMPC) on engine application [12] [13]. In this project, the neural net-

work based NMPC strategy to solve AFR regulation problem of SI engines will be investigated.

Radial basis function neural network is chosen among the feedforward neural networks not only because it has the attractive approximating power but also because it can achieve online updating efficiently, which is crucially important to deal with the system time-varying effects.

6.2 Control System Configuration

The idea of model predictive control with neural network has been introduced in details by some researchers, such as Draeger [69]. The strategy is shown in Figure 6.1. The obtained adaptive RBF neural network is used to predict the

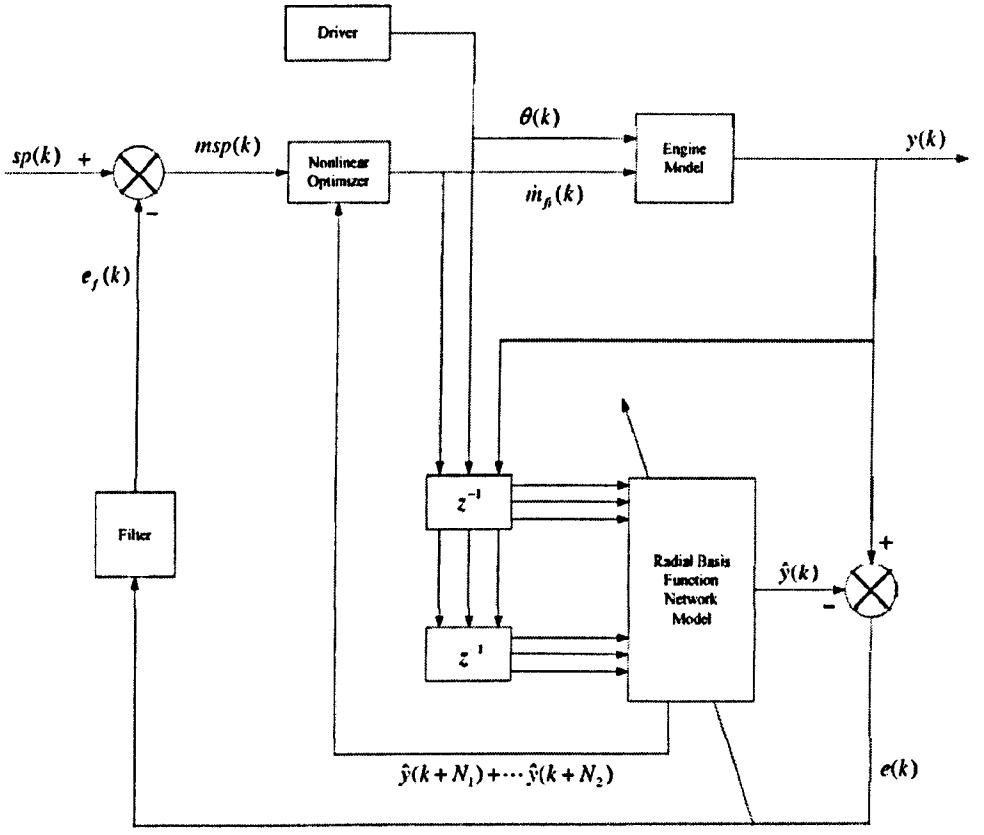

Figure 6.1: Control Structure of RBF-based MPC

engine output for N_2 steps ahead.The non-linear optimizer minimizes the errors between the set-point and the engine output by using the cost function,

$$J(k) = \sum_{i=k+N_1}^{k+N_2} [msp(i) - \hat{y}(i)]^2 + \xi \sum_{i=k}^{k+N_u} [\dot{m}_{fi}(i) - \dot{m}_{fi}(i-1)]^2 \qquad (6.1)$$

Here, N_1 and N_2 define the prediction horizon. ξ is a control weighting factor which penalizes excessive movement of the control input,the fuel injection $\dot{m}_{fi}(k)$. N_u is the control horizon. Then the remaining main problem of MPC is to solve the non-linear optimization problem in each sample period, calculate a series of optimal $\dot{m}_{fi}(k)$,$\dot{m}_{fi}(k+1)$,$\dot{m}_{fi}(k+2)$,$\cdots$,$\dot{m}_{fi}(k+N_u-1)$, from which the neural network model generates outputs to minimis $J(k)$ in Equation (6.1). Finally the first control variable in the series $\dot{m}_{fi}(k)$ is used to control the process and this procedure is repeated in the next sample period.

6.3 Engine modelling using RBFNN

6.3.1 Data Collection

In engine data collection, the training data must be representative of typical plant behavior in order to analyze the performance of different adaptive engine models in practical driving conditions. This means that input and output signals should adequately cover the region in which the system is going to be controlled [69]. As shown in Figure 2.16, the engine model used for this research has two inputs - fuel injection rate $\dot{m}_{fi}$ and throttle angle u; one output - air fuel ratio AFR. To obtain the engine data for neural network training, a set of RAS was designed for throttle angle and fuel injection. As shown in Table 6.1, throttle angle was bounded between 20 degree and 70 degree and the fuel injection rate range used in simulation engine is from 0.0007 to 0.0079 kg/s, the sample time is set to be 0.1 s. The first 1000 samples of excitation signals for engine model inputs are shown in Figure 6.2 and 6.3. After introducing the excitation signals to the

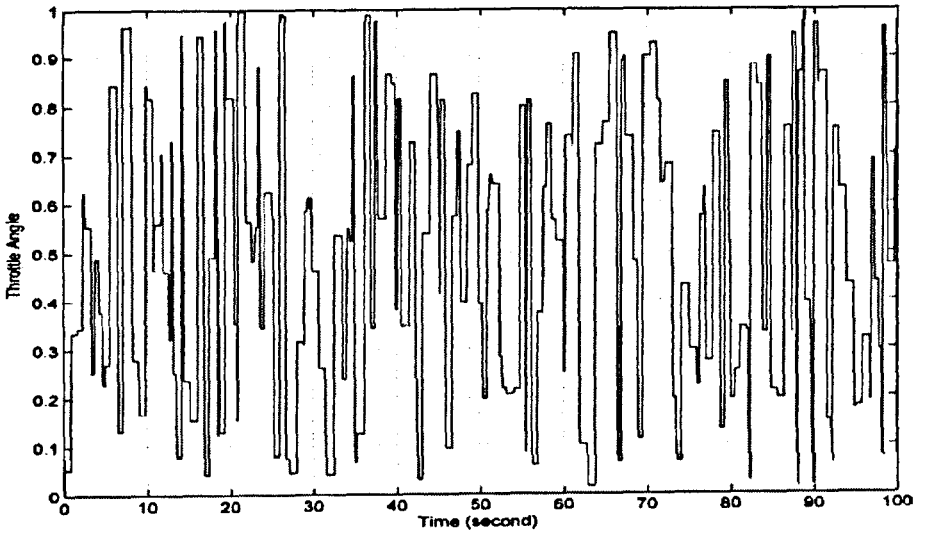

Figure 6.2: Scaled Throttle Angle Data for RBFNN Training

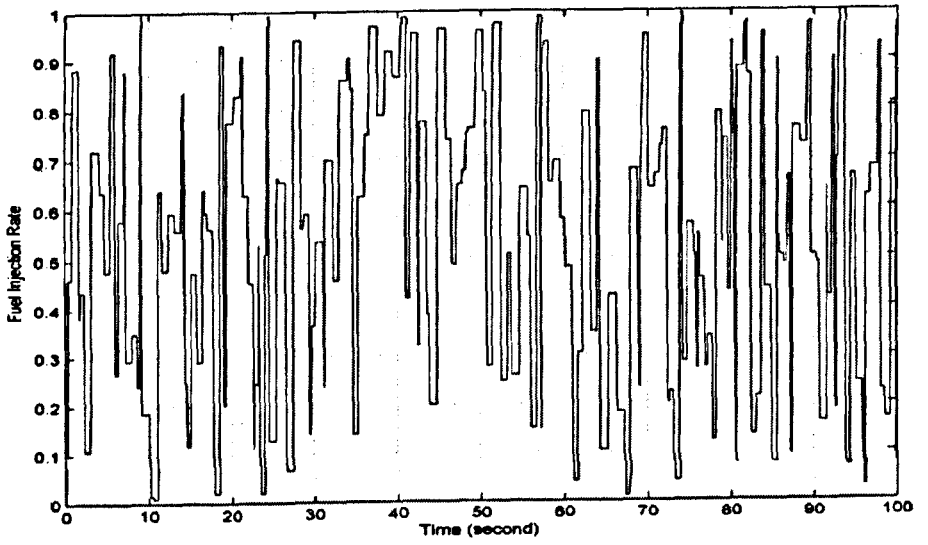

Figure 6.3: Scaled Fuel Injection Rate Data for RBFNN Training

RAS	Min	Max
θ	20°	70°
$\dot{m}_{fi}$	7×10^{-4}	7.9×10^{-3}

Table 6.1: Range of Excitation Signals for AFR Modelling using RBFNN

engine model in Figure 2.16, the data can be collected from the output of air fuel ratio, and Figure 6.4 shows the first 1000 samples of the AFR data for training RBFNN. All the data shown in the figures here are scaled using Equation (5.3). A set of 5000 data samples obtained was divided into two groups. The first 4500 data samples were used for training RBFN model and the rest would remain for model validation.

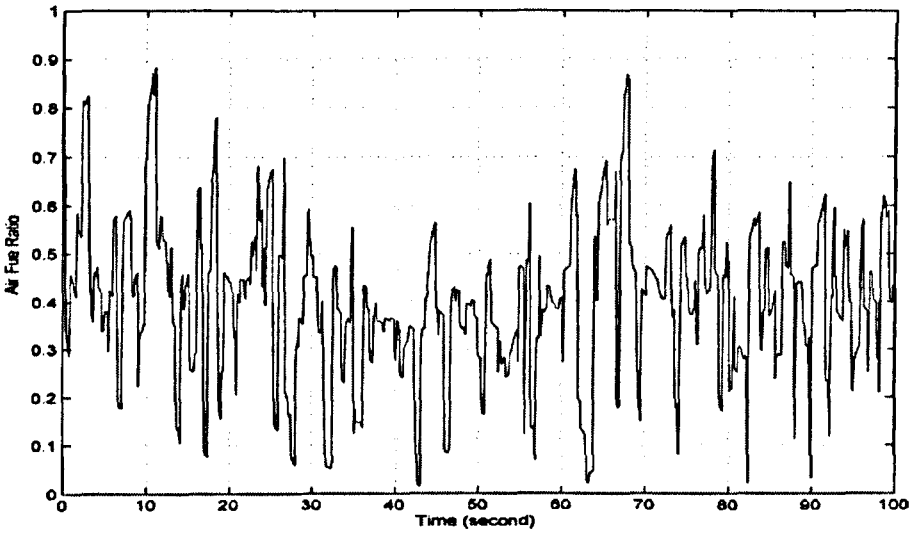

Figure 6.4: Scaled Air Fuel Ratio Data for RBFNN Training

6.3.2 Engine Modelling using RBFNN

Given the expanded engine model as shown in Figure 2.16, the RBFNN engine model has three inputs: fuel injection rate m_{fi}, throttle angle θ and air-fuel ratio y, and one output: air-fuel ratio. Different orders of network inputs and different number of hidden layer nodes have been used in training experiments and the second-order structure with 12 hidden nodes is chosen after experiments as shown in Figure 6.5, which gives the minimum prediction error. The centres

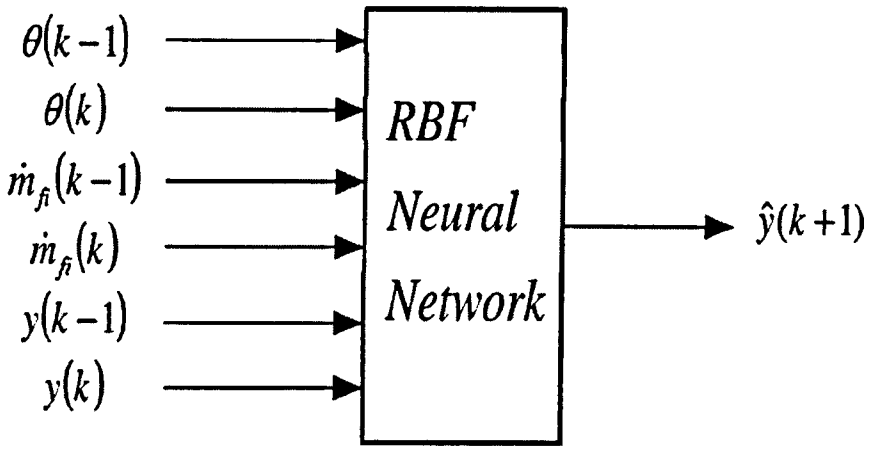

Figure 6.5: RBFNN Model Structure

c and the width in hidden layer nodes of the RBF network σ were determined using K-means algorithm and ρ-nearest neighborhood heuristic respectively and ρ here is set to 3. The recursive least square algorithm is used for training the network weights with its parameters set as $\mu = 1.0$, $w(0) = 1.0 \times 10^{-16} \times U_{n_h \times 1}$ and $P(0) = 1.0 \times 10^{8} \times I_{n_h \times n_h}$, where I is an identity matrix and U stands for a matrix whose components are ones.

All input and output data of the RBFNN have been scaled to the range of $[0, 1]$ before they are used for training and validation. The training data set with 4500 samples are used to train the RBFNN model. Then, the test data set with 500 samples is applied to the trained model and the model predicts results for are displayed in Figure 6.6. From the simulation result in Figure 6.6, it can be seen that the good match between the engine model output and the RBFNN output during the model validation phase. The AFR in the figure is normalized value and the MAE is 0.0365.

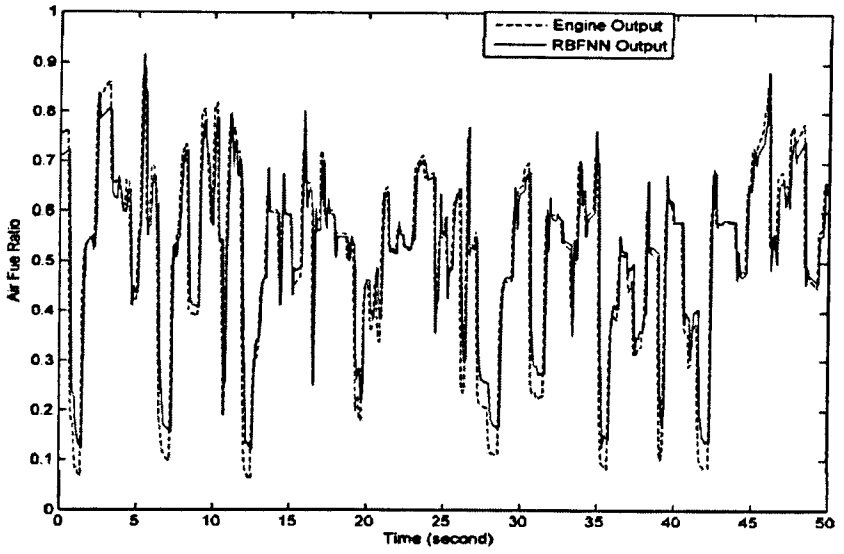

Figure 6.6: Illustration of AFR Validation Data for RBFNN Model

6.3.3 Adaptive Online Training of RBFNN Model

Adaptability is an attractive quality that can be brought by neural controller to deal with the ever-changing engine operating condition. The neural parameters of RBFNN model are updated online using RLS algorithm. $w(0)$ and $P(0)$ have been found in the stage of engine modelling. These obtained parameters are used as initial values for online training. As to the forgetting factor μ, it was set to 1 in the model training, which means all the data are weighted equally. With a smaller value of μ, the algorithm has a shorter memory length and is better adapted to changing dynamics. Therefore, it is necessary to tune μ to make the control algorithm robust against system uncertainties. Simulation results show 0.99 is a suitable value for μ in this application.

6.3.4 Multi-Step Prediction on AFR using RBFNN Model

The feedforward network, such as RBFNN, is often used for one-step prediction. However, to implementing RBFNN based NMPC, the trained RBFNN is required to make multi-step prediction on AFR. The structure of AFR prediction for model predictive control is given in Figure 6.7. Within a sample interval k, $\theta(k-1)$,

$\theta(k)$, $y(k-1)$, $y(k)$, $\dot{m}_{fi}(k-1)$ are constants as they are measured values in last or current sample time. The work left now is to utilize an appropriate optimization algorithm to find the vector $[\dot{m}_{fi}(k)\ \dot{m}_{fi}(k+1)\ \dot{m}_{fi}(k+2)\cdots\dot{m}_{fi}(k+N_u-1)]^T$ that can minimize the objective function $J(k)$ in (6.1).

6.4 Single-dimensional optimization approach

Practical applications often place emphasis on computation speed on the premise that all the performance requirements are met. Therefore, a simple control structure is chosen in this research and assumed that the input $\dot{m}_{fi}$ will remain constant over the prediction horizon: $\dot{m}_{fi}(k)$=$\dot{m}_{fi}(k+1)$= $\cdots$=$\dot{m}_{fi}(k+N_u-1)$. In this case there is only one parameter that are going to be found. The optimization problem to be solved is reduced to one-dimensional. Secant Method is chosen to find the solution of this nonlinear programming (NLP) problem and our experiments show that it is more efficient and reliable in this application than the other interpolation methods.

6.4.1 Problem Formulation

The general nonlinear programming problem could be defined as,

$$\min_{x\in R^n} J(x) \tag{6.2}$$

subject to

$$c_{eq}(x) = 0 \tag{6.3}$$

$$c_{in}(x) \leq 0 \tag{6.4}$$

where $J : R^n \rightarrow R$ is the objective function, $c_{eq} : R^n \rightarrow R^m$ and $c_{in} : R^n \rightarrow R^p$ are constraint functions. All of these functions are smooth. Only inequality constraint applied in our case, as fuel injection rate is bounded within a region.

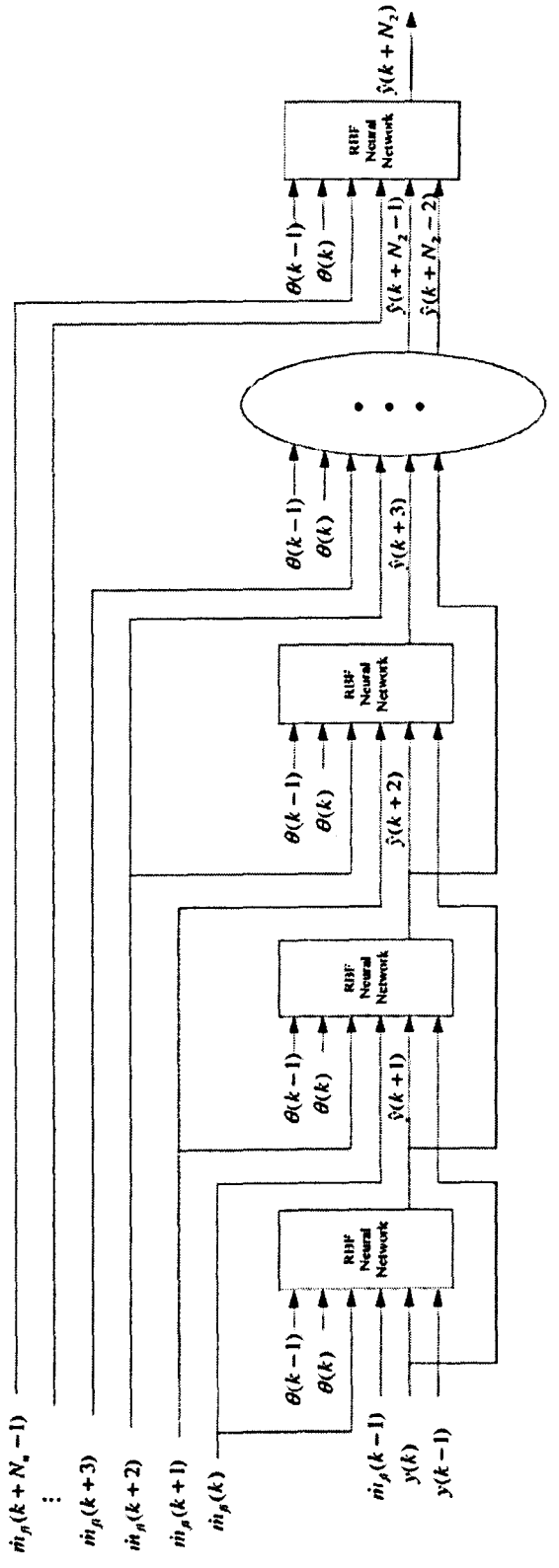

Figure 6.7: Multi-Step Prediction on AFR using RBFNN Model

The Secant Method is to find the improved design vector X_{i+1} from the current design vector X_i using the formula

$$X_{i+1} = X_i + \xi_i^* S_i \tag{6.5}$$

where S_i is the know search direction and ξ_i^* is the optimal step length found by solving the one-dimensional minimization problem as

$$\xi_i^* = \min_{\xi_i}[J(X_i + \xi_i^* S_i)] \tag{6.6}$$

Here the objective function J is to be evaluated at any trial step length t_0 as

$$J(t_0) = J(X_i + t_0 S_i) \tag{6.7}$$

Similarly, the derivative of the function J with respect to ξ corresponding to the trial step length t_0 is to be found as

$$\left.\frac{dJ}{d\xi}\right|_{\xi=t_0} = S_i^T \Delta J|_{\xi=t_0} \tag{6.8}$$

6.4.2 Secant Method

The necessary condition for $J(\xi)$ to have a minimum of ξ^* is that $J'(\xi^*) = 0$. The secant method seeks to find the root of this equation [70]. The equation is given of the form

$$J'(\xi) = J'(\xi_i) + s(\xi - \xi_i) = 0 \tag{6.9}$$

where s is the slope of the line connecting the two points $(A, J'(A))$ and $(B, J'(B))$, where A and B denote two different approximations to the correct solution, ξ^* . The slope s can be expressed as

$$s = \frac{J'(B) - J'(A)}{B - A} \tag{6.10}$$

Equation (6.9) approximates the function $J'(\xi^*)$ between A and B as a linear equation and the solution of Equation (6.9) gives the new approximation to the root of $J'(\xi^*)$ as

$$\xi_{i+1} = \xi_i - \frac{J'(\xi_i)}{s} = A - \frac{J'(A)(B-A)}{J'(B) - J'(A)} \tag{6.11}$$

The iteration process given in Equation (6.11) is known as the secant method, see Figure 6.8.

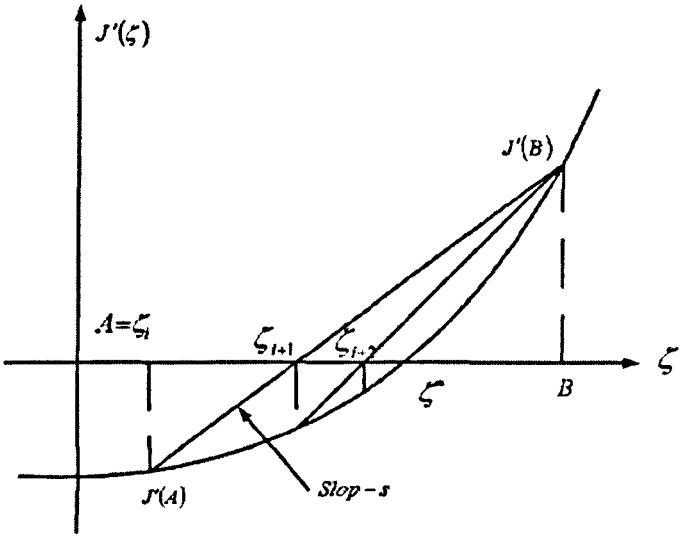

Figure 6.8: Iterative Process of Secant Method

6.4.3 Simulation Results

In the simulation, the set-point of the system is set to be the constant stoichiometric value 14.7. The throttle angle, as disturbance is set with a change from 25° to 65° and with 0.5° uncertainty as shown in Figure 5.10 in last chapter. The AFR is to be controlled between the bounds of the stoichiometric value(14.7). Choosing the sampling time to be 0.1s. The parameters of nonlinear optimization were chosen as $N_1 = 1$, $N_2 = 6$, $\xi = 0.1$, $N_u = 0$, then the MPC of SI engines can be considered as a sub-problem of NLP problems.

$$\min_{x \in R^n} f(\dot{m}_{fi}) \tag{6.12}$$

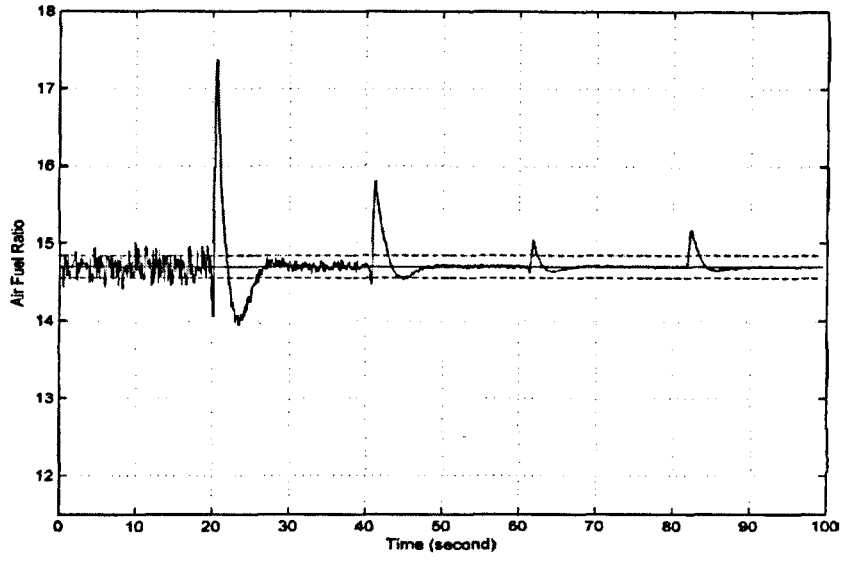

Figure 6.9: Simulation Result of RBFNN based MPC on AFR Using Secant Method

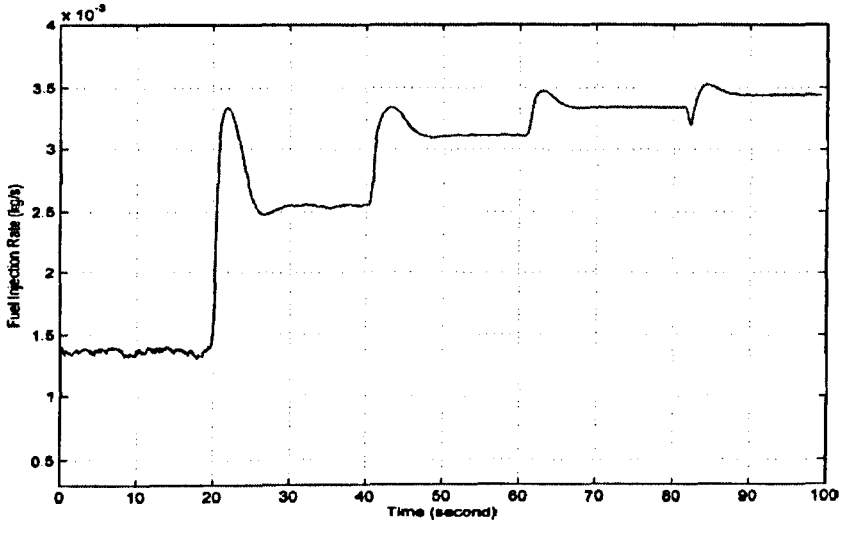

Figure 6.10: $\dot{m}_{fi}$ Produced by RBFNN based MPC Method Using Secant Method

subject to

$$\dot{m}_{fi}^{l} \leq \dot{m}_{fi} \leq \dot{m}_{fi}^{u} \tag{6.13}$$

where $f : R^n \rightarrow R$, $\dot{m}_{fi}^{l}$ and $\dot{m}_{fi}^{u}$ represent the lower bound and the upper bound of the control variable $\dot{m}_{fi}$.

The system output under the developed MPC is displayed in Figure 6.9, together

with the associated manipulated variable displayed in Figure 6.10. The MAE of the AFR tracking is 0.0483. One can see that the air-to-fuel ratio is regulated within a neighborhood of stoichiometric. This performance is much better than that of PI controller [49] that is widely used in automotive industry.

The time cost in optimization in each sample period is shown in Figure 6.11. The mean time cost in one sample period is 0.0277 seconds. Since the whole

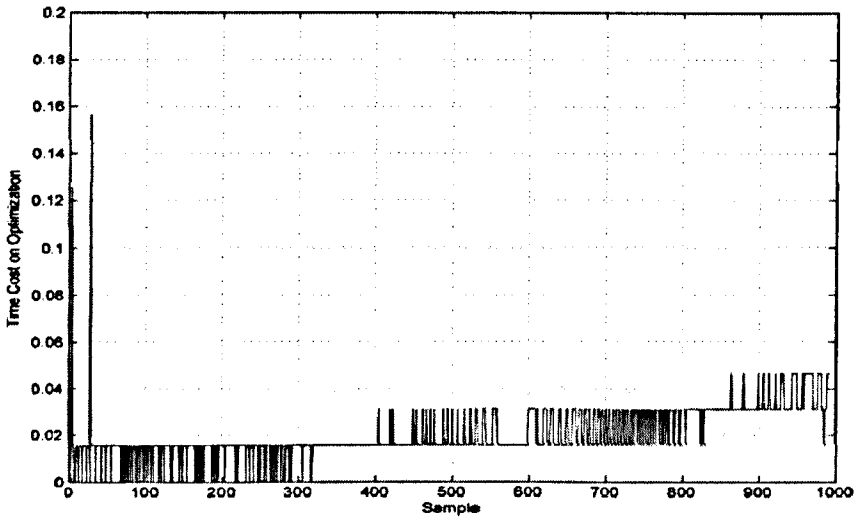

Figure 6.11: Time Cost on Optimization Using Secant Method

simulation was running in *MATLAB* environment, the further reduction on time cost of optimization could be achieved if optimization algorithm is realized by C code in practical application.

6.5 Multi-dimensional optimization approach

The multi-dimensional approach for MPC was implemented using reduced Hessian method and is compared with secant method, in terms of the control performance and time consumptions on optimization. The Reduced Hessian Method is reviewed in the following.

6.5.1 Reduced Hessian Method

By applying sequential quadratic programming(SQP), the general nonlinear programming problem reduces to solving the following quadratic programming (QP) at each iteration. Find d that minimizes

$$J(d) = min\{g(x_k)^T d + \frac{1}{2} d^T W(x_k) d\} \tag{6.14}$$

subject to

$$c_{eq}(x_k) + A_{eq}(x_k)^T d = 0 \tag{6.15}$$

$$c_{in}(x_k) + A_{in}(x_k)^T d \leq 0 \tag{6.16}$$

Here, g denotes the gradient of f, W denotes the Hessian Matrix (with respect to x) of the Lagrangian function $L(x, \lambda) = f(x) + \lambda^T c(x)$, $A_{eq}(x)$ and $A_{in}(x)$ stand for the $n \times m$ and $n \times p$ matrices of constraint gradients.

Among different SQP methods, the reduced Hessian method is a newly developed algorithm for solving NLP problems subject to equality constraints ($c_{eq}(x) = 0$) [71] [72] . It has been shown that the method is robust and less expensive to implement. In order to illustrate how it can be implemented into MPC of SI engines, the basic idea of the reduced Hessian method is discussed in this section. In the following parts of this section, $c(x)$ and $A(x)$ are used to represent c_{eq} and $A_{eq}(x)$ respectively.

The Search Direction

Assuming

$$A_k^T Z_k = 0 \tag{6.17}$$

then Z_k is a basis for the tangent space of the constraints. Now the solution d in Equation (6.14) can be stated as

$$d_k = Y_k p_y + Z_k p_z \tag{6.18}$$

where Z_k is a matrix spanning the null space of A_k^T , Y_k is a matrix spanning the range of A_k, p_Y and p_z are vectors in R^m and R^{n-m}, respectively. The problem of (6.18) becomes solving Y_k,p_Y,Z_k,p_z,. By grouping the components of x into m basic or dependent variables and $n-m$ non-basic or control variables, and $A(x)^T$ can be written as follows.

$$A(x)^T = [C(x)N(x)] = \left[\left(\frac{\partial c}{\partial x}\right)^T \left(\frac{\partial c}{\partial u}\right)^T\right] \tag{6.19}$$

$C(x)$ is assumed to be non-singular. Then defining

$$Z_k = \begin{bmatrix} -C(x)^{-1}N(x) \\ I \end{bmatrix} \tag{6.20}$$

$$Z_k = \begin{bmatrix} -N(x) \\ C(x) \end{bmatrix} \tag{6.21}$$

$$Y_k = \begin{bmatrix} I \\ 0 \end{bmatrix} \tag{6.22}$$

According to Equation (6.17), the following equation is obtained

$$p_Y = -\left[A_k^T Y_k\right]^{-1} c_k \tag{6.23}$$

Substituting p_Y into Equation (6.18) and then into Equation (6.15) to compute the minimum value under the assumption that $Z_k^T W_k Z_k$ is positive definite, the solution is

$$p_z = -\left(Z_k^T W_k Z_k\right)^{-1}\left[Z_k^T g_k + Z_k^T W_k Y_k p_Y\right] \tag{6.24}$$

$Z_k^T W_k Z_k$ is the reduced Hessian. In order to eliminate the computational work required to evaluate W_k and $Z_k^T W_k Z_k$, the BFGS algorithm is used to approximate the reduced Hessian $Z_k^T W_k Z_k$. As for the cross term $Z_k^T W_k Y_k p_Y$, there are two methods to deal with it. The first one is to omit it and the second one is to use a vector w_k as its approximation by means of finite-difference approximation or quasi-Newton approximation (Broyden's update).

Line Search and Stopping Criterion

Before updating $x*$ by using $x_{k+1} = x_k + \alpha_k d_k$, α_k needs to be tested according to

$$\phi_{\mu k}(x_k + \alpha_k d_k) \leq \phi_{\mu k}(x_k) + 0.1\alpha_k D\phi_{\mu k}(x_k; d_k) \tag{6.25}$$

where $\phi_{\mu k}(x_k) = f_k + \mu_k \|c_k\|_l$ and $D\phi_{\mu k}(x_k; d_k) = g_k^T d_k - \mu_k \|c_k\|_1$ (μ_k is chosen by users). If the above formula is not satisfied, a new α_k should be chosen as

$$\alpha_k = \max\left\{\frac{-0.5 D\phi_{\mu k}(x_k; d_k)\alpha_k^2}{\phi_{\mu k}(x_k + \alpha_k d_k) - \phi_{\mu k}(x_k) - \alpha_k D\phi_{\mu k}(x_k; d_k)}, 0.1\right\} \tag{6.26}$$

If a solution of this optimization problem is denoted by $x*$, define $e_k = x_k - x*$ and $\sigma_k = \max\{\|e_k\|, \|e_{k+1}\|\}$. If $\sigma_k \leq tol$ (tol is defined by users.), the algorithm can be stopped.

6.5.2 Simulation Results

During the implementation of Reduced Hessian Method, the parameters of the nonlinear optimization were the same as one-dimensional case. To solve the MPC optimization problem by reduced Hessian method, two slack variables x_1 and x_2 are introduced to convert the inequality constraint to become two equality constraints. Correspondingly, a quadratic penalty function $x_1^2 + x_2^2$ with a parameter μ_s is added into the objective function J in Equation (6.1). Then the MPC op-

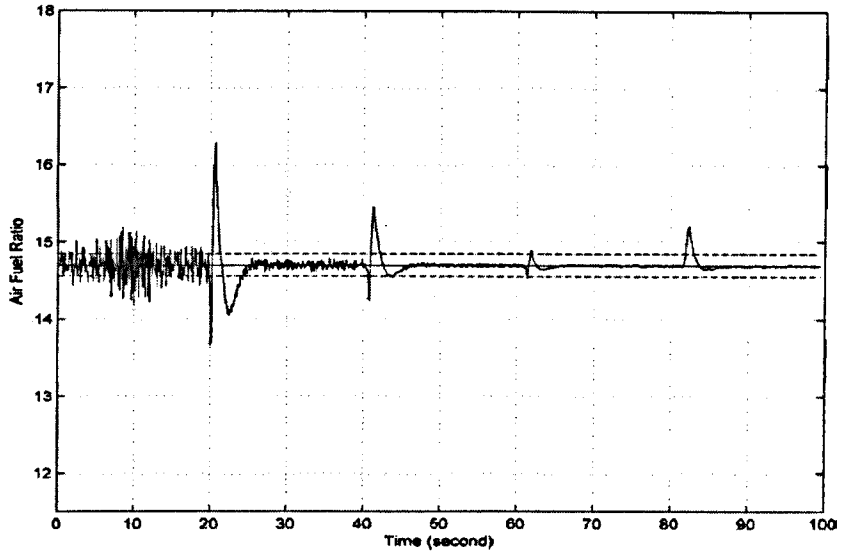

Figure 6.12: Simulation Result of RBFNN based MPC on AFR Using Reduced Hessian Method

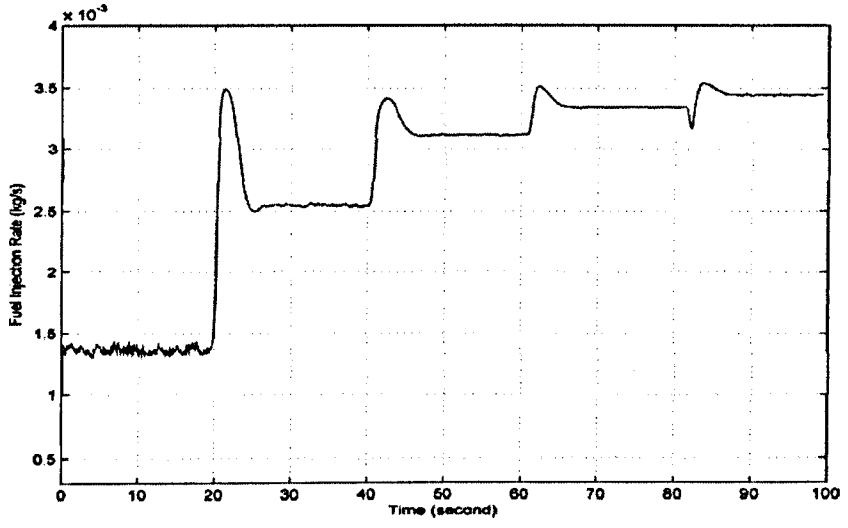

Figure 6.13: $\dot{m}_{fi}$ Produced by RBFNN based MPC Method Using Reduced Hessian Method

timization problem becomes: Find a suitable control variable $\dot{m}_{fi}$ to minimize

$$f = J + \mu_s(x_1^2 + x_2^2) \tag{6.27}$$

subject to

$$(\dot{m}_{fi} - \dot{m}_{fi}^{l}) - x_1^2 = 0 \tag{6.28}$$

$$(\dot{m}_{fi} - \dot{m}_{fi}^{u}) - x_2^2 = 0 \tag{6.29}$$

Here $\dot{m}_{fi}^{l}$ and $\dot{m}_{fi}^{u}$ are respectively 0.0014 kg/sec and 0.0079 kg/sec. After being scaled, the two bounds become 0 and 1. In order to reduce the influence of the slack variables x_1 and x_2 on the new objective function, the parameter μ_s is chosen to be 10^{-10} . To stop the optimization program at a suitable time, the tolerance *tol* is set be 10^{-7}.

With the above modification and parameters setting, the simulation results are shown in Figure 6.12 and 6.13 , the tracking mean absolute error is 0.0335. The time cost in optimization is shown in Figure 6.14 for comparison with the performance of Secant Method. It can be seen in Figure 6.14 that the optimization time used in one sample period is more than the sample period 0.1s when the sample instant $k > 470$. This would not be implemented in practice even if the ECU has the same computational power as the PC used to do simulation. However, considering the optimization time is much less when *C* language is used coding control algorithms, instead of *MATLAB*, the method is still possible to implement in real-time control.

6.5.3 Algorithms Comparison

In term of control performance, the simulation results show that if compared with secant method, reduced Hessian method on AFR control can obtain smaller overshoot under step disturbance. Its tracking performance is similar as that obtained by secant method. However, the computation load is one of the important issues in practical application of ECU. In our experiment, the mean time cost in one sample period using reduced Hessian method is 0.0473 seconds that

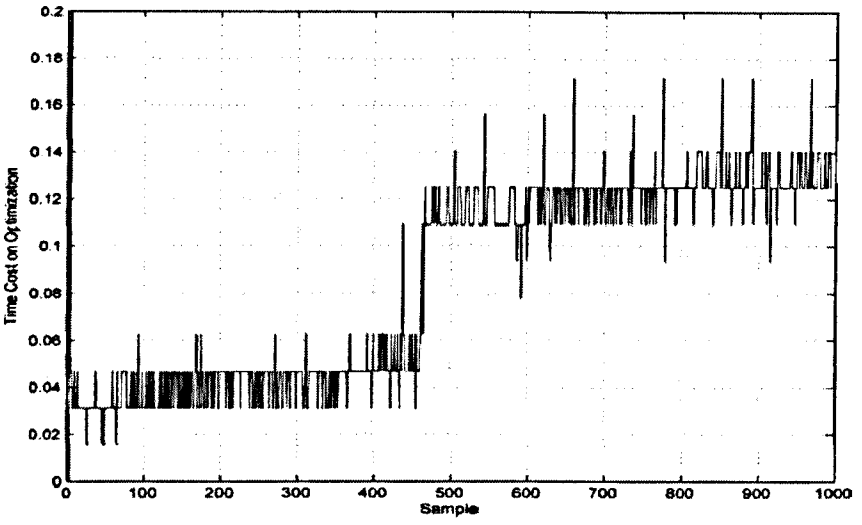

Figure 6.14: Time Cost on Optimization Using Reduced Hessian Method

is nearly twice as many as that used by secant method. From this perspective, the single-dimensional optimization approach implemented by secant method in this application is much better than multi-dimensional optimization approach by reduced Hessian method.

6.5.4 Control Performance with 10% Uncertainty

Air leakage problem is introduced to test the robustness of DRNN-base MPC on AFR. With the same simulation condition as previous section, 10% system uncertainty has been introduced to engine model after running for 26 seconds, to test the robustness of developed neural network-based MPC system to deal with air leakage problem. Figure 6.15 and 6.16 show the control performance obtained by RBFNN-based MPC method. The tracking error by MAE is 0.0881. From the simulation results, it can be seen that the internal RBFNN model used for model predictive control was not trained using the data with system uncertainty, however, adaptive training method adjusts the neural parameters according to the change of engine dynamics. This guarantees the correct prediction on AFR. The control algorithm therefore can produce the fuel injection rate appropriately.

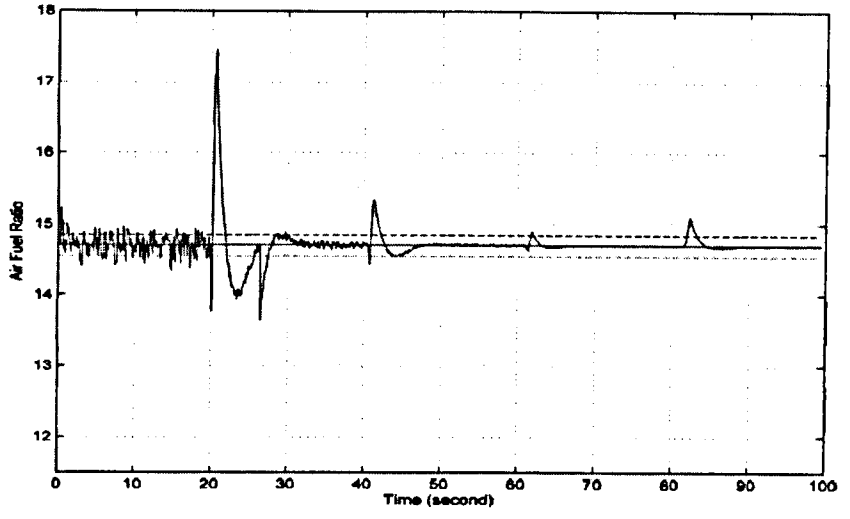

Figure 6.15: Simulation Result of RBFNN based MPC on AFR When Engine Model Has 10% Mismatch

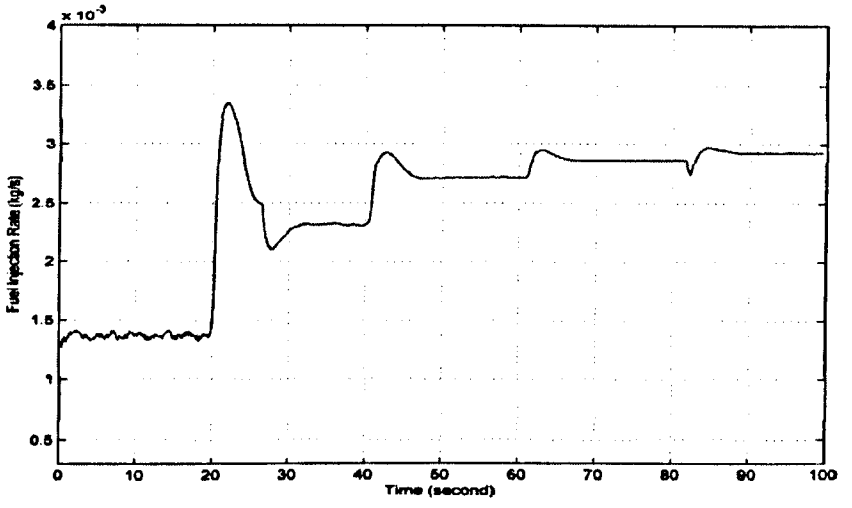

Figure 6.16: $\dot{m}_{fi}$ Produced by RBFNN based MPC Method When Engine Model Has 10% Mismatch

6.5.5 Control Performance with 30% Uncertainty

The system uncertainty has been increased to 30% here. The simulation results shown in Figure 6.17 and 6.18. It can be seen that, although there is a 20% overshoot of AFR when 30% system mismatch introduced, the AFR converges quickly into the ± 1 % bounds of the stoichiometric value(14.7). The online adaptation on neural parameters by RLS algorithm makes a good contribution for this

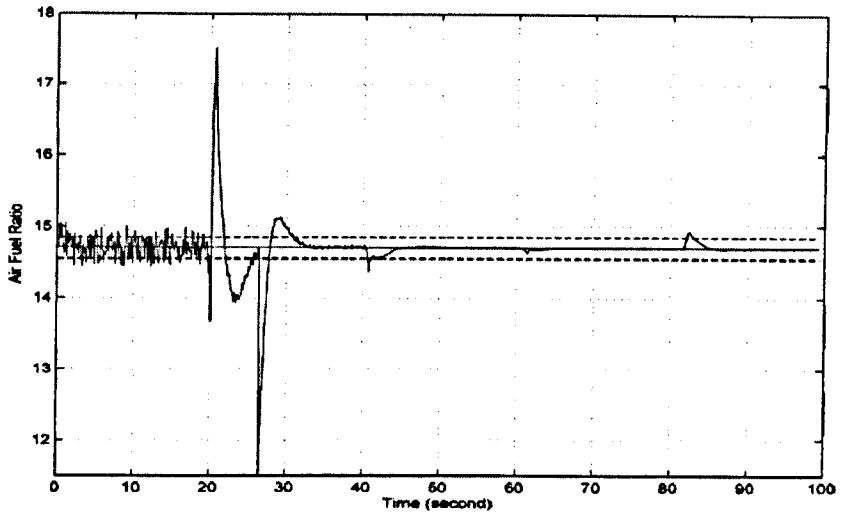

Figure 6.17: Simulation Result of RBFNN based MPC on AFR When Engine Model Has 30% Mismatch

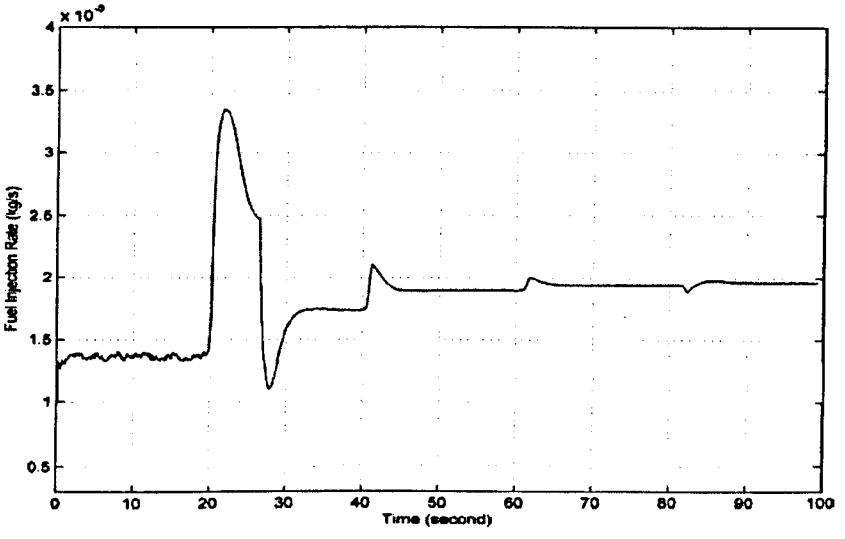

Figure 6.18: $\dot{m}_{fi}$ Produced by RBFNN based MPC Method When Engine Model Has 30% Mismatch

improvement on control performance. As the NMPC structure shown in Figure 6.1, there is a feedback of the AFR error between the engine output and RBFNN model output, which is used to adjust the value of AFR set-point. Therefore, even there is no on-line adaptation on neural parameters, it is still possible for the AFR to converge toward stoichiometric value(14.7) by this adjustment mechanism of NMPC. However, the RBFNN engine model in this research is trained

and updated online with RLS algorithm. Therefore, a better prediction on AFR can be achieved under system uncertainty. This improved accuracy on predicted AFR is very important to deal with the changes on system dynamics. The simulation result proves that the developed control method is robust enough to 30% uncertainty in the air intake system. The tracking error in this simulation by MAE is 0.1017.

6.6 Control Performance Comparison with PID

Under same simulation environment, the air fuel ratio control results of both PID control and RBFNN-based MPC are shown in Figure 6.19 and 6.20. Compared with traditional feedback control, the RBFNN-based MPC on AFR has better tracking performance. In addition, as discussed previously in Chapter 2, there is a time delay on the air fuel ratio measurement by the oxygen sensor, the overshoot in the PID control reaches 51.7%. The overshoot for the RBFNN-based MPC is just 12.9%, which is much smaller than PID's. This is because the RBFNN model used here has the throttle angle signal as one of its inputs. Therefore, when such a model is used for predictive control, the optimization algorithm can take the change of throttle angle into account and produce an appropriate fuel injection rate to minimize the influence it brought.

6.7 Comparison between RBF based MPC and Feedforward-Feedback Control

The air fuel ratio control results under same simulation environment, are shown in Figure 6.21 and 6.22, for both RBFNN-based MPC and FFC method. Obviously, in term of control performance, feedforward and feedback control is better than MPC method, as the RBF-based feedforward controller monitors the system states and can take correct action before the disturbances upset the process.

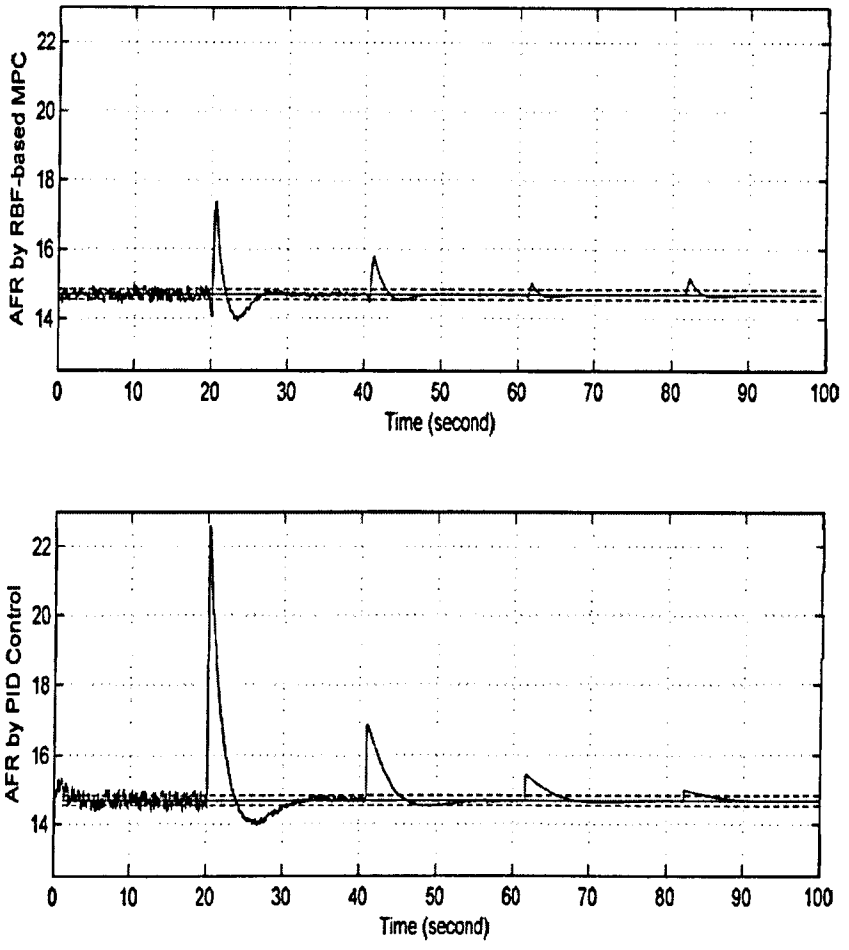

Figure 6.19: Comparison of AFR Control Result between RBFNN based MPC and PID

However, RBF-based MPC still has some advantages over FFC method. By the comparison of the structure of these two control methods, it can be seen that, in the RBF-based MPC, the controller only needs the three values - throttle angle, fuel injection rate and air fuel ratio for online adaptation of neural networks and the computation of control action. For the FFC method, besides the three measurements above, engine speed, the temperature in intake manifold and air flow rate into the engine port are needed to compute the control action. Therefore, taking hardware implementation into account, the RBFNN-based MPC method can achieve good control performance on AFR, at the same time, reduce the system complexity in a large degree.

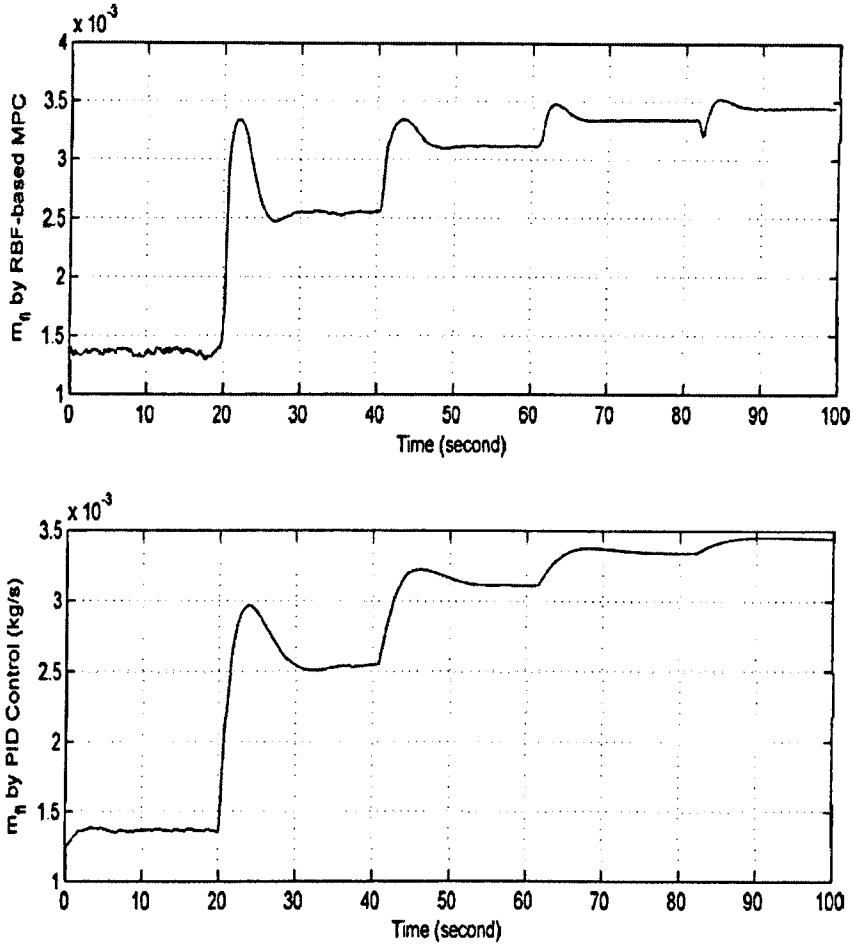

Figure 6.20: Comparison of $\dot{m}_{fi}$ Produced by RBFNN based MPC and PID

6.8 Summary

This chapter has discussed the RBFNN-base MPC scheme for air fuel ratio of SI engine. The adaptive RBF network is trained by the RLS method for modelling AFR dynamics. Based on the engine model, the MPC scheme is realized for controlling AFR. According to different MPC structures, the advantage and disadvantage of both single dimension and multi-dimension optimization methods are discussed. The robustness of this model predictive control is tested. The control performance by RBFNN-based MPC for AFR is much better than that of a traditional PI controller in a wide engine operating condition. Based on the comparison of control performance between RBFNN-based MPC and RBFNN-

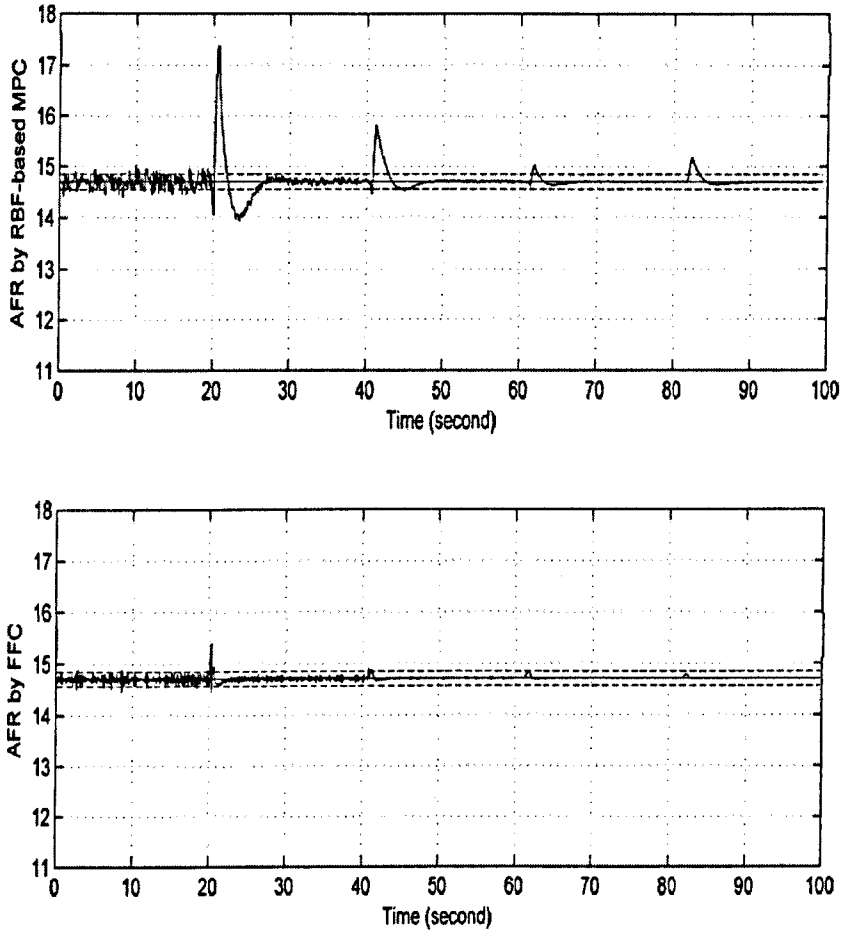

Figure 6.21: Comparison of AFR Control Result between RBFNN based MPC and RBFNN based Feedforward and Feedback Control

based feedfoward and feedback control, the advantage of MPC method that can be brought for engine management system is given.

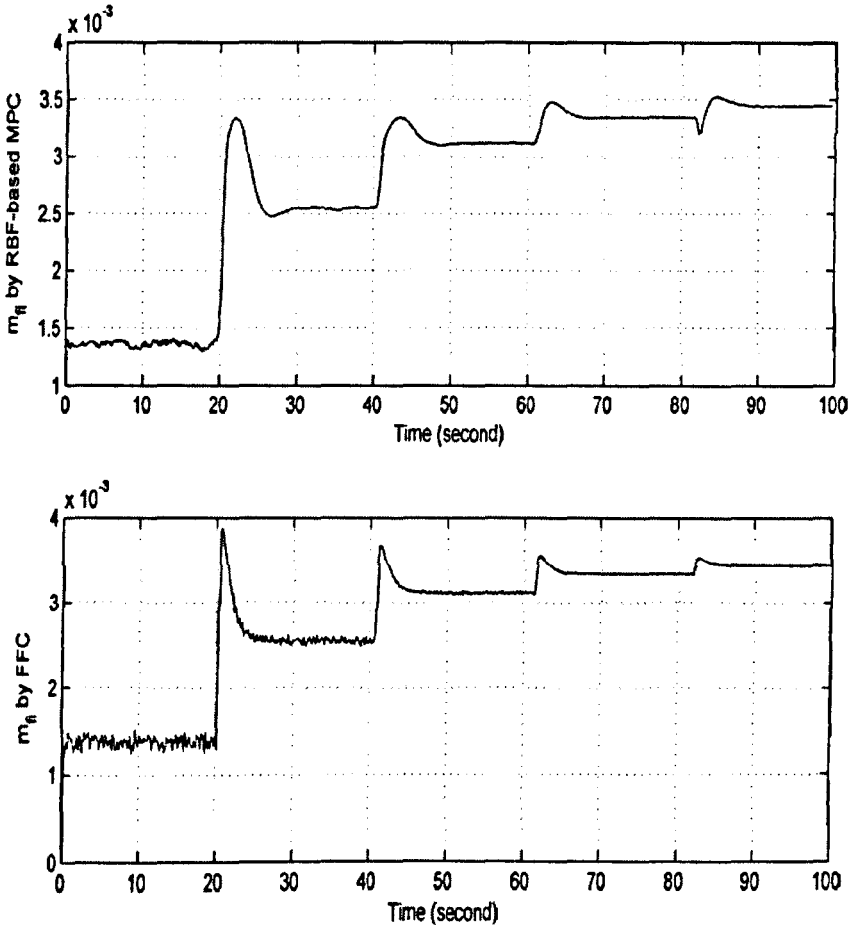

Figure 6.22: Comparison of $\dot{m}_{fi}$ Produced by RBFNN based MPC and RBFNN based Feedforward and Feedback Control

Chapter 7

DRNN-based Model Predictive Control

7.1 Introduction

The radial basis function network used previously for MPC represents a class of neural networks known as nonlinear layered feedforward networks. In this chapter another important class of neural networks that have a recurrent structure is considered, for model predictive control on air fuel rate of SI engine. Such recurrent neural networks have the ability to store information for later use, thus they are better suited for dynamical system modelling than the feedforward network. There are different structures of recurrent neural networks available, such as continuous-time neural network (CTRNN) [73], fully connected recurrent neural network (FRNN) [74], and etc. However, the computation for network training and online adaptation of these RNNs is very heavy and not suitable for real-time implementation in AFR control application. In our research, the diagonal recurrent neural network (DRNN) is chosen for engine modelling for nonlinear model predictive control on AFR. The architecture of DRNN is a modified mode of the fully connected recurrent neural network with one hidden layer, and the hidden layer is comprised of self-recurrent neurons as shown in Figure 4.3. This sim-

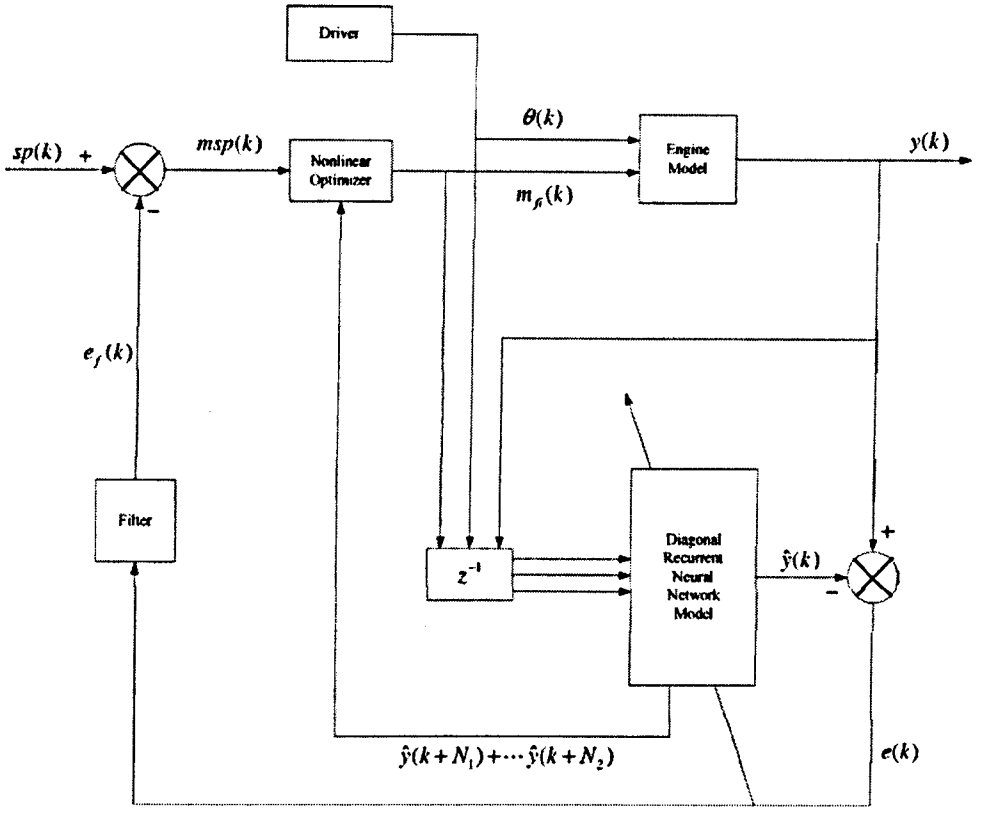

Figure 7.1: Control Structure of DRNN-based MPC

plification has considerably fewer weights than FRNN. Therefore, the controller based on this kind of recurrent neural network can converge with a relatively small number of training cycles. The dynamic backpropagation (DBP) training algorithm is used for the training of DRNN.

7.2 Control System Configuration

The strategy of DRNN-based model predictive control on air fuel ratio is shown in Figure 7.1. The trained adaptive diagonal recurrent neural network model is used to predict the engine output for N_2 steps ahead. The nonlinear optimizer finds out the optimal control variable online by minimizing the errors between

the set point and the predicted engine output by using the cost function.

$$J(k) = \sum_{i=k+N_1}^{k+N_2} [msp(i) - \hat{y}(i)]^2 + \xi \sum_{i=k}^{k+N_u} [\dot{m}_{fi}(i) - \dot{m}_{fi}(i-1)]^2 \qquad (7.1)$$

Here, N_1 and N_2 define the prediction horizon. ξ is a control weighting factor which penalizes excessive movement of the control input, the fuel injection $\dot{m}_{fi}$. N_u is the control horizon. Then, calculate a series of optimal $\dot{m}_{fi}(k)$, $\dot{m}_{fi}(k+1)$, $\cdots$, $\dot{m}_{fi}(k+N_u-1)$ from which the neural network model generates outputs to minimize the objective function J in Equation (7.1). Due to the operating limitation of fuel injector, there are constraints on the control variable - fuel injection rate $\dot{m}_{fi}$. Therefore, the problem to be solved here is of nonlinear optimization with constraints. Considering the computational efficiency, secant method is used in our simulation, which has been introduced previously. Finally, the first control variable $\dot{m}_{fi}(k)$ is used to control the process. This procedure is repeated in the next sample period.

7.3 Engine Modelling using DRNN

7.3.1 Data Collection

To obtain the engine data for DRNN modelling, two sets of RAS were designed for throttle angle θ and fuel injection rate m_{fi}. The ranges of these excitation signals are listed in Table 7.1 The sample time in the simulation was set to 0.1 sec. The

RAS	Min	Max
θ	20°	70°
$\dot{m}_{fi}$	7×10^{-4}	7.9×10^{-3}

Table 7.1: Range of Excitation Signals for AFR Modelling Using DRNN

simulated engine model MVEM was run for 500 seconds with a set of 5000 data samples collected for all three input and one output variables. These data were divided into two groups: the first 4500 data samples were used for training while

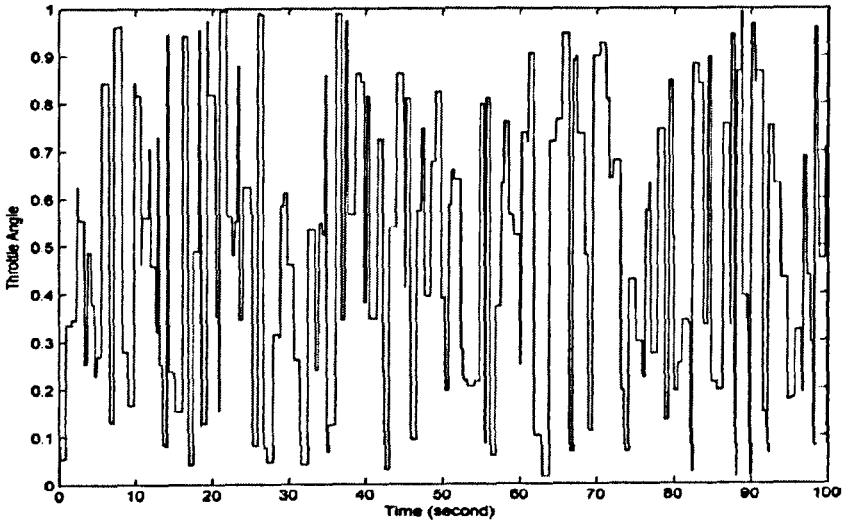

Figure 7.2: Scaled Throttle Angle Data for DRNN Training

the remaining 500 data samples for validation. All input and output data of the DRNN have been scaled to the range of $[0, 1]$ before they are used for training and validation. Figure 7.2, 7.3, and 7.4 show the first 1000 samples of the data for DRNN model training.

7.3.2 DRNN Model Structure

Given the expanded engine model as shown in Figure 2.16, the DRNN engine model for AFR control has three inputs: fuel injection rate $\dot{m}_{fi}$, throttle angle θ and air-fuel ratio AFR , and one output: air-fuel ratio. The DRNN model structure is shown in Figure 7.5. Different orders of network inputs and different number of hidden layer nodes have been used in training experiments. In the simulation study, the neural parameters $n = 3$, $q = 13$, $p = 1$, $v = 2$ were determined to be the best DRNN modelling structure, which gives the minimum prediction error. The training and online updating of DRNN model are realized by dynamic backpropagation algorithm [52] and the initial learning rate of DRNN, η^h, η^d and η^y are set to 1×10^{-7}.

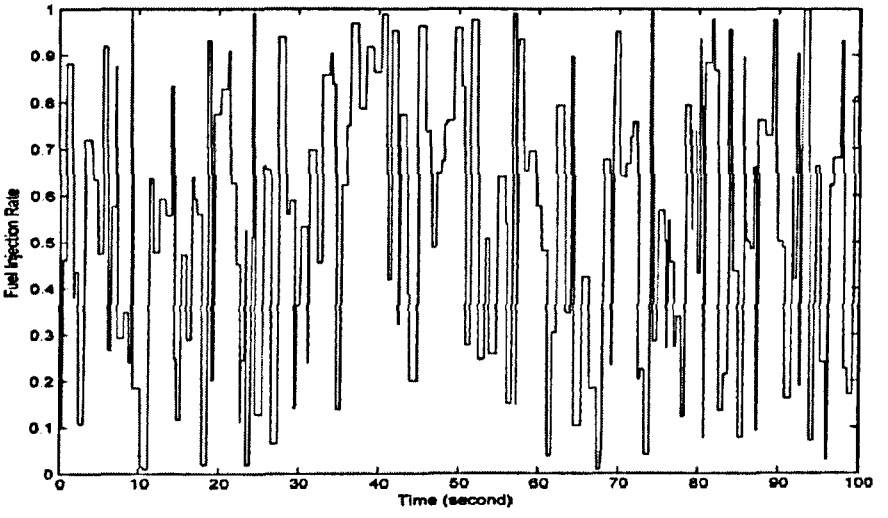

Figure 7.3: Scaled Fuel Injection Rate Data for DRNN Training

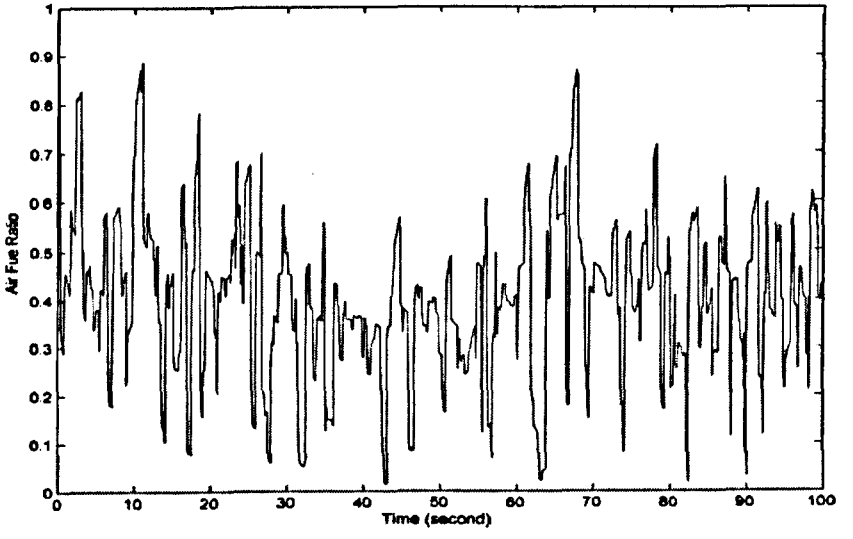

Figure 7.4: Scaled Air Fuel Ratio for Data RBFNN Training

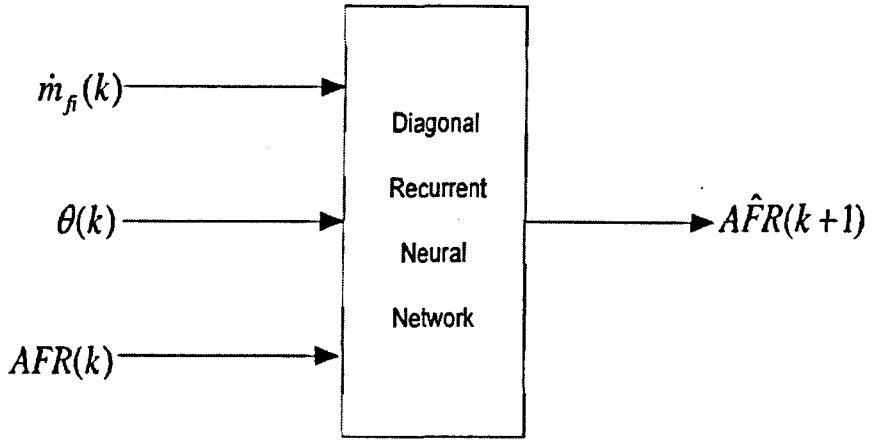

Figure 7.5: DRNN Model Structure

7.3.3 DRNN by ADOL-C

In this simulation, the DRNN engine model is built by *C/C++* language and ADOL-C library. Therefore, we can benefit from the computation efficiency of gradient of error in DRNN by using automatic differentiation. Besides, the control algorithm developed by *C/C++* language is easy to be transferred to real application. The procedure is given as follows:

Declaring Active Variables

The key ingredient of automatic differentiation by overloading is the concept of an active variable. Typically, the independent variables, dependent variables, and all quantities that directly or indirectly depend on them are declared as active. For the training purpose in our application, we want to obtain the output gradients with respect to output, recurrent and input weights shown in Equation (4.25), therefore, the DRNN output $\hat{y}$ and three weight matrices W^h, W^y, W^d are defined as active variable.

Marking Active Sections

As defined by **ADOL-C**, all calculations involving active variables that occur between the void function calls *trace_on* and *trace_off* are recorded on a sequential data set called tape. The sequence of statements executed between a particular call to *trace_on* and the following call to *trace_off* is referred as an *active section* of the code. After the definition of active variables, the code to realize the diagonal recurrent neural network as described in Equation 4.19 through 4.23 must be written between *trace_on* and *trace_off* as *active section.*

Selecting Independent and Dependent Variables

ADOL-C marks independent and dependent variables using <<= and >>= statements. The derivative values calculated after the completion of an active section always represent derivatives of the final values of the dependent variables with respect to the initial values of the independent variables. Therefore, to make a proper function calling, all the variables in matrices W^h, W^y, W^d must be initialized by <<= at the beginning of *active section*, and the DRNN output $\hat{y}$ must be give an assignment by >>= at the end of *active section.*

Calling Routines in Drivers for Optimization

In **ADOL-C**, there are many drivers provided for solving optimization problems and nonlinear equations. The driver routine used to calculate the gradient of error $\partial E_m/\partial W$ in Equation (4.25) is *gradient*, its prototype can be found in the header file *<adolc/drivers/drivers.h>*.

Building a Interface between MATLAB and C++

MATLAB has the capability of running functions written in *C/C++*. The files which hold the source for these functions are called MEX-Files. As the engine model used in this research is developed in *MATLAB/Simulink* computation environment, to make the matrices of gradient of error for DRNN, which are

obtained by automatic differentiation technique, available for simulation in each sample time, an interface with *MATLAB* environment is built by MEX warp.

7.3.4 Engine Modelling Results by DRNN

After training DRNN for 60 epochs by DBP algorithm, the DRNN-based engine model is tested with validation data. The MAE of the trained DRNN model is 0.0252. Figure 7.6 shows the simulation result. It can be seen that the good match between the engine data and the DRNN output during the model validation phase.

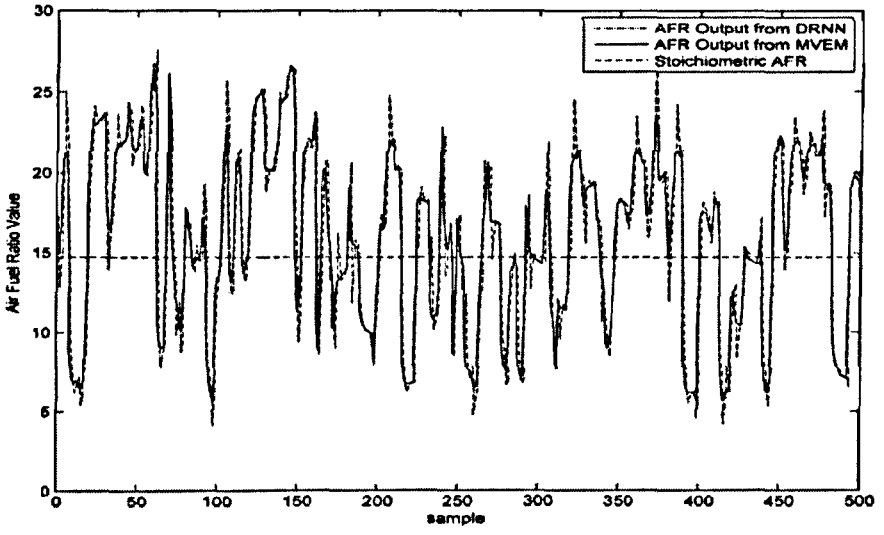

Figure 7.6: DRNN Modelling Result

7.4 DRNN-based MPC Simulation

7.4.1 Simulation Environment Setting

The developed DRNN is used as internal model in the MPC scheme for AFR control. The performance of the MPC scheme is evaluated by the simulating the system on the engine simulation model, the MVEM. In the simulation, the set-point of the system is set to be the constant stoichiometric value 14.7. The

throttle angle used as disturbance here is the same as that in previous simulations, it increases from 20° to 70° with 0.5° uncertainty, by changing 10° every 20 seconds as shown in Figure 5.10. The AFR is to be controlled between the bounds of the stoichiometric value(14.7). The sampling time is 0.1s, which is the same as that in engine modelling.

7.4.2 Tuning of Control Parameters

The control parameters in the objective function were chosen as $N_1 = 1$, $N_2 = 5$, as further increasing on the prediction horizon leads to heavier computation load. In fact, there is no significant improvement of control performance brought by a larger prediction horizon. As the optimization algorithm used to solve the objective function (7.1) is secant method that is a single-dimensional approach, the control horizon N_u is set to zero in this case. The setting of control weighting factor is by trial-and-error. In our simulation, if ξ is too small, the control system becomes unstable when throttle angle of MVEM operates between 25° to 45°. If ξ is too large, it is very difficult to obtain a good tracking performance when a disturbance is introduced because too much penalties are given on excessive movement of the control input. Simulation result in this research shows that $\xi = 0.34$ is an appropriate setting for ξ.

7.4.3 Simulation Results

The system output - air fuel ratio obtained by DRNN-based MPC is shown in Figure 7.7. The control variable $\dot{m}_{fi}$ produced by the developed controller is displayed in Figure 7.8. The figure shows that the air-to-fuel ratio is regulated within a ± 1 neighborhood of stoichiometric value. The system makes prompt control actions by making a multi-step prediction on AFR when there are step changes on throttle angle. Therefore, a good control performance has been achieved in both steady state and transient state. The MAE (see Equation (4.43)) of the AFR tracking is 0.0248,

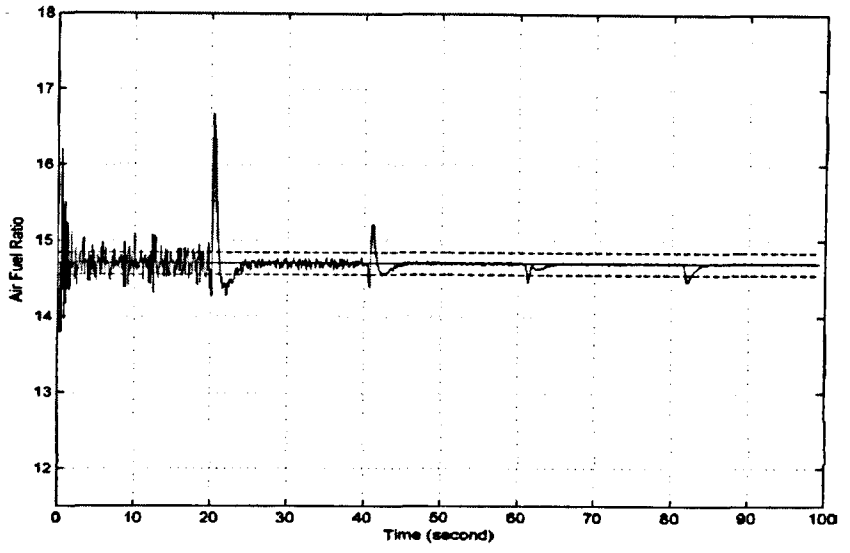

Figure 7.7: Simulation Result of DRNN based MPC on AFR

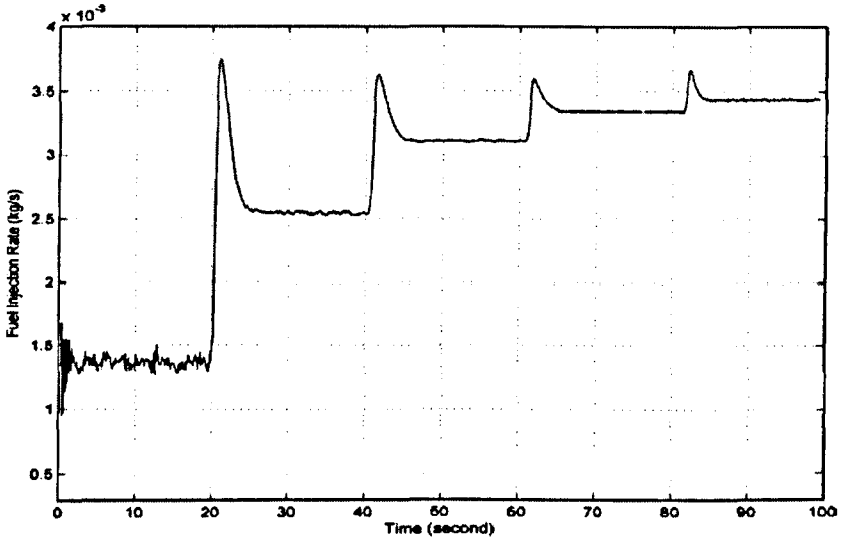

Figure 7.8: $\dot{m}_{fi}$ Produced by DRNN based MPC Method

7.4.4 Control Performance with 10% Uncertainty

As before, air leakage problem is used to test the robustness of DRNN-base MPC on AFR. With the same simulation condition in previous section, 10% system uncertainty has been introduced to engine simulation model after running for 26 seconds, to test the robustness of the developed neural network-based MPC system to deal with air leakage problem. Figure 7.9 and 7.10 show the control performance obtained by RBFNN-based MPC method. The tracking error by

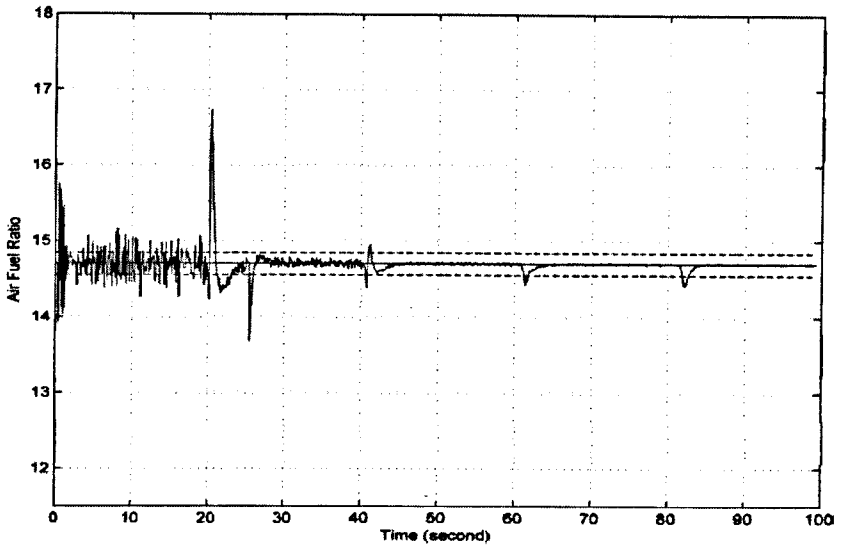

Figure 7.9: Simulation Result of DRNN based MPC on AFR When Engine Model Has 10% Mismatch

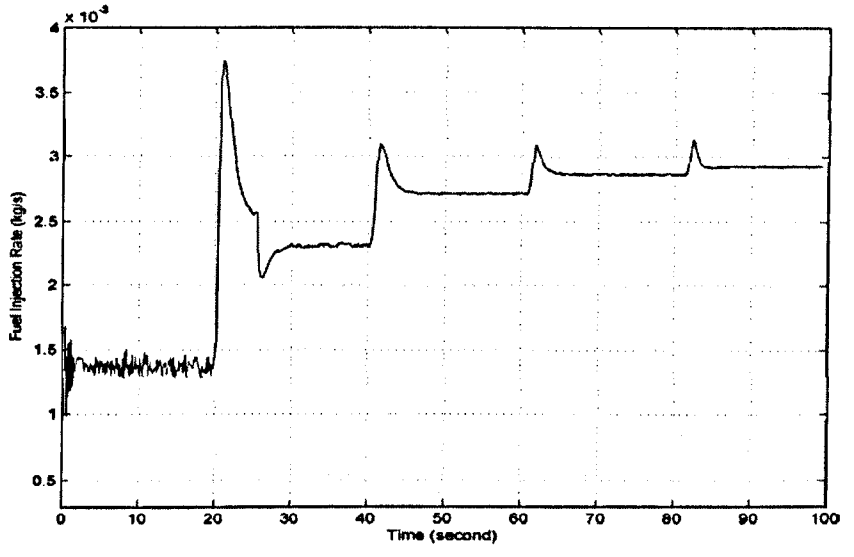

Figure 7.10: $\dot{m}_{fi}$ Produced by DRNN based MPC Method When Engine Model Has 10% Mismatch

MAE is 0.0277. It can be seen there is a 7.14% overshoot on AFR when there is a sudden reduction on the air mass flow into engine port due to the air leakage in manifold. Then, AFR comes back to the bounds of the stoichiometric value(14.7) within 2 seconds by the adjustment on the fuel injection rate. This is because that the DRNN model can adapt itself online by DBP algorithm to catch the change of engine dynamics. Consequently, the accurate prediction on AFR makes DRNN

based MPC produce the appropriate $\dot{m}_{fi}$ for stoichiometric operation.

7.4.5 Control Performance with 30% Uncertainty

Further investigation on system robustness under uncertainty has been done by introducing 30% mismatch of $\dot{m}_{ap}$ to the engine model. Simulation results are

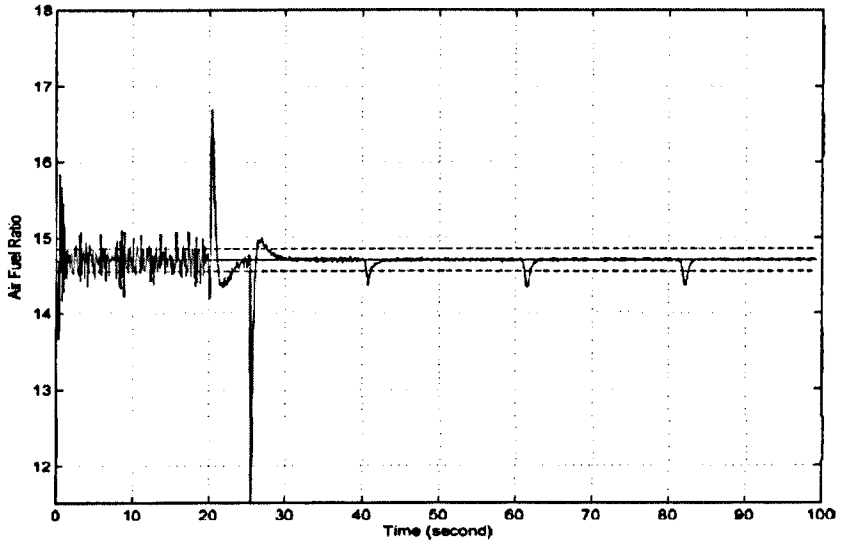

Figure 7.11: Simulation Result of DRNN based MPC on AFR When Engine Model Has 30% Mismatch

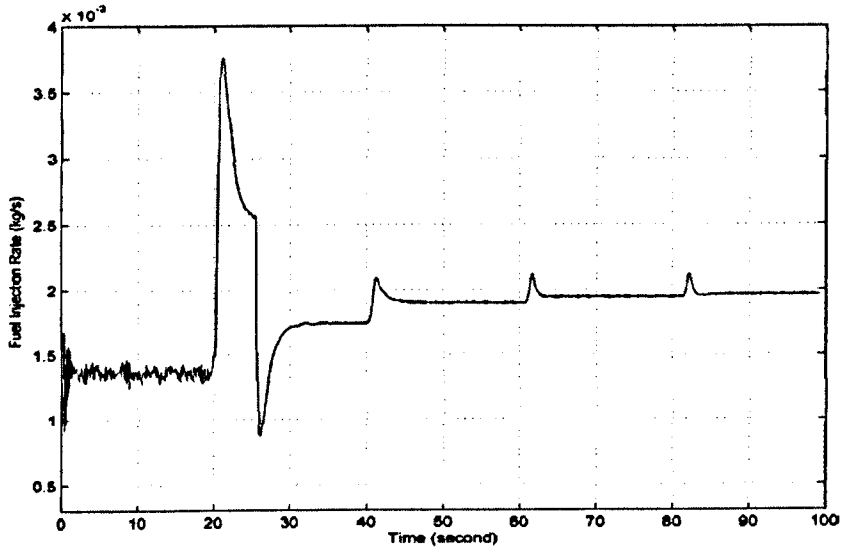

Figure 7.12: $\dot{m}_{fi}$ Produced by DRNN based MPC Method When Engine Model Has 30% Mismatch

shown in Figure 7.11 and 7.12. The tracking error by MAE is 0.0350. The overshoot of AFR has reached 21.7%. However, the AFR settles within the bounds of the stoichiometric value(14.7) after 2.3 seconds. In addition, a good control performance has been achieved in steady state. The result proves that the developed control method has the capability to deal with 30% uncertainty in the air intake system.

7.5 Comparison with PID Control

Under same simulation environment, the air fuel ratio control results of both PID control and DRNN-based MPC are shown in Figure 7.13 and 7.14. Compared

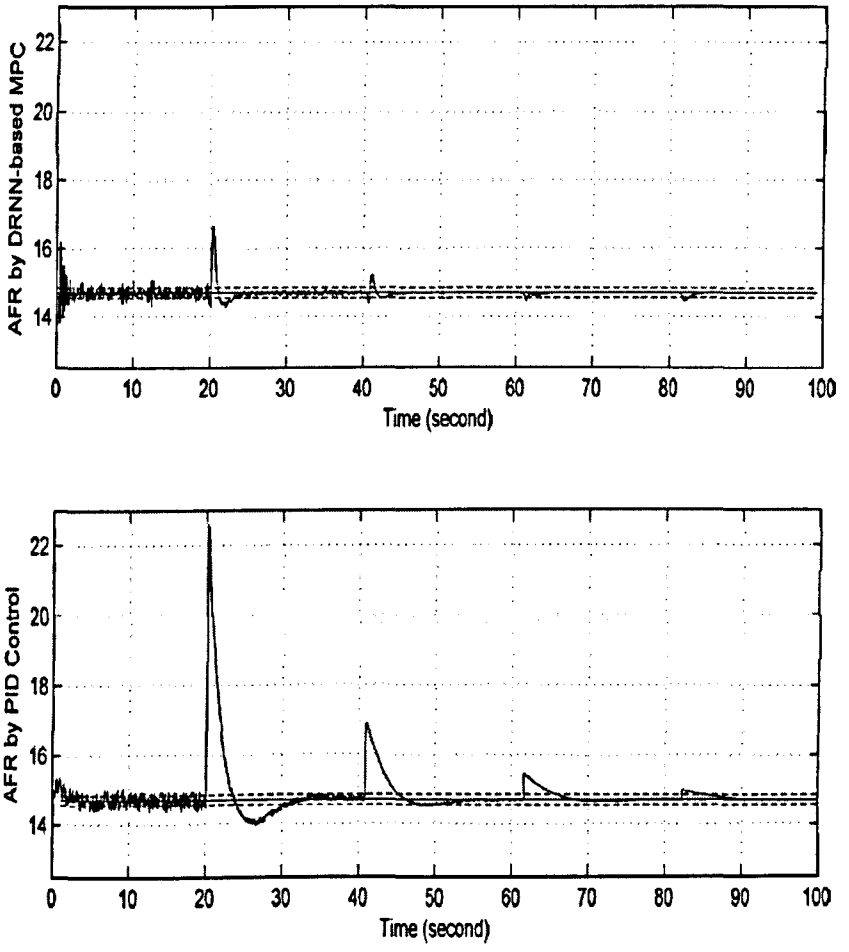

Figure 7.13: Comparison of AFR Control Result between DRNN based MPC and PID

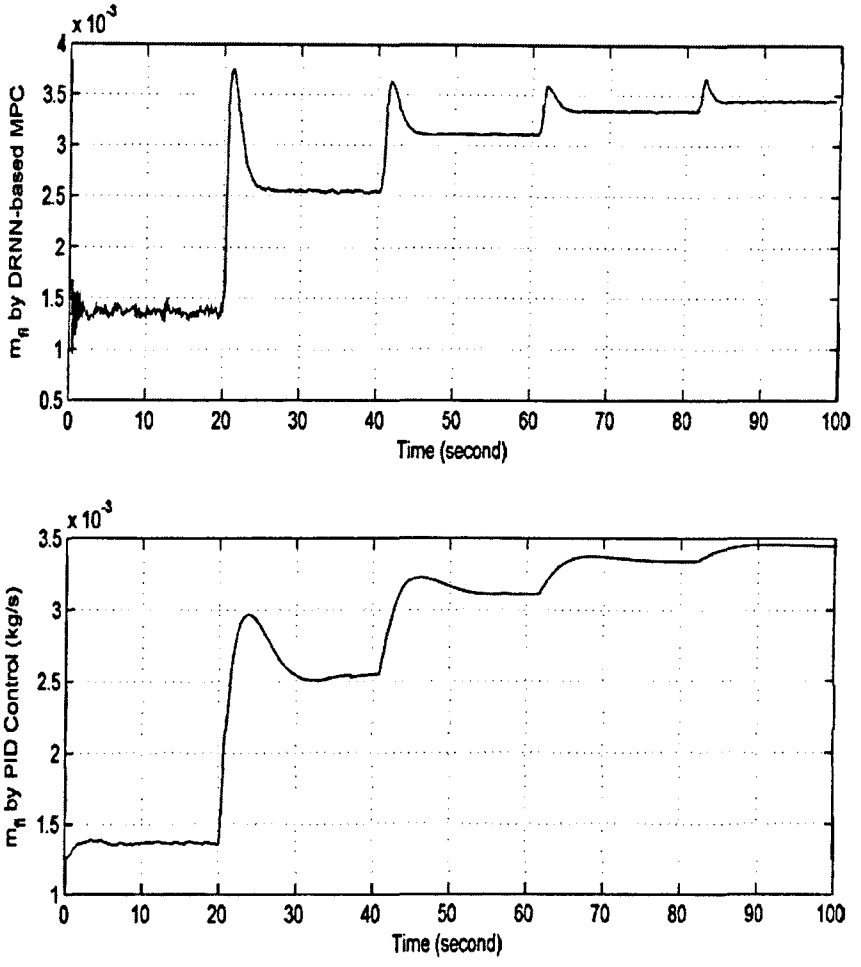

Figure 7.14: Comparison of $\dot{m}_{fi}$ Produced by DRNN based MPC and PID

with traditional feedback control, the DRNN-based MPC on AFR has better tracking performance. In addition, as discussed previously in Chapter 2, there is a time delay on the air fuel ratio measurement by the oxygen sensor, the maximum overshoot PID controller, caused by the change of throttle angle, reaches 51.7%. The maxim overshoot for the DRNN-based MPC is just 12.2%, which is much smaller than PID's.

7.6 Comparison with RBFNN based Feedforward and Feedback Control

The air fuel ratio control results, which is under same simulation environment, are shown in Figure 7.15 and 7.16, for both DRNN-based MPC and FFC method.

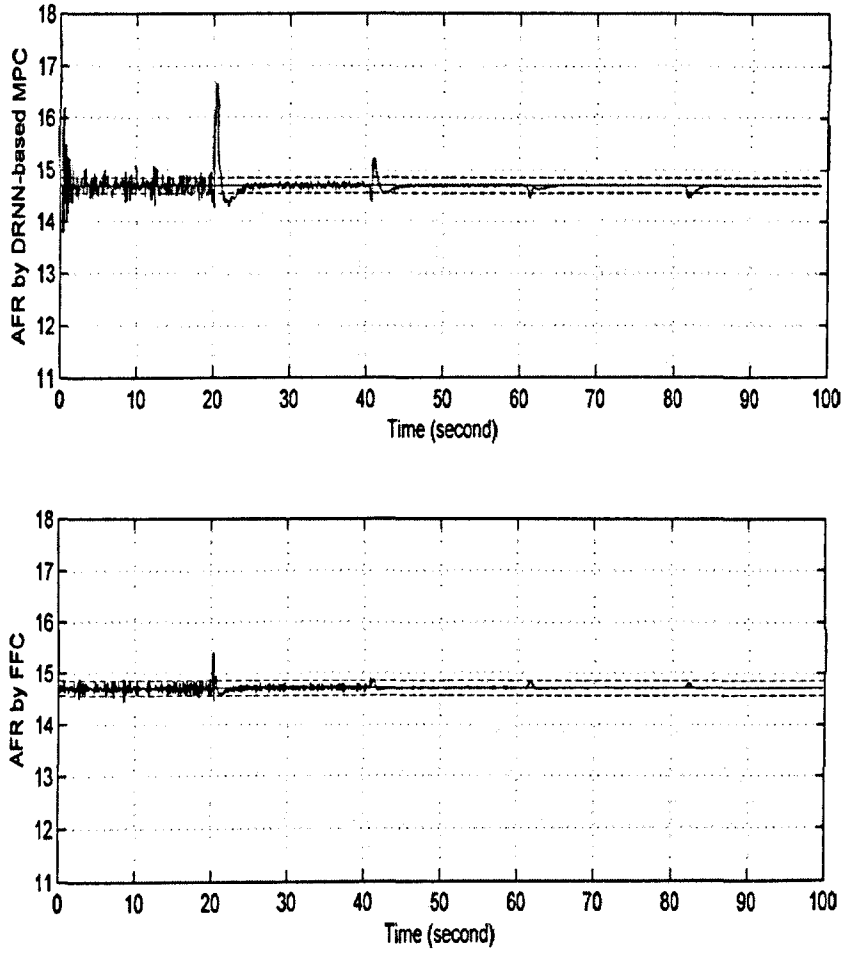

Figure 7.15: Comparison of AFR Control Result between DRNN based MPC and RBFNN based Feedforward and Feedback Control

In term of tracking performance, FFC method is better than the DRNN-based MPC. However, as discussed in Chapter 6, the advantage by implementing MPC is that, without losing too much tracking performance, the complexity of control system can be reduced significantly.

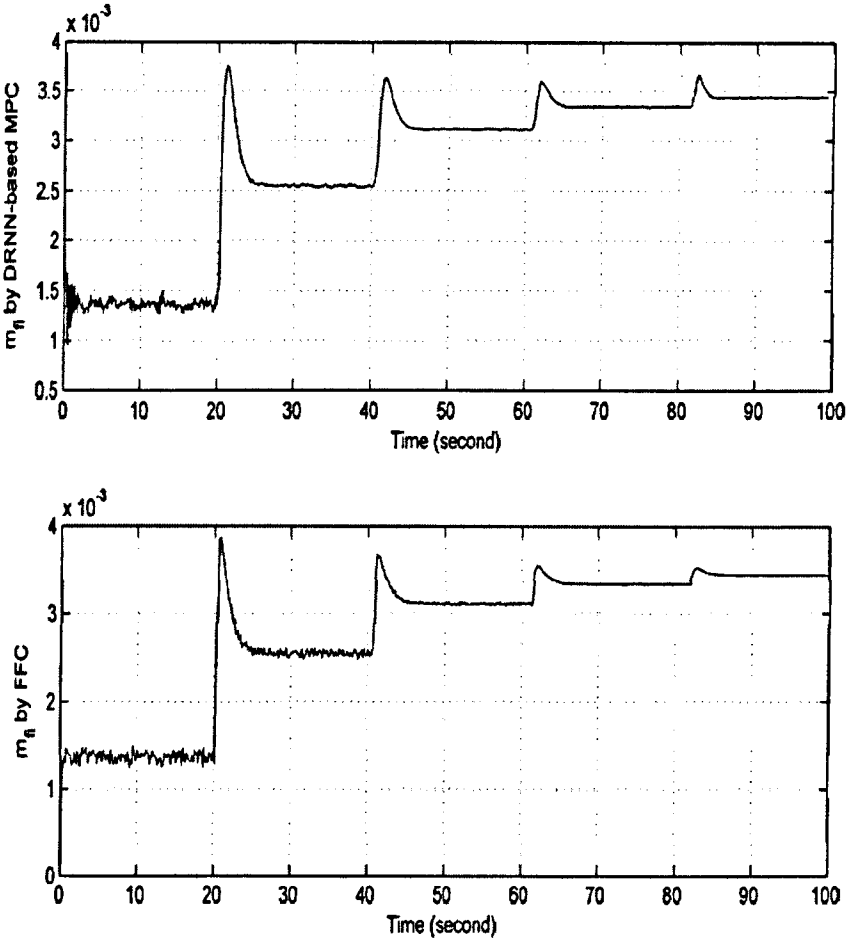

Figure 7.16: Comparison of $\dot{m}_{fi}$ Produced by DRNN based MPC and RBFNN based Feedforward and Feedback Control

7.7 Comparison with RBFNN based Model Predictive Control

The following simulation results in Figure 7.17 and 7.18 show that, given the same simulation condition and optimization algorithm, the tracking performance obtained by using DRNN-based model predictive control on air fuel ratio is better than that of RBF-based MPC. However, Diagonal recurrent neural network has more complicate structure than RBF neural network, which results in heavier computation load for micro-controller. In term of training algorithm, recursive least square method used for RBF neural network is more computationally ef-

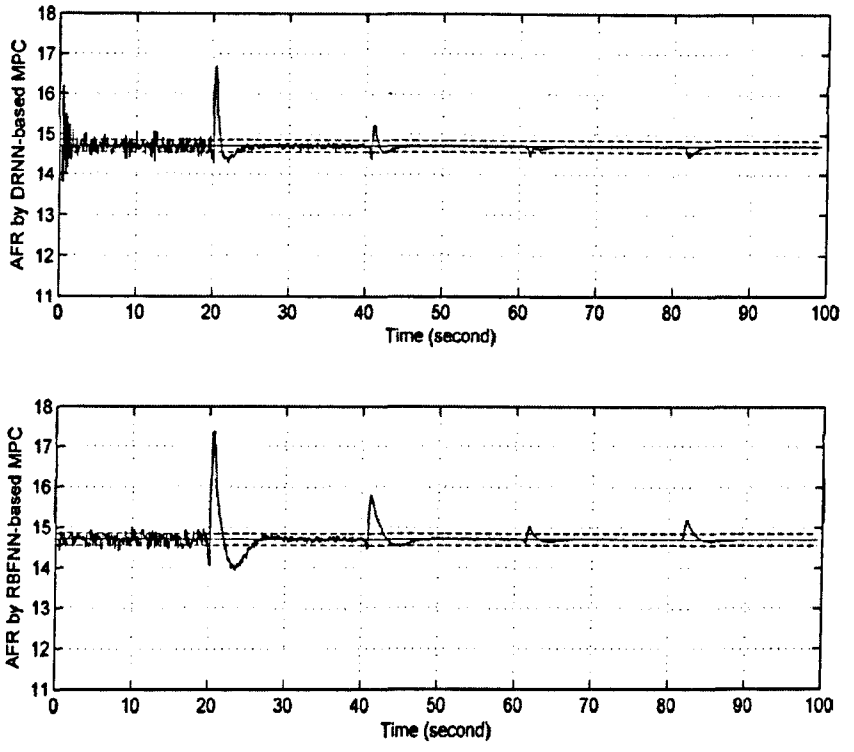

Figure 7.17: Comparison of AFR Control Result between DRNN based MPC and RBFNN based MPC

ficient than dynamic back-propagation algorithm for DRNN, which means that more powerful micro-controller is required if one wants to implement DRNN-based MPC for engine control system. Trade-off between hardware cost and control performance needs to be made in practical application.

7.8 Summary

This chapter has discussed the DRNN-base MPC scheme for air fuel ratio of SI engine. The adaptive DRNN trained by the dynamic back-propagation method is used for modelling AFR dynamics of an SI engine. Automatic differentiation technique is implemented in our application, to improve the computational efficiency of DRNN training algorithm. Based on this DRNN models, the MPC schemes are realized for regulating AFR. The robustness of this model predictive control is tested.

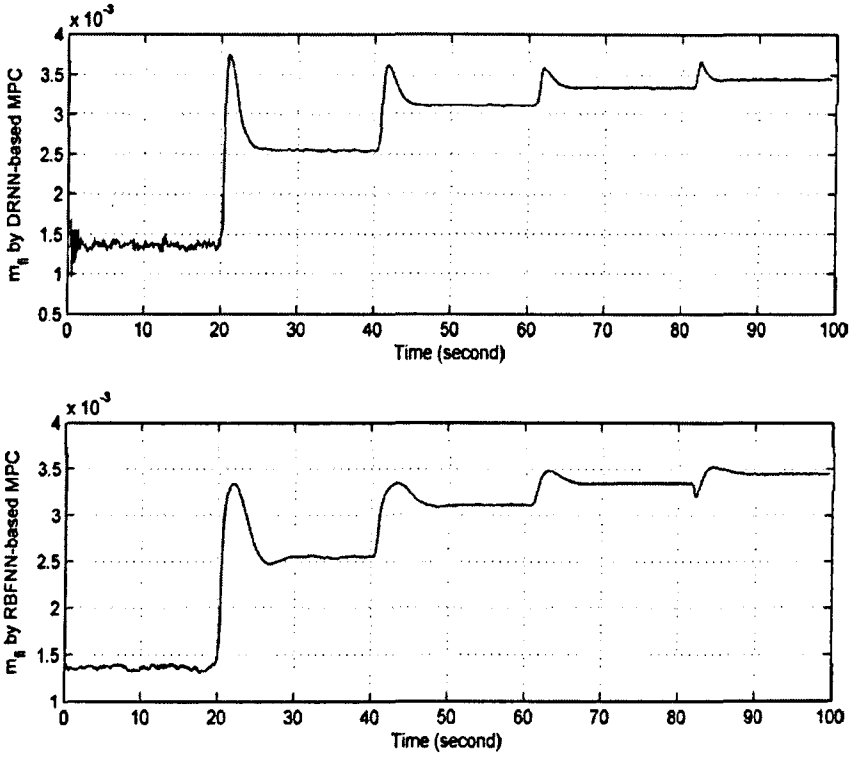

Figure 7.18: Comparison of $\dot{m}_{fi}$ Produced by DRNN based MPC and RBFNN based MPC

The significant improvement on control performance has been achieved by DRNN based MPC scheme if compared with PID control. FFC method described in Chapter 5 can obtain better control performance. However, DRNN-based MPC method reduces the system complexity in a large degree without losing satisfactory control on AFR. Finally, as expected at the beginning of this chapter, DRNN model is more suitable to be an internal model of MPC scheme if compared with RBFNN model.

Chapter 8

Conclusions and Future Work

8.1 Conclusions

Three control methods have been proposed in this thesis for air fuel ratio control of SI engines. In terms of control performance, these developed advanced control methods can obtain satisfactory effectiveness and robustness to large system uncertainty. All the aims and objectives listed in Chapter 1 of this thesis have been achieved and briefly concluded as below.

A new development in automotive engine AFR control with a feedforward-feedback control system structure has been made in this research. Two neural networks are used to estimate air flow rate into the cylinders and required fuel injection rate under variations of the throttle angle. The inverse of fuel injection dynamics is employed to obtain an accurate estimation of fuel injection rate. The developed system is evaluated with an engine benchmark of simulation throughout the whole operating space, and satisfactory performance is obtained. This method, inspired by look-up table, has two significant advantages over look-up table method. First, by the realizing input-output mapping with neural computation rather than data storing, the cost on system memory is reduced. Moreover, compared with the linear interpolation on control data used in look-up table method, a smoother

hyper-plane for control is achieved by using neural network model. In addition, a large amount of time and labor cost on control system calibration has been saved.

This research also proposed a RBFNN-base MPC scheme for air fuel ratio of SI engines. The adaptive RBF network is trained by the RLS method for modelling AFR dynamics of an SI engines. Based on the engine models, a MPC scheme is realized for controlling AFR. According to different MPC structures, both single dimension and multi-dimension optimization methods are implemented in the MPC scheme, and their advantages and disadvantages are discussed. The robustness of this model predictive control against air leakage in manifold is tested. The developed MPC scheme reduces the undesired overshoot of AFR response by predicting the future outputs and a good static and dynamic performance of the control system has been obtained. RBFNN model is adapted online by RLS algorithm, so that the change on engine parameters is compensated. In terms of control performance, RBFNN-based MPC for AFR is much better than a traditional PI controller in a wide range of engine operating condition.

Further research on MPC scheme on AFR regulation has been done by adopting recurrent neural network for modelling engine dynamics. An adaptive DRNN trained by the dynamic back-propagation method is used for modelling AFR dynamics of an SI engine. Due to the recurrent characteristic of DRNN, the implementation of DBP algorithm is complex and prone to make mistakes. This makes the tuning of neural parameters very difficult. Automatic differentiation technique is implemented in our application during training and online updating stages. The complexity of DBP algorithm has been reduced by the introduction of AD. In the meantime, the computational efficiency is improved . Based on this DRNN model, the MPC scheme is realized for regulating AFR. According to the comparison with RBFNN-based MPC methods, a better control performance in both steady state and transient has been obtained by using this DRNN as an in-

ternal model for MPC scheme on AFR. However, the advantage of a DRNN model is accompanied by an increase in complexity of the structure and computational load on training algorithm. This means that more powerful micro-controller is required if one wants to implement DRNN-based MPC for engine control system. The practice cost on the ECU with such control scheme is consequently higher.

8.2 Future Work

8.2.1 On-Board Application of Developed Control Algorithms

The effectiveness and robustness of control algorithms developed in this research have been investigated using the mean value engine model that is a widely used engine benchmark for engine modelling and control. However, before the practical application in automotive industry, the developed methods are necessary to tested on a real-time computer with real input-output devices. This process is usually called rapid control prototyping (RCP), which is shown in Fig. 8.1 [75].

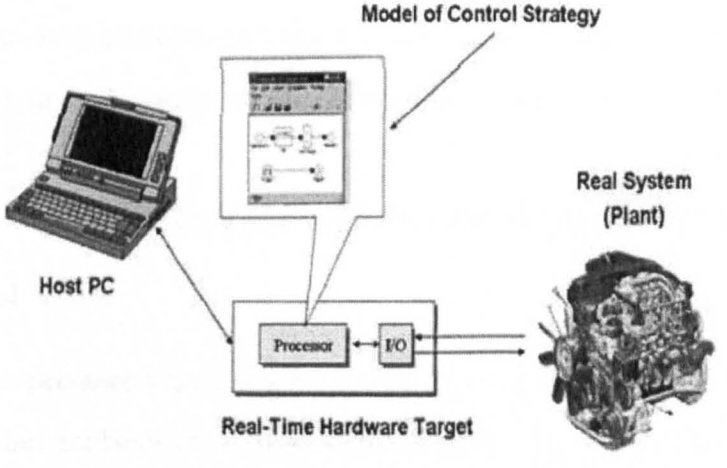

Figure 8.1: Rapid Control Prototyping Definition: control algorithm is simulated and plant is real

As the control algorithms we developed are all based on MVEM that runs in *MAT-*

LAB/SIMULINK computational environment, the RCP solution from *dSPACE* is the best choice to carry out further research work. The hardware and software required to complete the experimental setup are shown in Table 8.1 [76]. In addition, a high performance PC with *MATLAB/SIMULINK* installed and an engine test bed are required.

HARDWARE	
Single-board hardware	DS1103 PPC Controller Board
Accessories	PEX4 Expansion Box with PCMCIA Link CLP1103 Connector Panel
SOFTWARE	
Implementation software	Real-Time Interface (RTI) RTI CAN Blockset PowerPC Compiler
Test and Experiment Software	ControlDesk MLIB/MTRACE

Table 8.1: Hardwaer and Software Components for RCP

Given these research facilities, the developed control algorithms in *MATLAB* environment can be transferred from *M* files to *C* files. Then, the *C* files will be complied for a certain processor and exported to DS1103 PPC board for real-time engine control. DS1103 PPC board connects to PC with high-speed serial link by PX4 expansion box, therefore, all the real-time data of engine running condition can be display in the monitor of PC through the software *ControlDesk*.

8.2.2 In-Cylinder Pressure Re-Construction for AFR Control

An in-cylinder pressure sensor inside the combustion chamber can provide AFR information that replaces the oxygen sensor signal. By using such kind of sensors, the time delay in the feedback system can be reduced significantly [77]. The effort made on a look-up table controller could consequently be reduced by the improvement on control performance of feedback controller during fast transients. When considering production engine, cost and durability issues pose major prob-

lems. Several in-cylinder pressure re-construction methods have been proposed. Most of them are based on measured crank kinematics or measured structural vibrations. Linear models, frequency response functions and feedforward neural networks had been used in previous research [78] [79] [19]. Few research papers using recurrent neural networks have been found due to the difficulty on training and low computational efficiency for online updating. The diagonal recurrent neural network with AD tuning, which has been developed in this PhD project, can be the potential model for such application. In addition, the stability and robustness of AFR control system using NN based predictor need to be investigated for practical application.

8.2.3 Extension of NN-based Control for Other Engine Control Problems

The number of variables in an engine control system has risen from 40 to 600 in the last two decades [80]. However, more and more new technologies, such as active safety systems, hybrid powertrain, turbo-charged gasoline engines, and variable valve timing etc, have been used in automotive industry along with the growth of requirements in engine efficiency, customer safety, and vehicle ride comfort. The introduction of these new technologies makes the scale of control system huger and more complex.

For example, Figure 8.2 [81] shows a schematic diagram of a turbo-charged gasoline engine. A design challenge for such engines is to develop a system that provides adequate boost at low speed and load without creating an over-boost situation at high speed and loads. Typically, the solution is to control the wastegate very precisely according to the engine operating condition. This certainly causes an increase in complexity of the control design and calibration. The developed control schemes in this thesis can be easily extended to solve this kind control problem to meet strict standard. In addition, the development cost and

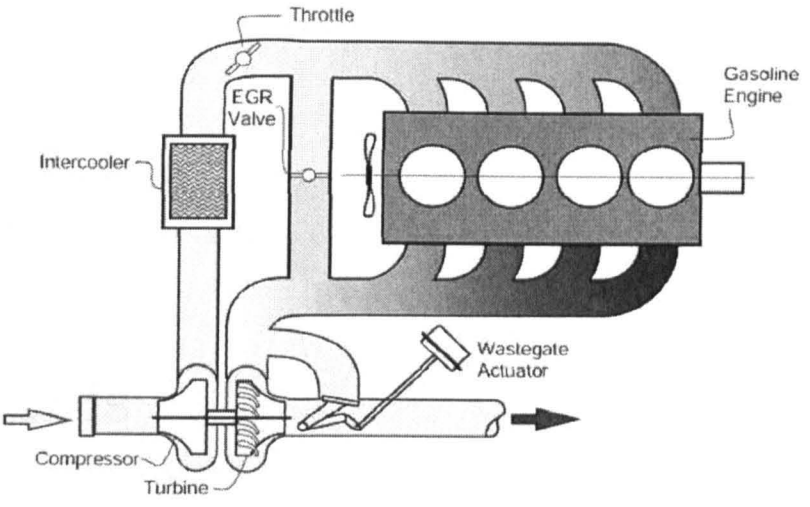

Figure 8.2: Schematic Diagram of a Turbo-Charged Gasoline Engine

system complexity will be reduced significantly by implementing NN-based methods and MPC on production ECU.

8.2.4 Neural Network for Engine Fault Diagnosis

On-board diagnosis of car engines has become increasingly important because of environmentally based legislative regulations such as on-board diagnostics-II (OBDII) [5]. A fault can be defined as an undesired deviation of one or more variables of the system from the normal behavior. For automotive engine, faults are often harmful to automobile and even lead to hazardous situation for vehicle driver if early detection is not made. On the other hand, early detection and isolation of engine faults can affect the fuel efficiency and reduced the air pollution. However, the fault detection based on first principals is often not effective due to the severe nonlinearity of engine dynamics. Recently, engine fault diagnosis using neural networks has been an active research area. RBFNN and DRNN have been proven good modelling tools in this PhD project, and further investigation of their effectiveness for engine fault diagnosis may produce satisfactory results.

Appendix A

Symbols

H_u	fuel lower heating value(kJ/kg)
I	crank shaft load inertia($kg \cdot m^2$)
L_{th}	stoichiometric air/fuel ratio 14.7
$\dot{m}_{EGR}$	EGR mass flow (kg/sec)
$\dot{m}_{ap}$	air mass flow into intake port (kg/sec)
$\dot{m}_f$	engine port fuel mass flow (kg/sec)
$\dot{m}_{ff}$	fuel film mass flow (kg/sec)
$\dot{m}_{fi}$	injected fuel mass flow (kg/sec)
$\dot{m}_{fv}$	fuel vapor mass flow (kg/sec)
N_1	minimum prediction horizon
N_2	maximum prediction horizon
N_u	control horizon
n	crankshaft speed (rpm)
P_a	ambient pressure (bar)
P_b	load power (kW)
P_f	friction power (kW)
P_i	manifold pressure (bar)
P_p	pumping power (kW)
R	gas constant (here 287×10^{-5})

T_a	ambient temperature ($Kelvin$)
T_{EGR}	EGR temperature ($Kelvin$)
T_i	intake manifold temperature ($Kelvin$)
t_d	time delay of fuel injection systems (sec)
u	throttle position ($degrees$)
V_d	engine displacement($liters$)
V_i	manifold + port passage volume (m^3)
λ	air/fuel ratio
$\Delta\tau_d$	injection torque delay time (sec)
κ	ratio of the specific heats = 1.4 for air

Appendix B

Abbreviations

AD	Automatic differentiation
ARX	Auto regressive with exogenous input
CCEM	Cylinder by cylinder engine model
COM	Control oriented model
CTRNN	Continuous time recurrent neural network
DBP	Dynamic back-propagation
DI	Direct injection
DRNN	Diagonal recurrent neural network
ECU	Electronic control unit
EGO	Exhaust gas oxygen sensor
EGR	Exhaust gas re-circulation
EMS	Electronic management system
FD	Finite difference
FPGA	Field programmable gate arrays
FRNN	Fully connected recurrent neural network
HC	Hydrocarbons
IC	Internal combustion
LS	Least square
MAE	Mean absolute error

MI	Manifold injection
MIMO	Multiple-input and multiple-output
MLP	Multiple-layer perceptron
MPC	Model predictive control
MVEM	Mean value engine model
NLP	Nonlinear programming
NN	Neural network
NO	Nitric oxide
NO_2	Nitrogen dioxide
ODE	Ordinary differential equation
PID	Proportional-integral-derivative
PLRBF	Pseudo-linear radial basis function
PM	Particulate matter
QP	Quadratic programming
RAS	Random amplitude signal
RBF	Radial basis function
RBFNN	Radial basis function neural network
RLS	Recursive least square
RNN	Recurrent neural network
SD	Symbolic differentiation
SI	Spark ignition
SISO	Single input single output
SQP	Sequential quadratic programming
TDC	Top dead centre
TWC	Three way catalytic
VVT	Variable valve timing

Appendix C

Author Publication List

Journal Paper:

[1] Yu-Jia Zhai, Ding-Li Yu, (2009), Neural network model-based automotive engine air/fuel ratio control and robustness evaluation, *Engineering Applications of Artificial Intelligence*, Volume 22, Issue 2, Pages 171-180

[2] Yu-Jia Zhai, Ding-Li, Yu, (2008), RBF-Based Feedforward-Feedback Control for Air-Fuel Ratio of SI Engines, *IMechE Journal of Automobile Engineering*, Volume 222, Issue D3, Pages 415C428.

[3] Yu-Jia. Zhai, Ding-Li Yu and Ke-Li. Wang, (2007), Comparison of single-dimensional and multi-dimensional optimization approaches in adaptive model predictive control for air-fuel ratio of SI engines, *International Journal of Information and Systems Sciences*, Volume 3, Number 1, Pages 129-149.

[4] Yu-Jia Zhai, Ding-Li, Yu, (2007), A Neural Network Model Based MPC of Engine AFR with Single-Dimensional Optimization, *Advances in Neural Networks*, Computer Science Springer, Volume 4491, Pages 339-348.

Conference Paper:

[1] Yu-Jia Zhai, Ding-Li, Yu, (2008), Automotive engine modeling with continuous time recurrent neural networks, *5th International Symposium on Automatic Control*, 18-19 September, Wismar, Germany.

[2] Yu-Jia, Ding-Li Yu, (2007), RBF-based Feedfoward-Feedback Control for Air-fuel Ratio of SI Engines, *Workshop on Advanced Fuzzy and Neural Control*, 29-30 October, Valenciennes, France.

[3] Yu-Jia Zhai, Ding-Li Yu, (2007). Advanced Neural Network-based Feedforward Control on Air Fuel Ratio of SI Engines, *13th Chinese Automation and Computing Society Conference*, 15 September, Staffordshire, England.

[4] Yu-Jia Zhai, Ding-Li Yu, (2007) A Fast Optimization Approach in Adaptive Model Predictive Control for Air-Fuel Ratio of SI Engines, *13th Chinese Automation and Computing Society Conference* 15 September, Staffordshire, England.

Submitted Journal Paper:

[1] Yu-Jia Zhai, Ding-Li Yu, Hong-Yu Guo, (2009), Robust Air/Fuel Ratio Control with Adaptive DRNN Model and AD Tuning , *Engineering Applications of Artificial Intelligence*

Bibliography

[1] Julia H. Buckland. Automotive emissions control. In *American Control Conference*, volume 5, pages 3290 – 3295, Ford Motor, Dearborn, MI, USA, June 2005.

[2] John B. Heywood. *Internal Combustion Engine Fundamentals.* McGRAW-HILL, 1988.

[3] T. K. Garrett, K. Newton, and W. Steeds. *The Motor Vehicle.* Butterworth-Heinemann, 2001.

[4] Lino Guzzella and Christopher Onder. *Introduction to Modelling and Control of Internal Combustion Engine Systems.* Springer, 2004.

[5] Uwe Kiencke and Lars Nielsen. *Automotive Control Systems for Engine, Driveline, and Vehicle.* Springer, 2005.

[6] Victor Hillier and Peter Coombes. *Fundamentals of Motor Vehicle Technology.* Nelson Thornes Ltd, 2004.

[7] Allan Bonnick. *Automotive Computer Controlled Systems Diagnostic Tools and Techniques.* Butterworth-Heinemann, 2001.

[8] Horst Bauer, editor. *Automotive Handbook.* Robort Bosch GmbH, 5 edition, 2000.

[9] Jonas Karlsson and Jonas Fredriksson. Cylinder-by-cylinder engine models vs mean value engine models for use in powertrain control applications. Technical Report 99P-174, SAE Technical Paper, 1998.

[10] E. Hendricks, D. Engler, and M. Fam. A generic mean value engine model for spark ignition engines. In *Proceedings of 41st Simulation Conference*, Denmark, 2000. DTU Lyngby.

[11] Elbert Hendricks and S. C. Sorenson. Mean value modelling for spark ignition engines. Technical Report 900616, Transactions of SAE, 1990.

[12] C. Manzie, M. Palaniswami, and H. Watson. Gaussian networks for fuel injection control. *Proceedings of the Institution of Mechanical Engineers, Journal of Automobile Engineering*, 215(D10):1053–1068, 2001.

[13] C. Manzie, M. Palaniswami, D. Ralph, H. Watson, and X. Yi. Model predictive control of a fuel injection system with a radial basis function network observer. *Journal of Dynamic Systems Measurement and Control Transactions of the ASME*, 124(4):648–658, 2002.

[14] Robort Bosch GmbH(Hrsg.). *Ottomotor-Management: System und Komponenten.* Friedr Vieweg Sohn Verlagsaesellschaft mbH, 2 edition, 2003.

[15] William Kimberley, editor. *Automotive Handbook.* Professional Engineering Publishing, 2004.

[16] B. Boning, W. Richter, W. Tuleweit, and B. Zeilinger. *Sytematische Optimierung von Kennfeldern fur Gemischbildung und Zundung bei Ottomotoren*, volume 2 of *44*. MotorTechnische Zeitschrift(MTZ), 1983.

[17] Jan Marian Maciejowski. *Predictive Control with Constraints.* Person Education Limited, 2001.

[18] Lennart Ljung. *System Identification Theory for The User.* Prentice-Hall, Inc., 1999.

[19] R. Potenza, J. F. Dunne, S. Vulli, D. Richardson, and P. King. Multicylinder engine pressure re-construction using narx neural networks and crank kinematics. *International Journal of Engine Research*, 8:499–518, 2007.

[20] G. De Nicolao, R. Scattolini, and C. Siviero. Modelling the volumetric efficiency of IC engines: Parametric, non-parametric and neural techniques. *Control Eng. Practice*, 4(10):1405–1415, 1996.

[21] T. M. Winsel, M. Ayeb, D. Lichtenhaler, and Theuerkauf. A neural estimator for cylinder pressure and engine torque. *SAE Technical*, (1999-01-1165), 1999.

[22] F. Heister and M. Froehlich. Nonlinear time series analysis of combustion pressure data for neural network training with the concept of mutual information. *Journal of Automobile Engineering*, 215:299–304, 2001.

[23] Tomas Ploni, Boris Rohal-Ilkiv, and Tor Arne Johansen. Multiple ARX model-based air fuel ratio predictive control for SI engines. volume 3 of *1*, University of Valenciennes et du Hainaut Cambrsis, France, 10 2007. IFAC.

[24] Paul Frank. Application of fuzzy logic to process supervision and fault diagnosis. In *Fault Detection, Supervison and Safty for Technical Processes*, pages 507–514, Espoo, Finland, 1994. IFAC.

[25] A. Soliman, G. Rizzoni, and Y. W. Kim. Diagnosis of an automotive emission control system using fuzzy inference. *Control Engineering Practice*, 7:209–216, 1999.

[26] S. Boverieb, B. Demayab, J. M. Le Quellecb, and A. Titlic. Contribution of fuzzynext term logic control to the improvement of modern car performances. *Control Engineering Practice*, 1(2):291–297, 1993.

[27] Yonghong Tan and Mehrdad Saif. Neural-networks-based nonlinear dynamic modeling for automotive engines. *Neurocomputing*, 30:129–142, 2000.

[28] J. A. F. Vinsonneau, D. N. Shields, P.J. King, and K. J. Burnham. Polynomial and neural network spark ignition engine intake manifold modeling. volume 2, pages 718–723. Conference on Systems Engineering, ICSE, 2003.

[29] R. Isermann and N. Muller. Modelling and adaptive control of combustion engines with fast neural networks. In *Hybrid System and Their Implementation on Smart Adaptive System*, Tenerife, Spain, 2001. European Symposium on Intelligent Technologies.

[30] I. Arsie, C. Pianese, and M. Sorrentino. A procedure to enhance identification of recurrent neural networks for simulating air fuel ratio dynamics in SI engines. *Engineering Application of Artificial Intelligence*, 19:65–77, 2006.

[31] J. N. Cercos and J. Q. Casin. Air fuel ratio control of a gasoline engine by means of a recurrent neural network with differentiated input. *Integrated Computer-Aided Engineering*, 8:243–255, 2001.

[32] R.K. Al Seyab and Y. Cao. Nonlinear system identification for predictive control using continuous time recurrent neural networks and automatic differentiation. *Journal of Process Control*, 18:568–581, 2008.

[33] Ventura Assuncao. Neuro-hybrid models for automotive system identification. *Proceedings of World Academy of Science, Engineering, and Technology*, 12:94–97, March 2006.

[34] Mohmad Marouf Wani and M. Arif Wani. Hybrid neural network based model for predicting the performance of a two stroke spark ignition engine. In *ICMLA '07: Proceedings of the Sixth International Conference on Machine Learning and Applications*, pages 470–475, Washington, DC, USA, 2007. IEEE Computer Society.

[35] S. Pischinger, C. Schernus, and G. Lutkemeyer. Model supported SI engine cold start calibration. *Meeting on Open Loop and Closed Loop Control of Vehicles and Engines*, pages 217–227, 2004.

[36] C. H. Onder and H. P. Geering. Model-based multivariable speed and air-to-fuel ratio control of an SI engine. Technical Report 930859, SAE Technical Paper, 1993.

[37] Maria Karlsson, Kent Ekholm, Petter Strandh, Rolf Johansson, and Per Tunestl. Lqg control for minimization of emissions in a diesel engine. Detroit, MI, USA, 2008. IEEE International Conference on Control Applications.

[38] A. I. Bhatti, S. K. Spurgeon, R. Dorey, and C. Edwards. Sliding mode configurations for automotive engine control. *International Journal of Adaptive Control and Signal Processing*, 13:49–69, 1999.

[39] P. Kaidantzis, P. Rasmussen, M. Jensen, T. Vesterholm, and E. Hendricks. Advanced nonlinear observer control of si engines. Technical Report 930768, Transactions of SAE, 1993.

[40] Shiwei Wang. *Advanced Air Fuel Ratio Control of Automotive* SI *Engines.* PhD thesis, Liverpool John Moores University, 2006.

[41] S. B. Choi and J. K. Hendricks. An observer-based controller design method for improving air-fuel characteristics of spark ignition engines. *IEEE Transactions on Control Systems Technology*, 6(3):325–334, 1998.

[42] Shivaram S. Kamat, Hossein Javaherian, Vivek V. Diwanji, Jessy G. Smith, and K. P. Madhavan. Virtual air-fuel ratio sensors for engine control and diagnostics. In *Proceedings of the 2006 American Control Conference*, Minneapolis, Minnesota, USA,, June 14-16 2006.

[43] Jr. L. I. Davis, G. V. Puskorius, F. Yuan, and L. Feldkamp. Neural network modelling and control of an anti-lock brake system. *Proceeding of Intelligent Vehicles 92*, pages 179–184, 1992.

[44] G. V. Puskorius and L. Feldkamp. Automotive engine idle speed control with recurrent neural networks. *Proceeding of American Control Conference*, pages 311–316, 1993.

[45] E. F. Camacho and C. Bordons. *Model Predictive Control.* Springer Verlag, 2000.

[46] Paolo Linoa, Bruno Maionea, and Claudio Amorese. Modelling and predictive control of a new injection system for compressed natural gas engines. *Control Engineering Practice*, (16):1216C1230, 2008.

[47] T. Coen, J. Anthonis, and J. De Baerdemaeker. Cruise control using model predictive control with constraints. *Computers and Electronics in Agriculture*, (63):227C236, 2008.

[48] Hans Joachim Ferreau, Peter Ortner, Peter Langthaler, Luigi del Re, and Moritz Diehl. Predictive control of a real-world diesel engine using an extended online active set strategy. *Annual Reviews in Control*, (31):293C301, 2007.

[49] Shiwei. Wang, Dingli Yu, J.B. Gomm, G.F. Page, and S.S. Douglas. Adaptive neural network model based predictive control for air-fuel ratio of SI engine. *Engineering Application of Artificial Intelligence*, 19:189–200, 2006.

[50] Oliver Nelles. *Nonlinear System Identification.* Springer, 2001.

[51] Zhi Tian, Kristine L. Bell, and Harry L. Van Trees. A recursive least squares implementation for lcmp beamforming under quadratic constraint. *IEEE Transactions on Signal Processing*, 49(6), 2001.

[52] C. C. Ku and K. Y. Lee. Diagonal recurrent neural networks for dynamic system control. *IEEE Transactions on Neural Networks*, 6(1):144–156, 1995.

[53] R. Fletcher. *Practical Methods of Optimization.* Wiley, 2nd edition, 2000.

[54] B. W. Char and K. O. Geddes. *MAPLE Reference Manual.* Watcome Publications, 1988.

[55] Andreas Griewank. ODE solving via automatic differentiation and rational prediction. Technical report, Technical University Dresden, Germany, 1995.

[56] A. Verma. An introduction to automatic differentiation. *Current Science*, 78:804–807, 2000.

[57] L. B. Rall. Automatic differentiation: Techniques and applications. *Notes in Computer Science, SpringerCVerlag*, 120, 1981.

[58] M. C. Castro, R. C. Vieira, and E. C. Biscaia. Automatic differentiation tools in the dynamic simulation of chemical engineering processes. *Journal of Chemical Engineering*, 17(4), 2000.

[59] A. Griewank, J. David, and U. Jean. ADOL-C: A package for the automatic differentiation of algorithms written in C/C++. *ACM*, 22(2):131–167, 1996.

[60] A. Griewank. Achieving logarithmic growth of temporal and spatial complexity in reverse automatic differentiation. *Optimization Methods and Software*, 1:35–54, 1992.

[61] A. Griewank. Evaluating derivatives. *SIAM*, 2000.

[62] Rihab Khalid Al Seyab. *Nonlinear Model Predictive Control Using Automatic Differentiation.* PhD thesis, Cranfield University, 2006.

[63] G. Lightbody and G. W. Irwin. Nonlinear control structures based on embedded neural system models. *IEEE Transactions on Neural Networks*, 8(3):553–567, 1997.

[64] T.H. Lee, S.N. Huang, K.Z. Tang, K.K. Tan, and A. Al Mamun. PID control incorporating RBF-neural network for servo mechanical systems. In *The 29th Annual Conference of the IEEE*, volume 3, pages 2789– 2793. Industrial Electronics Society, Nov 2003.

[65] G. M. Scott, J. W. Shavlik, and W. H. Ray. Refining PID controllers using neural networks. Technical Report Machine Learning Research Group Working Paper 91-3, Madison, WI, 1991.

[66] Gene F. Franklin, David Powell, and Michael L. Workman. *Digital Control of Dynamic Systems.* Addison Wesley, 3rd edition, 2001.

[67] Dale E. Seborg, Thomas F. Edgar, and Duncan A. Mellichamp. *Process Dynamics and Control.* John Wiley and Sons New York, 1989.

[68] Mooncheol Won, Seibum B. Choi, and J. K. Hedrick. Air-to-fuel ratio control of spark ignition engines using gaussian network sliding control. *IEEE Transactions on Control Systems Technology*, 6(5), September 1998.

[69] A. Draeger, S. Engell, and H. Ranke. Model predictive control using neural networks. *IEEE Control Systems Magazine*, 15(5):61–66, 1995.

[70] Singiresu S. Rao. *Engineering Optimization.* John Wiley Sons, Inc., 1996.

[71] L. T. Biegler, J. Nocedal, and C. Schmid. A reduced hessian method for large scale constrained optimization. *SIAM Journal of Optimization*, 5:314–347, 1995.

[72] L. T. Biegler, J. Nocedal, C. Schmid, and D. Ternet. Numerical experience with a reduced hessian method for large scale constrained optimization. *Computational Optimization and Applications*, 15(1):45–67, 2000.

[73] S. Mohamad and K. Gopalsamy. Exponential stability of continuous-time and discrete-time cellular neural networks with delays. *Applied Mathematics and Computation*, 135(1):17–38, Feb 2003.

[74] Edouard Leclercq, Fabrice Druaux, Dimitri Lefebvre, and Salem Zerkaoui. Autonomous learning algorithm for fully connected recurrent networks. *Neurocomputing*, 63:25–44, 2005.

[75] Frank W. Liou. *Rapid Prototyping and Engineering Applications, a Toolbox for Prototype Development.* Taylor and Francis(CRC Press), 2007.

[76] Herbert Hanselmann. *Solution for Control.* dSPACE Ltd UK, 2006.

[77] Per Tunestal and J. Karl Hedrick. Cylinder air fuel ratio estimation using net heat release data. *Control Engineering Practice*, 11:311–318, 2001.

[78] R. H. Lyon. Vibration based diagnostics of machine transients. *Journal of Sound and Vibration*, pages 18–22, 1988.

[79] P. Azzoni, G. Cantoni, G. Minelli, and E. Padovani. Cylinder pressure signal recovering from vibration signal. *Proceedings of the 20th International Symposium on Automotive Technology and Automation*, 1989.

[80] Michael Beham. *Modelling, Calibration and Control of VVT SI Engines Using Neural Networks.* PhD thesis, Liverpool John Moores University, 2004.

[81] N. Watson and M. S. Janota. *Turbocharging the Internal Combustion Engine.* Wiley-Interscience, 1982.