

THE UNIVERSITY *of* LIVERPOOL  
Department of Electrical Engineering and Electronics

# Advanced Local Predictors for Short Term Electric Load Forecasting

Thesis submitted in accordance with the  
requirements of the University of Liverpool  
for the degree of Doctor in Philosophy

by

**Ehab Elsayed Aly Elattar**

June 2010

# Advanced Local Predictors for Short Term Electric Load Forecasting

by

**Ehab Elsayed Aly Elattar**

Copyright 2010

To my father *Elsayed Elatter* and my *late mother*

## Acknowledgements

First of all, I would like to express my deep gratitude and appreciation to my supervisors Prof. Q. H. Wu and Dr. J. Y. Goulermas for their continuous guidance, support, precious help, advice, encouragement, inspiration and enthusiasm from the first day and throughout the time that I have been in Liverpool. I would like to thank Dr. K. Lau for his helpful discussions for the work presented in Chapter 4.

I am very much indebted and grateful to the Egyptian Government for providing me with the opportunity for this research, and for its financial support and encouragement.

I would like to offer my thanks to the Intelligence Engineering and Automation Group for their kind help during my studies. Moreover, I am grateful to the Department of Electrical Engineering and Electronics, The University of Liverpool, for providing the research facilities, making it possible for me to conduct this research.

Special thanks are due to my supportive and infinitely patient wife who has continually flooded me with her support during the compilation of this work. I would like also to thank my kids, Ahmed and Nada, who provided a much needed diversion which ensured that I always had something else to think about. Moreover, I can not forget my father to whom I am very much indebted and grateful. He is the main reason beyond all my achievements. Thanks are also due to the rest of my family for their interest and support.

Before all and after all, praise to Allah from Him we gain knowledge.

# Abstract

Operation decisions in power systems, such as unit commitment, reducing spinning reserve, economic dispatch, reliability analysis and maintenance scheduling depend on the future behaviour of loads. Therefore, accurate load forecasting helps the electric utility dispatcher to make these operation decisions properly. This makes short term load forecasting (STLF) an essential part of an efficient power system operation.

The research work in the STLF area is still a challenge to the electrical engineering scholars, in spite of the numerous literatures on STLF. This is due to high complexity of loads and deregulation. To solve the STLF problem, a new approach is presented in this research work by combining state-of-the-art predictors, such as radial basis function (RBF) networks, support vector regression (SVR) and Gaussian processes (GP), with a local prediction model. The proposed method employs powerful phase reconstruction algorithms, such as the correlation dimension and mutual information methods used in time series analysis for data preprocessing. It makes use of relevant information, so that only very similar historical data patterns in the reconstructed phase space are used to train the forecasting algorithms. Therefore, a new systematic method is proposed to calculate the best number of these historical data patterns. The performance is evaluated with real world datasets and compared with the standard global algorithms and two benchmarks through multiple experiments, load types and prediction steps. It is demonstrated that the introduced architecture significantly enhances the accuracy of global forecasting.

In the local predictors, all training data contribute to the accuracy of the model to the same extent. However it is common that some training points are more important than others in the load forecasting problem. Therefore, the

locally weighted support vector regression (LWSVR) method is proposed in this research work by modifying the risk function of the support vector regression algorithm with use of locally weighted regression. LWSVR gives a weight factor to each train load datum according to its distance to point under prediction. Furthermore, the weighted distance algorithm is proposed to optimise the weighting function's bandwidth. Two real world datasets have been used for evaluated the LWSVR method. It is demonstrated that LWSVR method significantly enhances the forecasting accuracy of other published methods.

To avoid the limitations of SVR and in order to follow the latest developments to have a modern system, a new approach based on group method of data handling (GMDH) for solving the load forecasting problem is proposed. The proposed method, locally weighted GMDH (LWGMDH), is derived by combining the GMDH with local regression method and weighted least squares regression. In the proposed method, the phase space is reconstructed based on multivariate time series using kernel principal component analysis (KPCA) method. Therefore, the drawbacks of traditional reconstruction techniques can be overcome. Two different real world datasets have been used to demonstrate the LWGMDH method. The numerical results show the superiority of the LWGMDH method over other methods.

Due to the limited generic structure of GMDH network (quadratic two-variable polynomial), it tends to produce an exceedingly complex network when it comes to highly nonlinear systems. In order to overcome this, the generalised LWGMDH based evolutionary algorithm is proposed in which every node in the network has different number of inputs and different polynomial order. In addition, the connectivity configuration in the proposed method is not limited to adjacent layers unlike GMDH. In order to design the whole architecture of the generalised LWGMDH network, the evolutionary algorithm is used where a new encoding scheme is proposed. The new method is examined using two different real world datasets and gives better performance compared with other methods.

# Contents

List of Figures	x
List of Tables	xiv
List of Abbreviations	xvi
<b>1 Introduction</b>	<b>1</b>
1.1 Importance of short term load forecasting . . . . .	3
1.2 Objectives . . . . .	4
1.3 Thesis outline . . . . .	5
1.4 Major contributions . . . . .	6
1.5 Publications . . . . .	8
<b>2 Short Term Load Forecasting Literature Survey</b>	<b>10</b>
2.1 Factors affecting the power system load . . . . .	10
2.2 Classification of developed STLF methods . . . . .	13
2.2.1 Regression methods . . . . .	14
2.2.2 Time series methods . . . . .	15
2.2.3 Neural networks . . . . .	18
2.2.4 Expert systems . . . . .	22
2.2.5 Integration of different algorithms . . . . .	25
<b>3 Mathematical Background</b>	<b>27</b>
3.1 Linear regression . . . . .	27
3.2 Introduction to nonlinear regression . . . . .	29
3.2.1 Curse of dimensionality . . . . .	30
3.3 Radial basis function (RBF) networks . . . . .	31
3.3.1 Radial basis function structure . . . . .	32
3.3.2 Basis functions . . . . .	33
3.3.3 Radial basis function training . . . . .	34
3.4 Support vector regression (SVR) . . . . .	37
3.4.1 The basic idea of SVR . . . . .	37
3.4.2 Primal and dual optimisation . . . . .	39
3.4.3 Nonlinear SVR . . . . .	41

3.4.4	Kernel function . . . . .	43
3.4.5	SVR optimisation . . . . .	45
3.5	Gaussian process (GP) . . . . .	48
3.5.1	Definition of Gaussian process . . . . .	48
3.5.2	Predicting with Gaussian process . . . . .	49
3.5.3	Covariance function . . . . .	51
3.6	Conclusions . . . . .	51
<b>4</b>	<b>Local Prediction Method for Short Term Load Forecasting</b>	<b>53</b>
4.1	Introduction . . . . .	53
4.2	Phase space reconstruction of time series based on CD method . .	54
4.2.1	Estimate the embedding dimension . . . . .	55
4.2.2	Estimate the time delay constant . . . . .	56
4.3	Local prediction method . . . . .	57
4.4	Experimental results . . . . .	61
4.4.1	Datasets . . . . .	61
4.4.2	Parameters . . . . .	63
4.4.3	Results . . . . .	66
4.5	Conclusions . . . . .	80
<b>5</b>	<b>Short Term Load Forecasting Based on Locally Weighted Support Vector Regression</b>	<b>81</b>
5.1	Introduction . . . . .	81
5.2	Phase space reconstruction of multivariate time series . . . . .	82
5.3	Locally weighted regression (LWR) . . . . .	83
5.3.1	Weighting functions . . . . .	84
5.4	Locally weighted support vector regression (LWSVR) . . . . .	86
5.4.1	LWSVR conceptual interpretation . . . . .	86
5.5	Weighted distance algorithm for optimising the bandwidth . . . .	87
5.6	Case studies . . . . .	91
5.6.1	Case 1: EUNITE competition dataset . . . . .	93
5.6.2	Case 2: North American electric utility dataset . . . . .	96
5.7	Conclusions . . . . .	102
<b>6</b>	<b>Locally Weighted Group Method of Data Handling Based KPCA for STLF</b>	<b>103</b>
6.1	Introduction . . . . .	103
6.2	Time series reconstruction based on KPCA . . . . .	105
6.2.1	Introduction to KPCA . . . . .	105
6.2.2	KPCA implementation . . . . .	105
6.3	Group method of data handling (GMDH) . . . . .	108
6.3.1	GMDH layers . . . . .	109
6.3.2	GMDH nodes . . . . .	109
6.3.3	GMDH connections . . . . .	110
6.3.4	GMDH network and its advantages . . . . .	110

6.4	Locally weighted group method of data handling (LWGMDH) . . .	112
6.5	Short term load forecasting results . . . . .	114
6.5.1	Forecasting results using North American electric utility dataset . . . . .	117
6.5.2	Forecasting results using Victoria dataset . . . . .	122
6.6	Conclusions . . . . .	127
<b>7</b>	<b>Evolutionary Design of the Generalised Locally Weighted GMDH for STLTF</b> . . . . .	<b>128</b>
7.1	Introduction . . . . .	128
7.2	Evolutionary algorithm (EA) . . . . .	130
7.2.1	Representation and evaluation . . . . .	130
7.2.2	Selection . . . . .	132
7.2.3	Crossover . . . . .	133
7.2.4	Mutation . . . . .	134
7.2.5	Termination . . . . .	135
7.3	Generalised LWGMDH (G-LWGMDH) algorithm based EA . . .	135
7.3.1	Representation of chromosome for appropriate information of G-LWGMDH network . . . . .	136
7.3.2	Fitness evaluation . . . . .	136
7.3.3	EA operators for G-LWGMDH network reproduction . . .	139
7.4	Numerical results . . . . .	144
7.4.1	New York dataset . . . . .	146
7.4.2	Victoria dataset . . . . .	154
7.5	Conclusions . . . . .	160
<b>8</b>	<b>Conclusions</b> . . . . .	<b>161</b>
8.1	Summary . . . . .	161
8.2	Suggestions for future work . . . . .	164
	<b>Bibliography</b> . . . . .	<b>166</b>

# List of Figures

1.1	A general input-output arrangement of a STLF system. . . . .	2
2.1	Typical seasonal workdays of England and Wales load profiles of the year 2005. . . . .	11
2.2	Typical load curve of England and Wales for few weeks in year 2005. . . . .	12
2.3	Organisation of the expert systems. . . . .	23
2.4	The functional integration of expert systems components. . . . .	24
3.1	Architecture of a radial basis function network. . . . .	33
3.2	The $\varepsilon$ -insensitive loss function. . . . .	39
3.3	Transformation process illustration of a SVR model. A nonlinear mapping function $\phi(x)$ defined to mapping a nonlinear problem in two dimensional input space (a) to linear problem in two dimensional feature space (b). . . . .	42
3.4	The $\varepsilon$ -insensitive loss function for a nonlinear regression function. The solid line is the approximation function and the dashed line is the contour of the margin. The points lying on or outside the $\varepsilon$ tube are support vectors. . . . .	42
4.1	Flowchart of the local predictor. . . . .	59
4.2	Half-hourly electricity demand in England-Wales from 1 <sup>st</sup> July 2005 to 31 <sup>st</sup> July 2005 for summer period. . . . .	63
4.3	Half-hourly electricity demand in England-Wales from 26 <sup>th</sup> December 2005 to 25 <sup>th</sup> January 2006 for winter period. . . . .	63
4.4	Correlation dimension for the England-Wales dataset. . . . .	65
4.5	Correlation dimension for the France dataset. . . . .	65
4.6	Correlation dimension for the Greece dataset. . . . .	66
4.7	Comparison among all methods using MAPE for the England-Wales dataset. . . . .	70
4.8	Comparison among all methods using NMSE for the England-Wales dataset. . . . .	70
4.9	Comparison among all methods using MAPE for the France dataset. . . . .	71
4.10	Comparison among all methods using NMSE for the France dataset. . . . .	71

4.11	Comparison among all methods using MAPE for the Greece dataset.	72
4.12	Comparison among all methods using NMSE for the Greece dataset.	72
4.13	England-Wales' one day ahead forecasted and actual load of (a) 22/08/2005, (b) 27/08/2005, (c) 27/02/2006 and (d) 04/03/2006 using local GP method. . . . .	74
4.14	France's one day ahead forecasted and actual load of (a) 22/08/2005, (b) 27/08/2005, (c) 27/02/2006 and (d) 04/03/2006 using local SVR method. . . . .	75
4.15	Greece's one day ahead forecasted and actual load of (a) 22/08/2005, (b) 27/08/2005, (c) 27/02/2006 and (d) 04/03/2006 using local GP method. . . . .	75
4.16	Mean MAPE plotted against lead time for the three datasets for both load types. . . . .	76
4.17	Mean MAPE plotted against lead time for the three datasets for both load types for the first few hours. . . . .	77
4.18	Average prediction MAPE of every day of the week for the three datasets for both load types. . . . .	79
5.1	LWSVR conceptual interpretation. . . . .	87
5.2	Flowchart of the LWSVR algorithm. . . . .	90
5.3	Hourly electricity demand for the EUNITE dataset from (a) 1 <sup>st</sup> January 1998 to 31 <sup>st</sup> January 1998 and (b) 1 <sup>st</sup> July 1998 to 31 <sup>st</sup> July 1998. . . . .	92
5.4	Comparison of LWSVR method and other methods using MAPE for the dataset of EUNITE competition. . . . .	94
5.5	Comparison of LWSVR method and other methods using NMSE for the dataset of EUNITE competition. . . . .	95
5.6	Forecasted and actual maximum daily load in January 1999. . . . .	95
5.7	Comparison of LWSVR method and other methods using MAPE for the dataset of North American electric utility. . . . .	99
5.8	Comparison of LWSVR method and other methods using NMSE for the dataset of North American electric utility. . . . .	99
5.9	Average prediction MAPE of every day of the week during the testing period using the dataset of North American electric utility.	100
5.10	One day ahead forecasted and actual load from 27 <sup>th</sup> November 1990 to 3 <sup>rd</sup> December 1990. . . . .	102
6.1	Illustration of the basic idea of KPCA: (a) Two dimensional input space, displaying a set of data points, and (b) Two dimensional feature space, displaying the induced images of the data points congregating around a principle eigenvector. . . . .	106
6.2	GMDH network. . . . .	109
6.3	Stopping criterion. . . . .	111
6.4	Flowchart of the LWGMDH method. . . . .	115

6.5	Hourly electricity demand in Victoria city from (a) June 2003 (b) February 2003. . . . .	116
6.6	Average prediction MAPE of every day of the week during the whole testing period. . . . .	119
6.7	Average prediction MAPE of each method at A.M. peak period during the whole testing period. . . . .	121
6.8	Average prediction MAPE of each method at P.M. peak period during the whole testing period. . . . .	121
6.9	MAPE plotted against lead time for each method during the period from Monday, 1 <sup>st</sup> September 2003 to Sunday, 7 <sup>th</sup> September 2003. . . . .	124
6.10	Average prediction MAPE of every day of the week during July 2003. . . . .	125
6.11	Average prediction MAPE of every day of the week during December 2003. . . . .	125
7.1	The main flowchart of EA. . . . .	131
7.2	Structure of the chromosome which represents a G-LWGMDH network. . . . .	137
7.3	G-LWGMDH network. . . . .	137
7.4	Crossover operation for two individuals in G-LWGMDH. . . . .	139
7.5	Crossover operation on two G-LWGMDH networks. . . . .	140
7.6	Mutation operation for an individual in G-LWGMDH. . . . .	140
7.7	Flowchart of the G-LWGMDH based EA method. . . . .	143
7.8	Hourly electricity demand in New York city from 1 <sup>st</sup> June 2003 to 31 <sup>st</sup> December 2003. . . . .	144
7.9	Comparison of the G-LWGMDH based EA method and other methods using MAPE and NMSE for winter month. . . . .	147
7.10	Comparison of the G-LWGMDH based EA method and other methods using MAPE and NMSE for spring month. . . . .	149
7.11	Comparison of the G-LWGMDH based EA method and other methods using MAPE and NMSE for summer month. . . . .	149
7.12	Comparison of the G-LWGMDH based EA method and other methods using MAPE and NMSE for autumn month. . . . .	150
7.13	One day ahead forecasted and actual load from 5 <sup>th</sup> January 2004 to 11 <sup>th</sup> January 2004 using the New York dataset. . . . .	151
7.14	One day ahead forecasted and actual load from 5 <sup>th</sup> April 2004 to 11 <sup>th</sup> April 2004 using the New York dataset. . . . .	153
7.15	One day ahead forecasted and actual load from 5 <sup>th</sup> July 2004 to 11 <sup>th</sup> July 2004 using the New York dataset. . . . .	153
7.16	One day ahead forecasted and actual load from 4 <sup>th</sup> October 2004 to 10 <sup>th</sup> October 2004 using the New York dataset. . . . .	154
7.17	MAPE plotted against lead time for each method during the period from Monday, 1 <sup>st</sup> September 2003 to Sunday, 7 <sup>th</sup> September 2003 for the Victoria dataset. . . . .	155

7.18	Average prediction MAPE of every day of the week during the whole testing period for the Victoria dataset. . . . .	157
7.19	One day ahead forecasted and actual load from 7 <sup>th</sup> April 2003 to 13 <sup>th</sup> April 2003 using the Victoria dataset. . . . .	158
7.20	One day ahead forecasted and actual load from 7 <sup>th</sup> July 2003 to 13 <sup>th</sup> July 2003 using the Victoria dataset. . . . .	158
7.21	One day ahead forecasted and actual load from 6 <sup>th</sup> October 2003 to 13 <sup>th</sup> October 2003 using the Victoria dataset. . . . .	159
7.22	One day ahead forecasted and actual load from 1 <sup>st</sup> December 2003 to 7 <sup>th</sup> December 2003 using the Victoria dataset. . . . .	159

# List of Tables

4.1	Load data used for load forecasting . . . . .	62
4.2	Phase reconstruction parameters for each dataset . . . . .	64
4.3	The parameters used for each dataset . . . . .	67
4.4	Error of the entire testing period for the England-Wales dataset .	68
4.5	Error of the entire testing period for the France dataset . . . . .	69
4.6	Error of the entire testing period for the Greece dataset . . . . .	69
4.7	MAPE of each day during testing period of the three datasets of both load types . . . . .	78
4.8	Improvement of the local RBF method over other methods . . . . .	79
4.9	Improvement of the local SVR method over other methods . . . . .	79
4.10	Improvement of the local GP method over other methods . . . . .	80
5.1	Phase reconstruction parameters for each dataset . . . . .	93
5.2	Comparison of the LWSVR method and other methods using the dataset of EUNITE competition . . . . .	94
5.3	Comparison of the LWSVR method and other methods using the dataset of EUNITE competition (MAPE) . . . . .	97
5.4	Improvement of LWSVR over other methods using the dataset of EUNITE competition . . . . .	98
5.5	Comparison of LWSVR method and other methods using the dataset of North American electric utility . . . . .	98
5.6	Comparison of the LWSVR method and other methods using the dataset of North American electric utility (MAPE of the 24 Hours)	101
6.1	Phase reconstruction parameters for each dataset . . . . .	117
6.2	Comparison of LWGMDH method and other methods using the dataset of North American electric utility . . . . .	118
6.3	Comparison of the LWGMDH method and other methods using the dataset of the North American electric utility (MAPE of the 24 Hours) . . . . .	120
6.4	Comparison among methods using the Victoria dataset (MAPE) .	123
6.5	Overall mean performance of each method for each testing period using the Victoria dataset . . . . .	126

6.6	MAPE improvements of LWGMDH over other methods using the Victoria dataset . . . . .	126
7.1	Different types of polynomials . . . . .	138
7.2	KPCA parameters of each dataset . . . . .	145
7.3	Design parameters of the G-LWGMDH based EA method . . . . .	145
7.4	Overall mean performance of each method for each testing month using the New York dataset . . . . .	148
7.5	MAPE improvements of G-LWGMDH based EA over other methods using the New York dataset . . . . .	150
7.6	Comparison of the G-LWGMDH based EA method and other methods using the New York dataset (MAPE of the 24 Hours) . . . . .	152
7.7	Overall mean performance of each method for each testing month using the Victoria dataset . . . . .	156
7.8	MAPE improvements of G-LWGMDH based EA over other methods using the Victoria dataset . . . . .	157

# List of Abbreviations

AGC	Automatic generation control
AI	Artificial intelligence
ANN	Artificial neural network
ANNSTLF	Artificial neural network short term load forecaster
AR	Auto-regression
ARIMA	Autoregressive integrated moving average
ARIMAX	Autoregressive integrated moving average with exogenous variables
ARMA	Autoregressive moving average
ARMAX	Autoregressive moving average with exogenous variables
CD	Coordinate delay
DA	Decomposition Algorithm
EA	Evolutionary algorithm
EM	Expectation maximization
EP	Evolutionary programming
ERM	Empirical risk minimisation
EUNITE	European network on intelligent technologies for smart adaptive systems
FNN	Fuzzy neural network
GA	Genetic algorithm
G-LWGMDDH	Generalised locally weighted group method of data handling
GMDH	Group method of data handling
GP	Gaussian process
KKT	Karush-Kuhn-Tucker
KNN	K-nearest neighbours

KPCA	Kernel principal component analysis
LWGMDH	Locally weighted group method of data handling
LWR	Locally weighted regression
LWSVR	Locally weighted support vector regression
MA	Moving average
MAPE	Mean absolute percentage error
MLP	Multilayer perceptron
MSE	Mean square error
NMSE	Normalised mean square error
NYISO	New York independent system operator
PCA	Principal component analysis
QP	Quadratic programming
RBF	Radial basis function
SLT	Statistical learning theory
SMO	Sequential minimal optimization
SOPNN	Self-organising polynomial neural network
SRM	Structural risk minimisation
STLF	Short term load forecasting
SV	Support vector
SVM	Support vector machine
SVR	Support vector regression
VKG	Volterra-Kolmogorov-Gabor
WLS	Weighted least squares

# Chapter 1

## Introduction

The electric power system is often described as the most complex system devised by humans. It is a dynamic system consisting of generators, transformers, transmission and distribution lines, linear and nonlinear loads, and protective devices. The operation of these components should ensure the stability of the system even in cases of disturbances.

The forecasting of the future electricity load is one of the most important aspects of effective management of electric power systems. The precise load forecasting impacts the economic feasibility and the reliability of every electricity company. It helps the electric utility to make many important operating decisions such as scheduling of power generation, scheduling of fuel purchasing, maintenance scheduling and planning for energy transactions. The load forecasting results are used by the system operators as a basis of off-line network analysis to determine if the system might be vulnerable. If so, corrective actions should be prepared, such as load shedding and power purchases.

Load forecasting is always defined as basically the science or art of predicting the future load on a given system, for a specified period of time ahead. It can be generally categorised into short term, medium term and long-term based on the time span under consideration and the operating decision that needs to be made. In long term load forecasting, the energy demand is determined in the time rang up to 10 years ahead. It is needed for power system planning. The time span of medium-term load forecasting is in the range of a few weeks to a few months and it is needed for maintenance and fuel supply planning. While

the forecast time in short term load forecasting (STLF) is in the range of hours to a few days ahead and it is needed for the day to day operation of the power system.

STLF provides the input data for scheduling functions such as hydro-thermal coordination, scheduling of energy transactions and unit commitment as well as off-line studies such as load flow analysis and power system security studies. Because of its importance, STLF has been widely researched and a number of models were proposed during the past few decades [1].

In general, there are three main categories of the data sources which used as input variables of STLF: the historical load and weather databases, the model's parameters database and the real-time data obtained from the automatic generation control (AGC). The telemetered measurements in the real-time database are used by the AGC to determine the measured loads which used by the STLF model. The manually entered data may include weather updates and parameter data. The typical outputs of STLF are the estimated average load for every hour in the day, the daily peak load, and the daily or weekly energy generation. Fig. 1.1 illustrates a general input-output arrangement of a STLF system and its main uses [2].

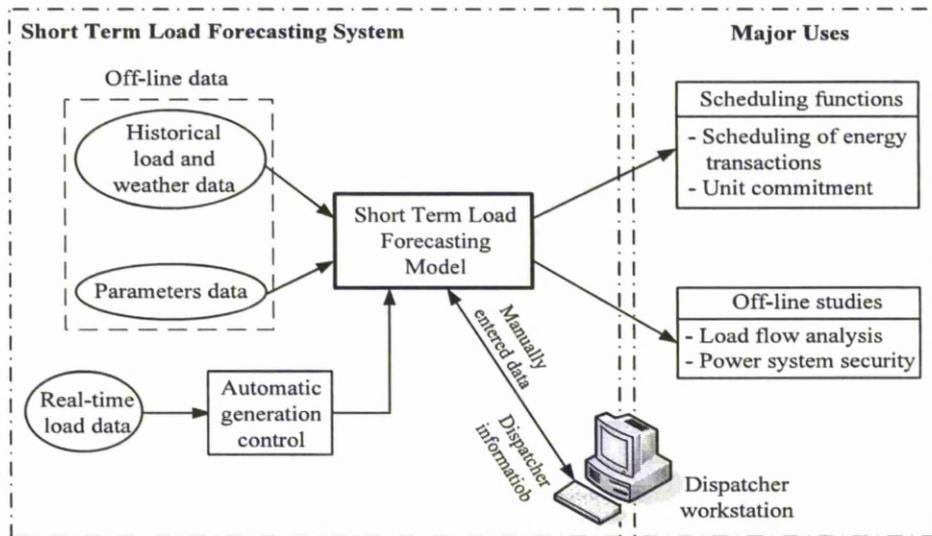


Figure 1.1: A general input-output arrangement of a STLF system.

## 1.1 Importance of short term load forecasting

STLF is the basis of the operation of today's power systems. Currently, no electric utility is able to operate in an economical, secure and reliable manner without STLF.

Sophisticated forecasting tools with higher accuracy are necessary to achieve lower operating costs and higher reliability of the electricity supply. Over-prediction of STLF wastes resources since more reserves are available than needed and, in turn, increases the operating cost. On the other hand, under-prediction of STLF leads to a failure to provide the necessary reserves which is also related to high operating cost due to the use of expensive peaking units [3]. It is estimated that in the British power system every 1% decrease in the forecasting error was associated with a commensurate decrease of the operating costs by 10 million per year [4].

STLF deals with load forecasting from one hour up to a few days and plays a vital role in a multitude of different day-to-day operational functions, such as generation scheduling, system security, load shedding, dispatcher, purchasing of additional power from neighbouring utilities, increasing the spinning and standby reserves of the system, etc. [1]. With the recent trend of deregulation of electricity markets, the need for more accurate STLF has gained more importance and greater challenges. In the market environment, precise forecasting is the basis of electrical energy trade and spot price establishment for the system to gain the minimum electricity purchasing cost.

STLF has been a very commonly addressed problem in power systems literature [5, 6, 7]. However there are numerous literatures on STLF published in last few decades, the accurate STLF is still a very important research issue to the electrical engineering scholars because of high complexity of loads, the system requirements, the stricter power quality requirements, and deregulation. How to estimate the future load with the historical data has remained a difficulty up to now, especially for the load forecasting of holidays, days with extreme weather and other anomalous days.

## 1.2 Objectives

Considering the complexity of the historical load data and the uncertainty of the influencing factors such as weather and environments, global methods, in which all data of time series are involved in modelling experiments, have been developed with phase space reconstruction and proposed to capture complicated load characteristics. They can exhibit a good performance in analysing and predicting the short term evolution in phase spaces reconstructed from analysis of the original time series.

In global methods, a prediction model is trained based on the entire data history and used to predict the load at a specific time with a fixed data window. However the application of global methods in solving STLF problem is highly addressed in literature [3, 5, 6], they have some drawbacks. One of the disadvantages of this method is that if new information is taken into consideration, all parameters of the model may need to be updated, and also a lengthy parameter re-estimation stage is required. Another disadvantage is concerned with its disability of capitalising the historical information of the time series directly, as only the current window is used for prediction at a specific time.

One of the objectives of this research work is to overcome the problems of global methods for STLF. Therefore, radial basis function (RBF) network, support vector regression (SVR) and Gaussian process (GP) based local predictors are introduced for solving the STLF problem.

In STLF, there are some training points more important than others. However using local SVR all training data contribute to the accuracy of the model to the same extent. This is due to the use of regularisation constant as a fixed value. The forecasting accuracy of the local SVR can be improved by weighting each training point according to its importance. So, another objective of this work is to extend the local SVR algorithm to achieve this goal by combining it with locally weighted regression.

Since precise STLF remains a great challenge, another objective of this work is to develop a new and practical algorithm with some up-to-date techniques. The group method of data handling (GMDH) is a self-organising method that used as a multivariate analysis method for complex systems modelling and iden-

tification. To improve the accuracy of solving the STLTF problem, the locally weighted group method of data handling (LWGMDH) method is developed in this research work.

The conventional GMDH algorithm, which generates an optimal structure of the model through successive generations of nodes being regarded as quadratic regression polynomials with two input variables, has some drawbacks [8]. Therefore, another objective of this work is to overcome these drawbacks by developing the generalised LWGMDH (G-LWGMDH) based on evolutionary algorithm (EA).

### 1.3 Thesis outline

The thesis is organised as follows:

Chapter 2 gives a literature survey for the short term load forecasting problem. The factors affecting the power system load are introduced firstly, followed by a description of the various forecasting methods.

In Chapter 3, the linear regression and nonlinear regression methods are introduced. The limitations of the linear regression and how can be overcome using nonlinear regression such as RBF, GP and SVR are discussed. Then this chapter outlines the mathematical background of these three nonlinear regression algorithms (RBF, GP and SVR).

Chapter 4 introduces the basics of time series phase space reconstruction. In order to reconstruct the time series phase space, the embedding dimension and time delay constant must be estimated from the time series. Therefore, a method for the estimation of the embedding dimension and time delay constant is introduced. In addition, three local predictors based on RBF, SVR and GP are presented and applied to solve the STLTF problem. Then the performance of these local predictors are compared with their global versions, seasonal autoregressive integrated moving average (ARIMA) and Holt-Winters exponential smoothing methods based on three real world datasets.

Chapter 5 extends the previous work of time series reconstruction to multivariate time series reconstruction. In addition, the basics of locally weighted

regression method are introduced. Moreover, our previous work on local SVR predictor is extended to locally weighted support vector machine. The weighted distance algorithm based on Mahalanobis distance is then presented. The performance of the proposed LWSVR method is evaluated using two real world datasets and the results are then discussed.

Chapter 6 proposes a novel method for STLTF. The kernel principal component analysis (KPCA) method and its application for time series reconstruction are presented. In addition, the basics of conventional GMDH and its advantages are discussed. Then the implementation of the proposed LWGMDH based KPCA method is described. Two test cases are presented to show the effectiveness of the proposed LWGMDH based KPCA method by comparing its performance with other methods.

Chapter 7 describes the design procedures of the proposed G-LWGMDH based EA method. A brief introduction to EA method and its operators are given firstly. Then, the implementation of the G-LWGMDH based EA method is described in details. The proposed method is applied to STLTF using two real world datasets. Moreover, the simulation results and comparisons with other methods are presented.

Chapter 8 concludes the work done in this thesis based on the outcomes obtained in this study. In addition, related research work that can be investigated in the future is suggested.

## 1.4 Major contributions

To overcome the problems of global methods for STLTF, a local predictor approach based on proven powerful regression algorithms, such as RBF, SVR and GP combined with phase space reconstruction of time series is presented. The local methods overcome the drawback of global methods by utilising part of the relevant history directly in the prediction model. Specifically, only the set of points ( $K$  nearest neighbours) of the reconstructed phase space which are close enough to the point under prediction are used to fit the local function. To get the best value of the number of nearest neighbours, a new systematic method

is proposed.

The performance of the proposed local predictors is evaluated with three real-world datasets and compared with the global versions of RBF, SVR and GP, as well as two conventional time-series predictors. The experimental results show the superiority of the local predictors over the global ones and the conventional time series predictors.

In Local SVR method, the regularisation parameter which determines the trade-off between the complexity and noise is constant. So, all training data contribute to the accuracy of the model to the same extent. However there are some training points more important than others. So, a new approach is proposed by combining the support vector regression and locally weighted regression, which can be called as locally weighted support vector regression (LWSVR).

LWSVR gives a weight factor to each train load datum. This means that the points that are close to the point under prediction have large weights, and the points far from the point under prediction have small weights. The weighting function's bandwidth plays an important role in local modelling. In order to overcome the disadvantage of using this bandwidth as a fixed value, the weighted distance algorithm is proposed to optimise it. To evaluate the performance of the proposed method, two real world datasets have been used. The results show that the LWSVR method can outperform other published methods.

The Group Method of Data Handling (GMDH) is a self-organising method that was firstly developed by Ivakhnenko [9]. There are few works reported about using GMDH to solve the STLF problem [10]. To improve its performance in solving the STLF problem, the LWGMDH method is proposed by combining the GMDH with local regression method and weighted least square regression.

The traditional time series reconstruction techniques have a serious problem which influence the quality of phase space reconstruction and modelling effect [11]. Consequently, the KPCA method has been used in the proposed method to improve the quality of nonlinear time series reconstruction. This leads to enhance the LWGMDH method's performance. Two real world datasets have been used to evaluate the performance of the proposed model. The simulation results show that the prediction performance can be greatly improved by using

the LWGMDH method.

In order to overcome the drawbacks of conventional GMDH which is used in LWGMDH implementation [8], the G-LWGMDH based on EA is proposed. In the proposed method, a new encoding scheme is presented to evolutionary design the G-LWGMDH network in which the connectivity configuration in such network is not limited to adjacent layers. Moreover, each node in the generalised LWGMDH network has different number of inputs and different polynomial order. To evaluate the performance of the proposed method, two real world datasets have been used. Comprehensive comparisons show that the forecasting performance of the G-LWGMDH based EA is significantly improved compared with LWGMDH and other published methods.

## 1.5 Publications

Below is the list of the publications that have arisen from this work including some that are under review:

### Journal manuscripts

- E. E. Elattar, J. Y. Goulermas, and Q. H. Wu. Electric load forecasting based on locally weighted support vector regression. *IEEE Transactions on Systems, Man and Cybernetics, Part C: Applications and Reviews*, In Press (Published online on 5<sup>th</sup> February 2010).
- E. E. Elattar, J. Y. Goulermas, and Q. H. Wu. Integrating KPCA and locally weighted GMDH for prediction of nonlinear time series. Submitted to *Int. Journal of Forecasting*, September 2009.
- E. E. Elattar, J. Y. Goulermas, and Q. H. Wu. Generalized locally weighted GMDH for prediction of nonlinear time series. Submitted to *IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics*, December 2009.
- E. E. Elattar, J. Y. Goulermas, and Q. H. Wu. Local prediction methods for short-term load forecasting. Submitted to *European Transactions on*

---

*Electric Power*, February 2010.

### Conference papers

- E. E. Elattar, J. Y. Goulermas, and Q. H. Wu. Forecasting electric daily peak load based on local prediction. In *IEEE Power Engineering Society General Meeting (PESGM 2009)*, Calgary, Canada, July 26-30, pp. 1-6, 2009.
- E. E. Elattar, J. Y. Goulermas, and Q. H. Wu. Integrating KPCA and locally weighted support vector regression for short-term load forecasting. In *the 15<sup>th</sup> IEEE Mediterranean Electrotechnical Conf. (MELECON 2010)*, Valletta, Malta, April 25–28, pp. 1528–1533, 2010.
- E. E. Elattar, J. Y. Goulermas, and Q. H. Wu. Evolutionary design of the modified locally weighted GMDH for short term load forecasting. Submitted to *14<sup>th</sup> IEEE International Middle East Power System Conference (MEPCON 2010)*, March 2010.

## Chapter 2

# Short Term Load Forecasting Literature Survey

Generally, the load of an electric utility is composed of very different consumption units. Industrial activities consume a large part of the electricity while another part is of course used by private people and many services offered by society. The sum of all the consumers load at the same time is called the system load.

The rest of this chapter is organised as follows: In Section 2.1 the factors which affect the system load is discussed. Then some main STLF methods are reviewed in Section 2.2.

### 2.1 Factors affecting the power system load

Generally, the load of an electric utility is composed of very different consumption units. Industrial activities consume a large part of the electricity while another part is of course used by private people and many services offered by society. The sum of all the consumers load at the same time is called the system load. Good understanding of the system characteristics helps to design reasonable forecasting models and select appropriate models in different situations.

There are many factors that affect the load changes. They can be generally classified as weather, calendar, economical, and random factors. The effects of these factors are introduced as follows to provide a basic understanding of the

power system load characteristics.

- Weather factors: These factors include temperature, humidity, cloud cover, light intensity and so on [12]. The change of the weather causes the change of consumers comfort feeling and in turn the usage of some appliances such as heaters and air conditioners. Weather-sensitive load also includes appliance of agricultural irrigation due to the need of the cultivated plants. The load patterns differ greatly in the areas where summer and winter have great meteorological difference. Figure 2.1 shows the typical different seasonal weekdays of England and Wales load profiles of the year 2005.

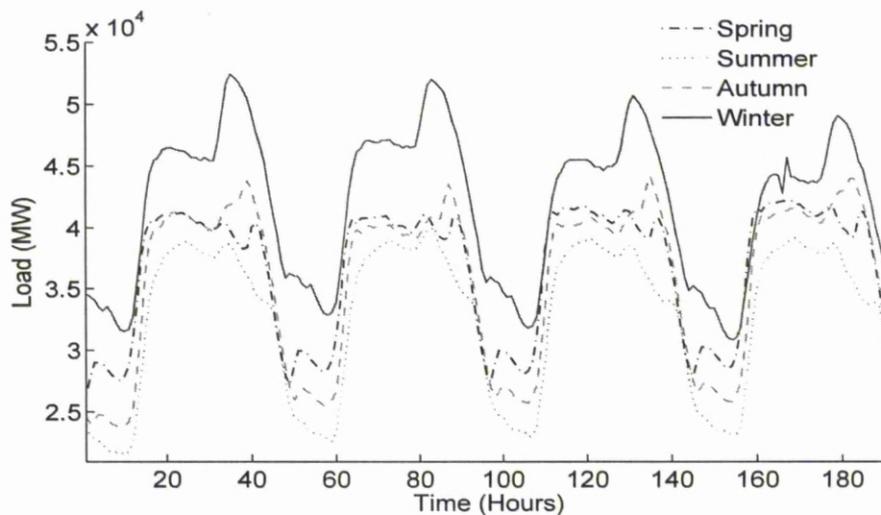


Figure 2.1: Typical seasonal workdays of England and Wales load profiles of the year 2005.

This pattern indicates the relation between the load usage and weather conditions in different seasons. The previous days' temperatures also affect the load profile. For example, continuous high temperature days might lead to heat buildup and in turn a new system peak. Humidity affects the human beings comfort feeling greatly, so humidity is also an important factor. People feel hotter in the environment of  $37^{\circ}\text{C}$  and 80% relative humidity than in the environment of  $40^{\circ}\text{C}$  and 40%. That is why temperature humidity index is sometimes employed as an affecting factor of load forecasting. In addition, wind chill index is another factor that measures

the cold feeling. Selecting the appropriate weather variables as the inputs of load forecasting is a meaningful topic.

- Calendar factor: Examples of calendar factors are day of the week, season, hour of forecast, sunrise/sunset, etc. From the observation of the load curves it can be seen that there is very strong daily, weekly, seasonal and yearly periodicity in the load data. For example, Fig. 2.2 shows the typical load curve of England and Wales for few weeks in year 2005.

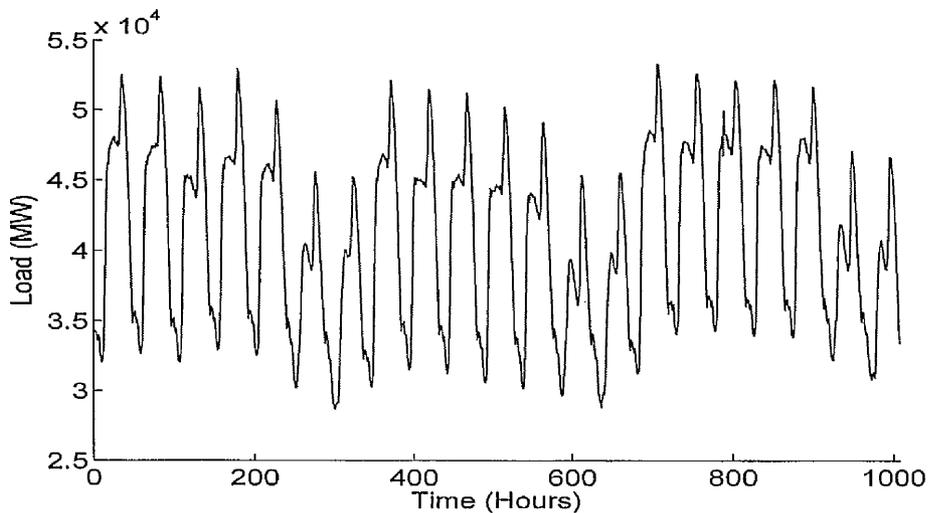


Figure 2.2: Typical load curve of England and Wales for few weeks in year 2005.

From Fig. 2.2, we can observe that the load in weekdays (Monday through Friday) is usually higher than that of weekend. The load on different weekdays also can behave differently. In addition, the load in Sunday is a little lower than that on Saturday.

Holidays are more difficult to forecast than non holidays because of their relative infrequent occurrence [13].

- Economic factors: The economic factors play an important role to determine the evolution of the electricity demand. Economic factors, such as the degree of industrialisation, price of electricity and load management policy have significant impacts on the system load growth/decline trend. The relationship between electricity price and load profile is even stronger with

the development of modern electricity markets [14, 15]. Although time-of-use pricing and demand-side management had arrived before deregulation, the volatility of spot markets and incentives for consumers to adjust loads are potentially of a much greater magnitude. At low prices, elasticity is still negligible, but at times of extreme conditions, price-induced rationing is a much more likely scenario in a deregulated market compared to that under central planning.

- Random factors: The modern power system is composed of numerous electricity users. The startup and shutdown of the large loads, such as steel mill is an important source of random disturbance and always lead to an obvious impulse to the load curve. The startup and shutdown time of these users is quite random, *i.e.*, there is no obvious rule of when and how they get power from the grid. In addition, special events are another source of random disturbance. These events may be known in advance to the dispatcher but their effect on load is not quite certain. A typical special event is, for example, a large social event or a popular TV show, which the dispatchers know for sure will cause increasing usage of electricity, but cannot best decide the amount of the usage [16, 17].

## 2.2 Classification of developed STLF methods

Short Term load forecasting (STLF) of electric power not only plays a very important role in operation scheduling, like economic emission dispatch, unit commitment, energy transactions, and fuel purchasing, but also has a significant impact on the secure operation of power systems. Accurate load forecasts will lead to appropriate operation and planning for the power systems, thus achieving a lower operating cost and higher reliability of the electricity supply.

Over the last few decades a number of forecasting methods have been developed. A variety of methods, which include similar day approach, various regression models, time series, neural networks, expert systems, fuzzy logic, etc. are used for short term load forecasting. The development, improvements and investigation of the appropriate mathematical tools will lead to the development

of more accurate load forecasting techniques.

In general, the research approaches of STLF can be mainly divided into two categories: statistical methods and artificial intelligence (AI) based techniques [5]. The former include multiple linear regression [18], stochastic time series [19], state space [20], Kalman filtering [21], etc.

Statistical methods usually employ a mathematical model that represents load as function of different factors such as time and weather. However, the statistical methods can predict the load curve of ordinary days very well, they lack the ability to analyse the load property of holidays and other anomalous days, due to the inflexibility of their structure.

In order to improve the performance of statistical STLF techniques in predicting load patterns, researchers have focused much of their attention to AI based techniques, such as artificial neural networks [22], expert system [23], fuzzy inference [24], radial basis function [25], etc. These methods try to imitate human beings' way of thinking and reasoning to get knowledge from the historical data and forecast the future load.

Recently support vector regression (SVR), which is a very promising statistical learning method, has also been applied to STLF with success [26]. Another method used for function regression is the Gaussian Process (GP) which is based on Bayesian modelling. The application of the GP to STLF problems in [27] has showed a high accuracy achieved especially at noisy environments.

Some main short term load forecasting methods can be introduced as follows.

### 2.2.1 Regression methods

Regression methods are widely used for electric load forecasting because they are relatively easy to implement. Regression methods normally assume that the load can be divided into a standard load component and a component linearly dependent on some explanatory variables such as weather and day type [1]. Of all the weather variables, the temperature is the most significant variable commonly used in regression analysis [28]. Other weather related variables such as humidity and wind speed can also be included in the model to provide a better forecasting results.

Selection the proper input variables is the first step in performing regressive load method. The effect of each input variable can be identified by correlation analysis. The basic functional element is then formulated among the group of load affecting variables. The least square estimation technique is widely used to estimate the proper regression coefficients.

Haida and Muto [29] presented a regression-based daily peak load forecasting method with a transformation technique to deal with seasonal load change, annual load growth and the latest daily load shape.

Ramanathan *et al.* [30] developed a regression model which applied to historical data for the North American electric utility. The approach is a multiple regression model, one for each hour of the day (with weekends modeled separately), with a dynamic error structure as well as adaptive adjustments to correct for forecast errors of previous hours. Its application to forecast hourly loads, for two years gives better performance than a wide range of alternative models.

El-Hawary and Mbamalu [31] describe a method to forecast short-term load requirements using an iteratively reweighted least squares algorithm. The proposed model is applied to predict the Nova Scotia power corporation's 24 hours ahead hourly load. The results obtained are compared with results obtained using the ordinary least squares procedure in order to show the superior performance of the proposed approach. References [32, 33, 34], describe other applications of regression methods to load forecasting.

Although regression-based methods are widely used, they suffer from some drawbacks. There may be inherent problems in identifying the correct model because of the nonlinear and complex relationship between the load demand and the influencing factors [1]. In addition, regression-based methods may suffer from numerical instability.

### 2.2.2 Time series methods

Time series methods model the load demand as a function of historical data by assuming that the data follow a certain stationary pattern that depends on autocorrelation, trends in the data, and daily, weekly and seasonal variations [28].

Time series have been used for decades in different fields such as digital sig-

nal processing, economics, as well as electric load forecasting. In the literature, time series models appear in different forms such as Box-Jenkins, stochastic models, autoregressive moving average (ARMA), autoregressive integrated moving average (ARIMA), autoregressive moving average with exogenous variables (ARMAX), autoregressive integrated moving average with exogenous variables (ARIMAX), and state space models.

The basic idea in time series prediction of load demand is that the load value is assumed to be a combination of two components. The first component namely the basic component is a nonlinear function representing the periodic behaviour of load. This function depends only on the time of the day. The second component reflects the load deviation from normal load curve due to the environmental and random factors [1]. This deviation can be evaluated in terms of ARMA process. The effect of recent load data is introduced by the auto-regression term (AR) while the term moving average (MA) describes the current hour random component as a weighted random sequence from previous hours.

The combined model derived from above two principles (AR and MA) is called the autoregressive-moving average (ARMA) model. The ARMA process of order  $p$  and  $q$  can be expressed as follows [35]:

$$\Phi(B)y_t = \Theta(B)a_t \quad (2.2.1)$$

where  $y_t$  is the time series of load deviation component,  $a_t$  is a series of random variables in normal distribution ( $t = 1, \dots, N$ ),  $N$  is the number of data points,  $B$  is the backward shift operator,  $\Phi(B) = 1 - \phi_1 B - \phi_2 B^2 - \dots - \phi_p B^p$  is the AR operator,  $\Theta(B) = 1 - \theta_1 B - \theta_2 B^2 - \dots - \theta_q B^q$  is the MA operator,  $p$  and  $q$  are the AR and MA order, respectively, ,

ARIMA is a generalisation of ARMA model, where it can be applied for the non-stationary time series. The basic ARIMA model can be written as:

$$\Phi(B)\nabla^d y_t = \Theta(B)a_t \quad (2.2.2)$$

where  $\nabla^d = (1 - B)^d$  is the backward difference operator and  $d$  is the difference order.

The load series incorporates seasonal variation, so this basic ARIMA model is not suitable for describing the load time series [28]. Therefore, the differencing with the period of seasonal variation (usually 24 for one day and 168 for one week) is required. The obtained model is called a seasonal ARIMA model and can be written as follows [35]:

$$\Phi(B)\Phi_S(B^S)\nabla^d\nabla_S^D y_t = \Theta(B)\Theta_S(B^S)a_t \quad (2.2.3)$$

where  $\nabla_S^D = (1 - B^S)^D$ ,  $S$  is the period of the seasonal variation and  $D$  is the seasonal difference order.

ARMA and ARIMA models use the time and load as the only input parameters. However, an external input variable, such as temperature in the case of load time series, can also be included in the model. Such a variant of the ARIMA model is called an ARIMAX model [36], and can in general be written:

$$\Phi(B)\nabla^d y_t = w(B)u(t) + \Theta(B)a_t \quad (2.2.4)$$

where  $u(t)$  is the exogenous external variable (such as temperature) at time  $t$ ,  $w(B) = w_0 + w_1B + w_2B^2 + \dots + w_rB^r$  is the exogenous external variable operator and  $r$  is the external variable order.

Amjady [19] uses ARIMA to forecast hourly loads of weekdays, as well as, of weekends and public holidays. The ARIMA is used to tune the unknown parameters using past values of the load demand and past values of the inputs, and then uses the model to forecast the load demand for unknown points of the operating system. The obtained results from extensive testing on the Iran's power system network show the validity of the developed approach.

The evolutionary programming (EP) is used in [36] to identify the ARMAX model parameters for one day to one week ahead hourly load demand forecast. In [36], the temperature variable is incorporated in the ARMAX model as an exogenous variables. Forecasting results of the EP based identification method was verified to be superior to that of the traditional identification techniques.

Huang *et al.* [7] proposed a new particle swarm optimisation approach to identifying the ARMAX model for one-day to one-week ahead hourly load forecasts. A periodic autoregression model used by Espinoza *et al.* to develop a

set of 24 seasonal equations with 48 parameters each [37]. The set of equations is extended to include exogenous variables that describe the temperature effects and the monthly and weekly seasonal variations. In [38], the seasonal autoregressive moving average model is used in the comparison of univariate methods for prediction up to a day ahead using data obtained from 10 European countries.

The weakness in the time series methods is in the adaptability. If there is no change in the variables that affect load demand (such as environmental or social variables), time series methods can give satisfactory results. In reality, the load behaviour can change quite quickly at certain parts of the year, so the model can not adapt to the new conditions very quickly, even if model parameters are estimated recursively.

The treatment of the anomalous load conditions is another problem in time series methods. If the load behaviour is abnormal on a certain day, this divergence from the normal conditions will be affect the forecasts in the future. Furthermore, time series methods require a significant computational time and may result in numerical instabilities because there is a need to use a large amount of historical data and a large number of complex relationships [1].

### 2.2.3 Neural networks

The use of artificial neural networks (ANN) has been a widely studied electric load forecasting technique since 1990 [39, 40, 41]. Neural networks are suitable for load forecasting because of their approximation ability for nonlinear mapping and generalisation, its clear model, easy implementation and good performance [42].

#### Short introduction to neural networks

Artificial neural networks are mathematical tools based on models of biological neurons. Their basic unit is the artificial neuron. The neuron receives information through a number of input nodes, processes it internally, and puts out a response. The processing is usually done in two stages [43]. First, the input values are linearly combined, then the result is used as the argument of a nonlinear activation function. The combination uses the weights attributed to

each connection, and a constant bias term. The activation function must be a nondecreasing and differentiable function; the most common choices are either the identity function, or bounded sigmoid (s-shaped) functions [5].

In practice network, the neurons are arranged in a relatively small number of connected layers of elements between network inputs and outputs. One of these arrangement types is the multilayer perceptron (MLP) type, in which the neurons are organised in layers which are input layer, output layer and some hidden layers (the layers between the input nodes and the output layer). The neurons in each layer may share the same inputs, but are not connected to each other. If the architecture is feed-forward, the outputs of one layer are used as the inputs to the following layer. The parameters of this network are the weight matrix ( $W$ ).

The estimation of the weight matrix is called the training of the network, and is done by the minimisation of a loss function. The back-propagation training algorithm is the first devised training algorithm, which uses a steepest-descent technique based on the computation of the gradient of the loss function with respect to the network parameters (that is the reason why the activation functions must be differentiable). Many other training algorithms such as Hopfield, and Boltzmann machine are now available [43].

### STLF based on neural networks

Back propagation neural networks is the most popular artificial neural network architecture for electric load forecasting. Back propagation neural networks use continuously valued functions and supervised learning. That is, under supervised learning, the actual numerical weights assigned to element inputs are determined by matching historical data (such as time and weather) to desired outputs (such as historical electric loads) in a pre-operational training session. Artificial neural networks with unsupervised learning do not require pre-operational training [28].

Bakirtzis *et al.* [3] developed an ANN based short term load forecasting model for the energy control centre of the Greek public power corporation. A fully connected three-layer feedforward ANN and back propagation algorithm was used

for training in the developed model. The historical hourly load data, temperature, and the day of the week are used as the inputs for the model to forecast load profiles from one to seven days.

Khotanzad *et al.* [44] described a load forecasting system known as artificial neural network short term load forecaster (ANNSTLF) which based on multiple ANN strategies that capture various trends in the data. In the development model, they used a multilayer perceptron trained with the error back propagation algorithm and consider the effect of temperature and relative humidity on the load. ANNSTLF also contains forecasters that can generate the hourly temperature and relative humidity forecasts needed by the system. A new generation of the above system was described in [45]. In the improved model, ANNSTLF consists of two ANN forecasters, one forecasts the base load and the other forecasts the change in load. The final forecast is computed by an adaptive combination of these forecasts. Moreover, the improved model considers the effects of humidity and wind speed through a linear transformation of temperature.

Senjyu *et al.* [22] proposed a one-hour-ahead load forecasting method using the correction of similar day data. In this method, the forecasted load power is obtained by adding a correction to the selected similar day data. The correction is yielded from the neural network. Since the neural network yields the correction which is a simple data, it is not necessary for the neural network to learn all similar days data. Therefore, the neural network can forecast load power by simple learning. If the forecast day is changed, the neural network is retrained and it can obtain the relationship between load and temperature around the forecast day. Therefore, it is possible to deal with seasonal change by using the proposed neural network.

The electricity pricing is suggested as an additional term that can be included in the model in [14]. Naturally, price decreases/increases affect electricity consumption. Large cost sensitive industrial and institutional loads can have a significant effect on loads. The Pennsylvania-New Jersey-Maryland (PJM) spot price data (as it related to Ontario Hydro load) is used in [14] as a neural network input. The results show that accurate estimates were achieved more quickly with the inclusion of price data.

Some other researchers presented a three layer neural network for the forecasting of the next hour, peak and total daily load. Ramezani *et al.* [42] presented the development of three neural network based models for the forecasting of the next hour load, the next day load and the next day peak load. They used the multilayer perceptron (a feed-forward one) with three layer and back-propagation for off-line training. This model uses load profile and weather situation in input layer.

The ANN method, using data from the few weeks proceeding the target day as training data, can not accurately forecast seasonal trends mainly due to insufficient learning pattern, so Afkhami *et al.* [46] presented the development of an artificial neural network based short term load forecasting model. The purpose of this model is to forecast load accurately, using actual data from the same period of previous several years as training data in order to expand learning pattern. The ANNs described in this paper are multilayer perceptron structures designed to forecast the hourly load for 24 hours ahead. The back propagation has been for ANN training. Networks are trained using hourly historical Load data and daily historical max/min temperature and humidity data.

A conventional ANN model sometimes can suffer from a sub-optimisation problem. To improve the use of ANN in load forecasting, some researchers used other optimisation techniques. Ling *et al.* [47] proposed a novel neural network model which uses genetic algorithm (GA) in learning process. In this model two activation functions are used in the neuron and a node-to-node relationship is proposed in the hidden layer. This network model is found to be able to give better performance than the traditional feedforward neural network. A GA with arithmetic crossover and nonuniform mutation can help in tuning the parameters of the proposed network.

To create a superior forecasting method, Liao *et al.* [48] proposed to use a fuzzy neural network (FNN) combined with a chaos-search genetic algorithm and simulated annealing (hereafter called the FCS method or simply FCS) to exploit the advantages of the two methods and, furthermore, to eliminate the known drawback of the traditional ANN trained by the back propagation method. This method is next put to the test for STLF using some data obtained from a study

for various time periods, such as one year, one week, or 24 h. An extensive review and evaluation of neural network methodologies for short term load forecasting is provided in [5].

Although ANN based STLF methods are highly addressed in the literature, they suffer from some important drawbacks such as long training time and slow convergent speed. In addition, ANNs have multiple local minima problem when using with nonlinear systems because of the adoption of empirical risk minimisation (ERM) principle. Moreover, there was no reliable theory to determine the structure of the network.

#### 2.2.4 Expert systems

A successful application of AI methodologies applies specific knowledge and inference to simulate human reasoning in solving difficult problems. Expert Systems are computer software programs that solve problems in well-bounded problem areas (domain) that would require the knowledge and reasoning skills of an expert [17]. Expert systems are organised in three distinct levels:

- Knowledge base consists of problem-solving rules, procedures, and intrinsic data relevant to the problem domain.
- Working memory refers to task-specific data for the problem under consideration.
- Inference engine is a generic control mechanism that applies the axiomatic knowledge in the knowledge base to the task-specific data to arrive at some solution or conclusion.

The organisation of these three levels shown in Fig. 2.3 [49].

The knowledge base constitutes the problem-solving rules, facts, or intuition that a human expert might use in solving problems in a given problem domain. The knowledge base is usually stored in terms of if-then rules.

The working memory represents relevant data for the current problem being solved. The inference engine is the control mechanism that organises the problem data and searches through the knowledge base for applicable rules. The development of a functional expert system usually centres around the organisation of

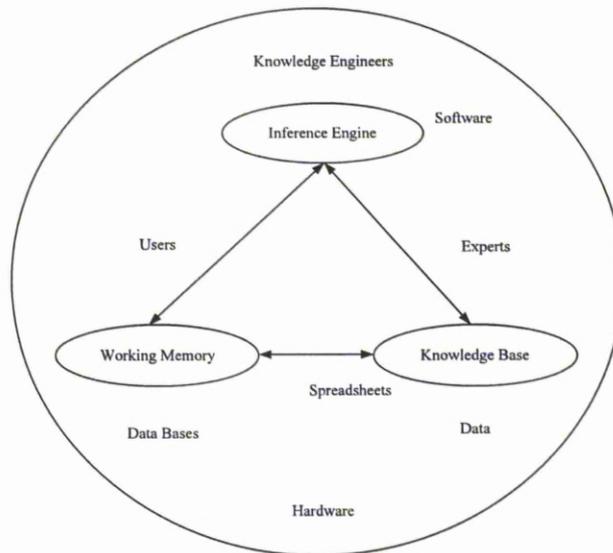


Figure 2.3: Organisation of the expert systems.

the knowledge base. A functional integration of expert systems components is shown in Fig. 2.4 [49].

The idea behind this algorithm is the utilisation of rule-based techniques derived from human experts; then, building a model that can emulate human reasoning and come out with decisions concerning a particular problem. For the load forecast problem, a possible method is to create the database associated with particular day types, social factors, and weather factors. Using an automated search process, the load pattern in a similar day can be predicted [49].

Ho *et al.* [23] described a knowledge-based expert system for short term load forecasting which developed for Taiwan power system. Operators knowledge and the hourly observations of system load over the past five years were employed to establish eleven day types. he developed algorithm also consider the weather parameters and gives better performance than the conventional Box- Jenkins method.

Rahman and Hazim [50] developed a short term load forecasting algorithm by combining the knowledge based expert systems and statistical techniques. The proposed method uses a limited set of historical data that resembles the target day. To make method site-independent, this data set is adjusted to location

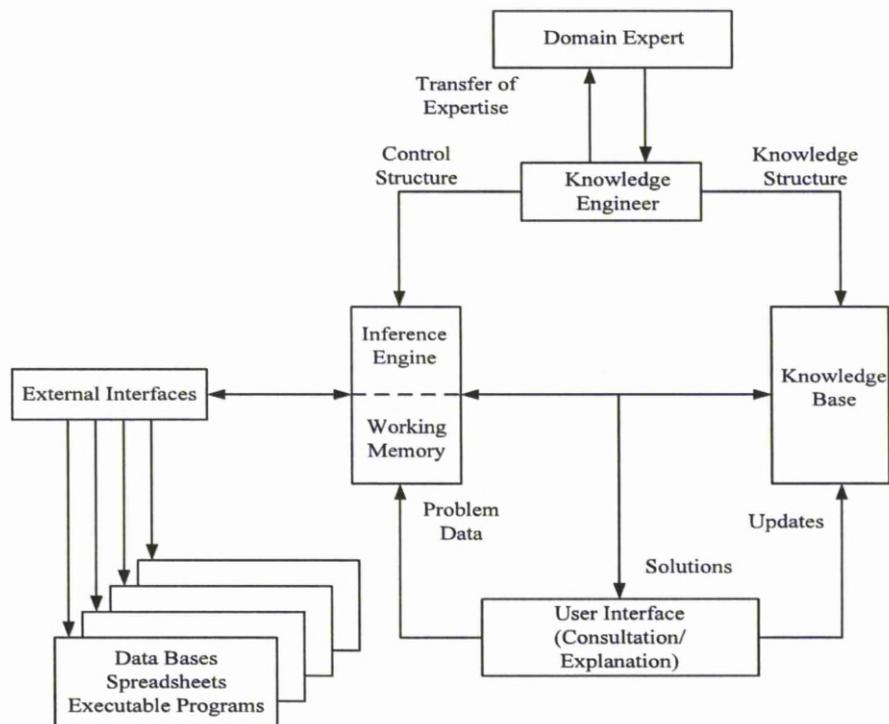


Figure 2.4: The functional integration of expert systems components.

specific conditions. Four different electric utilities in the United States were used to show the adaptiveness of the developed method. The authors analysed the knowledge acquisition and rule development process in implementing the expert system in a later paper [17]. The technique was tested in several sites in the United States with good forecasting performance. The load model, the rules, and the parameters presented in the paper have been designed using no specific knowledge about any particular site.

Hwang *et al.* [51] proposed a new practical knowledge-based expert system for short term load forecasting equipped with graphical user interface. The performance of the proposed method was tested using the load data of past 4 years. The simulation results show that the forecasting accuracy is improved comparing to the conventional methods.

Although expert systems improve the accuracy of STLF, they have some drawbacks [52]. First, they suffer from lacks of common sense needed in some decision making. In addition, errors may occur in the knowledge base, and lead

to wrong decisions. Finally, they can not adapt to changing environments, unless knowledge base is changed.

### 2.2.5 Integration of different algorithms

As there are many presented methods for STLF, it is natural to combine the results of several methods. Chen *et al.* [53] presented a novel similar day-based wavelet neural network method. The key idea is to use a similar day technique to select good input load, use wavelet to decompose the load into low and high frequency components, and then use separate neural networks to predict the different frequency components.

The combination of fuzzy logic and artificial neural networks creates a hybrid system that is able to combine the merits of each technique and overcome their drawbacks [54]. Chauhan *et al.* [55] presents a neuro-fuzzy hybrid system to forecast the hourly load. The author compare between two approaches the first one is ANN using back propagation algorithm and second one is neuro-fuzzy hybrid system considering type-2 fuzzy sets instead to type-1 fuzzy sets. The performance has been tested by one year load and temperature data. The results show that the performance of the system using neuro-fuzzy hybrid approach is better than conventional ANN.

Liao and Tsao [48] proposed a fuzzy neural network combined with a chaos search genetic algorithm and simulated annealing to forecast the electric power load profile. The author proposed this combination to exploit their advantages and, furthermore, to eliminate the known downside of the traditional ANN. To test the performance of proposed method, the author used two years load, temperature and rainfall index data. The results show that the proposed method is superior to other commonly used forecasting methods. Fuzzy inference was also used in combination with ANN [24] for one step ahead daily maximum load forecasting. A number of other papers in the literature describe applications of the neural fuzzy network to load forecasting [56, 57].

Mori and Kosemura [58] proposed a data mining method for discovering STLF rules. The method is based on a combination between the optimal regression tree and an artificial neural network. First, the proposed method classifies

the load range into several classes, and decides which class the forecasted load belongs to according to the classification rules. Then the artificial neural network is used to predict one-step ahead loads. The proposed method is demonstrated with actual data and gave a good performance.

Li *et al.* [59] introduced a new method based on data mining to reflect the influence of weather factor on load. In this method, the decision tree is used to construct the model and used it to make short term load forecasting. The principle of maximum information plus which can reduce the complexity of decision tree was used to sort the test attributes during the construction of decision tree. Numerical results indicates that this method can improve the precision of forecasting.

The combined of support vector machine (SVM) method and similar day method for next day load forecasting has been presented in [60]. This method forecasts the load of next day using SVM. Then, the load curve of a similar day is selected to correct the curve forecasted by SVM, which can avoid the appearance of large forecasting error effectively. To evaluate the method proposed in [60] the practical power load data from China is used to predict the daily energy consumption 24-hour ahead and the final forecasting result was accurate and reliable. Cai and Min [61] proposed a novel method based on SVM using similar day's load data as the training sample data for power load forecasting. The study in [61] used the city of Henan province China load data and compared the forecasting data with the operational data. The authors reported that the method can increase training efficiency by properly choosing the similar days.

## Chapter 3

# Mathematical Background

The linear regression method is the simplest data modelling technique. It attempts to model the relationship between two variables by fitting a linear equation to observed data. In order to overcome the limitations of linear regression methods the nonlinear regression methods are used. However, the training time for the nonlinear regression methods are longer than linear regression. This happens because the nonlinear regression methods require many more training examples than linear regression.

The rest of this chapter is organised as follows: In Section 3.1 the basics of linear regression method are reviewed. Then the limitations of linear regression method is discussed. Section 3.2 presents an introduction of the nonlinear regression. Sections 3.3 outline the basics of RBF networks and its training techniques. In Section 3.4, the SVR method is presented and the kernel which is one of the characteristics of SVR is studied. The GP method and its prediction process are reviewed in Section 3.5. Finally, this chapter is concluded in Section 3.6.

### 3.1 Linear regression

Linear regression is a parametric regression method [62]. It refers to any approach to modelling the relationship between one or more variables denoted  $y$  and one or more variables denoted  $X$ , such that the model depends linearly on the unknown parameters to be estimated from the data. Given a dataset  $\{y_i, \mathbf{x}_i\}_{i=1}^N$ ,  $\mathbf{x}_i = [x_{i1}, \dots, x_{id}]^T$ , where  $N$  is the number of data points and  $d$  is the

dimension of input space. Linear regression model assumes that the relationship between the variable  $y$  (target function) and  $x$  is approximately linear. The difficulty is that the sample can be corrupted by additive normal distribution noise. This noise distribution is a good approximation to many real world cases. The model takes the form [62]:

$$y_i = \beta_1 x_{i1} + \dots + \beta_d x_{id} + \epsilon_i = \mathbf{x}_i^T \boldsymbol{\beta} + \epsilon_i \quad i = 1, \dots, N, \quad (3.1.1)$$

or matrix form as follows:

$$\mathbf{y} = f(\mathbf{x}) = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon} \quad (3.1.2)$$

where  $\mathbf{y}$  is a vector whose  $i^{\text{th}}$  element is  $y_i$ ,  $\mathbf{X}$  is a matrix whose  $i^{\text{th}}$  row is  $\mathbf{x}_i^T$ ,  $\boldsymbol{\beta} = [\beta_1, \beta_2, \dots, \beta_d]^T$  are the unknown parameters to be estimated from the data and  $\epsilon_i$  are uncorrelated errors (noises) having zero means ( $E[\epsilon] = 0$ ) and the same variance. Thus the dimensionality of  $\mathbf{X}$  is  $N \times d$ .

Gauss's theorem of least squares may be stated in the following form: the estimates  $\hat{\boldsymbol{\beta}}^T = (\hat{\beta}_1, \hat{\beta}_2, \dots, \hat{\beta}_d)$  of the parameters  $\boldsymbol{\beta}$ , which are linear in the observations and which minimise the mean square error of any linear function of the parameters, are obtained by minimising the sum of squares [35]:

$$S(\boldsymbol{\beta}) = \boldsymbol{\epsilon}^T \boldsymbol{\epsilon} = (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^T (\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) \quad (3.1.3)$$

The function  $S$  is known as the square loss function. The minimum of this loss function ( $S(\boldsymbol{\beta})$ ) can be established by differentiating the square loss function (3.1.3) with respect to  $\boldsymbol{\beta}$  and setting them equal to zero.

$$\frac{\partial S(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = 2(\mathbf{X}^T \mathbf{y} - \mathbf{X}^T \mathbf{X} \boldsymbol{\beta}) = 0 \quad (3.1.4)$$

yields the well-known 'normal equation'

$$\mathbf{X}^T \mathbf{X} \hat{\boldsymbol{\beta}} = \mathbf{X}^T \mathbf{y} \quad (3.1.5)$$

therefore, the minimum least square estimator  $\hat{\boldsymbol{\beta}}$  is:

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y} \quad (3.1.6)$$

The approximation function of the target function is:

$$\hat{f}(\mathbf{x}) = \mathbf{x}^T \hat{\boldsymbol{\beta}} \quad (3.1.7)$$

The linear regression requires fewer examples than nonlinear regression. Therefore, its estimation of the parameters is faster than the nonlinear regression methods.

There are several limitation of the least square estimator. In many cases, the noise is not Gaussian distributed and the variance of the noise is varying. So, one may assume that the parameters  $\beta$  are Gaussian distributed instead of assuming the noise is Gaussian. this assumption leads to the Gaussian Process (GP) regression algorithm [63].

Obviously, there are infinite functions that minimise the loss function ( $S(\beta)$ ), hence the estimation of the least square method is ill-posed. To solve this problem, various regularisation techniques can be applied in such cases. The most common of which is called Tikhonov regularisation [64].

In addition, the square loss function used in linear regression gives a heavy weight to a relatively small number of examples when the error is large. Therefore, a small number of examples dominates the value of the loss function. To overcome this problem, support vector machine uses a special loss function [65].

## 3.2 Introduction to nonlinear regression

Nonlinear regression is a form of regression analysis in which the predictor does not take a predetermined form but is constructed according to information derived from the data. Nonlinear regression requires larger sample sizes than regression based on parametric models because the data must supply the model structure as well as the model estimates.

The general nonlinear regression model is written in a similar manner to the linear regression, but the function  $f$  is left unspecified:

$$y_i = f(x_{i1}, x_{i2}, \dots, x_{id}) + \epsilon_i \quad i = 1, \dots, N, \quad (3.2.1)$$

The object of nonlinear regression is to estimate the regression function  $f$  directly, rather than to estimate the parameters. Most methods of nonlinear regression implicitly assume that  $f$  is a smooth, continuous function [66]. Moreover, in the nonlinear regression, it is not restricted to approximation functions

that have to be linear like linear regression. The approximation function is assumed to belong to a class of functions such as a class of polynomial functions or a class of spline functions.

This assumption is added to the loss function through a constraint. So, the loss function of the nonlinear regression always involves two terms. The first one specifies the fitting of the data while the second one specifies the constraint of the functions.

There are many specific methods of nonlinear regression including kernel estimation [66, 67], local polynomial regression [66, 68], smoothing splines [69], K-nearest neighbours (KNN) regression [70, 71], etc. These all allow for great flexibility in the possible form of the regression surface and make no assumption about the parametric form of the model. However, many methods of nonlinear regression do not perform well when the number of independent variables in the model is large. The sparseness of data in this setting causes the variances of the estimates to be unacceptably large. The problem of rapidly increasing variance for increasing dimensionality is sometimes referred to as the “curse of dimensionality”.

### 3.2.1 Curse of dimensionality

The curse of dimensionality is the main practical problem of nonlinear regression estimation. It comes from the fact that all finite training samples are very sparse in the input space. The curse of dimensionality arises because nonlinear regression estimators are dependent variable averages local to the point at which the regression function is to be estimated. The number of observations ‘local’ to the point of estimation increases exponentially with the number of dimensions.

The curse of dimensionality problem can be overcome by restricting the target function  $f$  to a member of a class functions, hence restricting the complexity of the approximation function [72]. Suppose the parameters  $\hat{\beta}$  have been estimated using the least square method, it can be shown that the expectation of the square distance between target function and the approximation function over

the distribution of noise is [73]:

$$E(f(x) - \hat{f}(x))^2 = \frac{d\sigma^2}{N} \quad (3.2.2)$$

which increases only linearly with the dimensionality of input space, where  $N$  is the number of data points,  $d$  is the dimension of the input vector ( $x$ ) and  $\sigma^2$  is the variance. In general, the fixed basis function is considered as follows:

$$\hat{f}(x) = \sum_{i=1}^H \beta_i \psi_i(x) \quad (3.2.3)$$

where  $H$  is the number of basis function and  $\psi_i$  is a smooth nonlinear function. In this case, equation (3.2.2) can be rewritten as follows [73]:

$$E(f(x) - \hat{f}(x))^2 = \frac{H\sigma^2}{N} \quad (3.2.4)$$

There are many kinds of fixed basis function such as RBF, GP and SVM. These methods will be introduced in the following sections.

### 3.3 Radial basis function (RBF) networks

RBF network is the main practical alternative to the multi-layer perceptron (MLP) for nonlinear modelling. It achieves a smooth interpolation of scattered data in arbitrary dimensions and provides good approximations for multivariate functions [74]. RBF network is able to combine local representations of a multi-dimensional space by using restricted influence zones of the basis functions. Due to its nonlinear approximation properties, RBF is able to model complex arbitrary mappings. The architecture of an RBF network is simple and consists of one hidden layer which performs the nonlinear mapping. The activation of the hidden units in an RBF network is given by a nonlinear function of the distance between the input vector and a weight vector [75].

There are a number of significant differences between RBF networks and MLPs [43]:

- The RBF generally has a simple architecture consisting of two layers of weights, in which the first layer contains the parameters of the basis functions while the second layer forms linear combinations of the activations of

the basis functions to generate the outputs. Whereas, the MLP often has many layers of weights and a complex pattern of connectivity. In addition, a variety of different activation functions may be used within the same network.

- The RBF network is typically trained in two stages where the basis functions being determined first by unsupervised techniques then the second-layer weights subsequently being found by fast linear supervised methods. On the other hand, all of the parameters in the MLP are usually determined at the same time as part of a single global training strategy involving supervised training.
- The activation function of each hidden neuron in an RBF network computes the distance (usually by using the Euclidean norm) between the input vector and the centre of that neuron. On the other hand, the activation function of each hidden neuron in an MLP network computes the inner product of the input vector and the weight vector of that neuron.

### 3.3.1 Radial basis function structure

The RBF network consists of three layers (input, hidden, output). Figure 3.1 shows the basic structure of the RBF network. The hidden layer of the RBF network is nonlinear, while the output layer is linear.

Mathematically, the network output is the linear combination of the output for all hidden neurons which can be expressed for multi-input, multi-output network as follows:

$$y_k(x) = \sum_{i=1}^M w_{ki} \cdot \phi_i(\|x - c_i\|) + w_{k0} \quad (3.3.1)$$

where  $x$  is the input vector with elements  $x_d$  (where  $d$  is the dimension of the input vector),  $M$  is the number of nodes,  $\phi_i(\cdot)$  are radial basis functions,  $c_i$  is the vector determining the centre of the basis function  $\phi_i$ ,  $w_{ki}$  are the output layer weights,  $w_{k0}$  is the biases and the norm ( $\|\cdot\|$ ) is typically taken to be the Euclidean distance. The basis function  $\phi(\cdot)$  provides the nonlinearity and it will be discussed as follows.

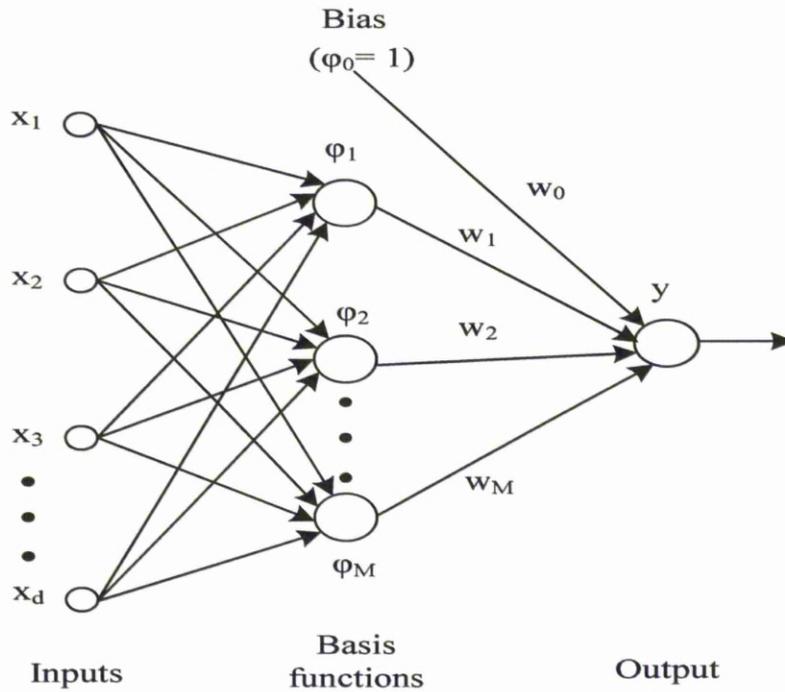


Figure 3.1: Architecture of a radial basis function network.

### 3.3.2 Basis functions

There are several basis functions, which are recognised as having useful properties for RBF networks. Typical choices for the basis functions are the following: [74]:

- Multiquadric:

$$\phi(r) = \sqrt{r^2 + \sigma^2} \quad (3.3.2)$$

- Gaussian:

$$\phi(r) = \exp\left(-\frac{r^2}{2\sigma^2}\right) \quad (3.3.3)$$

- Thin plate spline:

$$\phi(r) = r^2 \ln r \quad (3.3.4)$$

- Cubic:

$$\phi(r) = r^3 \quad (3.3.5)$$

where  $r = \|x - c_i\|$  and  $\sigma$  is the basis width or smoothing parameter. Gaussian basis function is local (gives a significant response only in a neighbourhood near the centre) with the property that  $\phi \rightarrow 0$  as  $|r| \rightarrow \infty$  and is more commonly used than other types which have a global response.

### 3.3.3 Radial basis function training

The training process can be used to obtain the RBF network's parameters. These parameters are the centre and the influence field of the radial function (the basis width,  $\sigma$ ) and the output weight (between the intermediate layers neurons and those of the output layer). Training an RBF which is considerably faster than the methods used to train MLP is accomplished in two stages. In the first stage, an unsupervised learning procedure is used to choose the basis function parameters, while in the second stage a supervised method is used to optimise the output layer weights. These two stages are discussed below.

#### First stage (Unsupervised)

This stage is unsupervised and its aim is to choose the basis function centres  $c_i$  and, where appropriate, the basis widths  $\sigma_i$ . The simplest procedure is to choose a subset of the data points at random and use them as the basis function centres. Another approach is to start with all data points as basis functions centres and then selectively remove centres in such a way as to have minimum disruption on the performance of the system [76].

The above techniques set the basis function centres only so that the widths of basis functions must be chosen using some other procedure. One heuristic approach is to choose all the widths to be equal and to be given by some multiple of the average distance between the basis function centres. This ensures that the basis functions overlap to some degree and hence give a relatively smooth representation of the distribution of training data. We might also recognise that the optimal width may be different for basis functions in different regions of input space [77].

A more principled approach for selecting a subset of the data points as basis function centres is based on the technique of orthogonal least squares. But

this approach would be computationally intensive since at each step a complete pseudo-inverse solution for each possible choice of basis functions would be necessary obtained .

A more sophisticated method is to divide the data into an appropriate number of clusters using clustering techniques and then to use the centres of these clusters for the basis function centres [75]. Moody *et al.* [78] use the  $K$ -means clustering algorithm in which the number  $k$  of centres must be pre-determined.

Given a dataset consists of  $N$   $d$ -dimensional input vector  $x$  in total, and we wish to find a set of  $k$  representative vectors  $c_i$  where  $i = 1, \dots, k$ . The algorithm seeks to partition the data points  $x$  into  $k$  disjoint subsets  $S_i$  containing  $N_i$  data points, in such a way as to minimise the sum-of-squares clustering function given by [77]:

$$J = \sum_{i=1}^k \sum_{z \in S_i} \|x_z - c_i\|^2 \quad (3.3.6)$$

Then, the centre of the basis function can be estimated as follows [77].

$$c_i = \frac{1}{N_i} \sum_{z \in S_i} x_z \quad (3.3.7)$$

When the centres have been established, the width of each basis function can be calculated [79].

$$\sigma_i = \frac{1}{N_i} \sum_{z \in S_i} (x_z - c_i)(x_z - c_i)^T \quad (3.3.8)$$

Another method used to find the parameters is expectation maximisation (EM) method [80]. This technique is based on the analogy between the RBF network and the Gaussian mixture models. The basis function centres are determined by fitting a Gaussian mixture model, while their widths are then set to be the largest squared distance between centres [79]. Also, it can be noted that the  $k$ -means algorithm can be seen as a particular limit of the EM optimisation of a Gaussian mixture model [77].

### Second stage (Supervised)

This stage is supervised and accomplished by solving a set of linear equations, the solution of which can be obtained by a matrix inversion technique such as

singular value decomposition or by least squares [81], to compute the output layer weights and bias.

In equation (3.3.1) the biases  $w_{k0}$  can be absorbed into the summation by including an extra basis function  $\phi_0$  whose activation is set to 1:

$$y_k(x) = \sum_{i=0}^M w_{ki} \cdot \phi_i(\|x - c_i\|) \quad (3.3.9)$$

This can be written in matrix notation as follows:

$$Y(x) = \Phi W \quad (3.3.10)$$

The weights can be optimised by minimisation the sum of square error between the actual output and the desired output of the network. Since the error function is a quadratic function of the weights, its minimum can be found in terms of the solution of a set of linear equations.

$$\Phi^T \Phi W = \Phi^T Y_d \quad (3.3.11)$$

where  $Y_d$  is the desired output (target) matrix.

If  $\Phi^T \Phi$  is square and non-singular, the optimal solution for the weights, with fixed basis functions, can be written as follows:

$$W = (\Phi^T \Phi)^{-1} \Phi^T Y_d \quad (3.3.12)$$

The pseudo-inverse can be used to avoid problems due to possible ill-conditioning of the matrix  $\Phi$ . So, equation (3.3.12) can be rewritten as follows:

$$W = \Phi^+ \Phi^T Y_d \quad (3.3.13)$$

where  $\Phi^+$  denotes the pseudo-inverse of  $\Phi^T \Phi$  which can be defined as follows:

$$\Phi^+ = \lim_{\nu \rightarrow 0} \Phi^T (\Phi^T \Phi + \nu I)^{-1} \quad (3.3.14)$$

where  $I$  is the unit matrix. Thus, the weights can be found by fast, linear matrix inversion techniques.

### 3.4 Support vector regression (SVR)

Support vector machine (SVM), which is proposed by Vapnik and his research co-workers [82, 83], is a novel powerful machine learning method based on statistical learning theory (SLT). SVM replaces the ERM principle which is generally employed in traditional artificial neural network, by structural risk minimisation (SRM) principle. The most important concept of SRM is the application of minimising an upper bound to the generalisation error instead of minimising the training error. Based on this principle, SVM will be equivalent to solving a linear constrained quadratic programming problem so that the solution of SVM is always unique and globally optimal.

Originally, SVM has been developed for solving the classification problems and achieved good performances [84, 85, 86, 87]. With the introduction of Vapnik's  $\varepsilon$ -insensitive loss function, SVM has been extended to solve the regression problems, called support vector regression (SVR) [88]. Recently, SVR has been applied to various applications with excellent performances [89, 90, 91]. The SVR has shown a high accuracy achieved when applied to solve the STLF problem [26, 92].

#### 3.4.1 The basic idea of SVR

Suppose there is a set of training data  $\{x_i, y_i\}_{i=1}^N$  where each  $x_i \in \mathbb{R}^d$  denotes the input space of the sample and has a corresponding target value  $y_i \in \mathbb{R}$  for  $i = 1, \dots, N$ , where  $N$  corresponds to the size of the training data.

SVR looks for an approximation function  $f(x)$  that has at most  $\varepsilon$  deviation from the targets for all the training data and is as flat as possible for good generalisation. This means that, we do not care about errors as long as they are less than the  $\varepsilon$  deviation.

Let the function  $f(x)$  is linear as follows:

$$f(x) = \langle w, x \rangle + b \quad (3.4.1)$$

where  $\langle \cdot, \cdot \rangle$  denotes the dot product,  $w$  contains the coefficients that have to be estimated from the data and  $b$  is a real constant.

The flatness of the function (3.4.1) implies seeking a small  $w$ , through minimising the square norm  $\|w\|^2$ . Minimising  $\|w\|^2 = \langle w, x \rangle$  is equivalent to maximising the distance between the data point and the approximation function [88].

The coefficients  $w$  and  $b$  can thus be estimated by minimising the regularised risk function [93].

$$R_{SVR} = R_{emp} + \frac{1}{2}\|w\|^2 = \frac{C}{N} \sum_{i=1}^N L_{\epsilon}(y_i, f(x_i)) + \frac{1}{2}\|w\|^2 \quad (3.4.2)$$

where  $R_{SVR}$  and  $R_{emp}$  represent the regression and empirical risks, respectively while  $L_{\epsilon}(y_i, f(x_i))$  is  $\epsilon$ -insensitive loss function.  $C$  is the regularisation constant which determines the trade-off between the flatness of  $f$  and its accuracy in capturing the training data.

In the regularized risk function given by equation (3.4.2), the regression risk (test set error),  $R_{SVR}$ , is the possible error committed by the function  $f$  in predicting the output corresponding to a new (test) example input vector. In equation (3.4.2), the first term  $\frac{C}{N} \sum_{i=1}^N L_{\epsilon}(y_i, f(x_i))$  denotes the empirical error (termed “training set error”), which is estimated by the  $\epsilon$ -insensitive loss function. The second item,  $\frac{1}{2}\|w\|^2$ , is the regularization term.

To estimate  $w$  and  $b$ , equation (3.4.2) is converted to the primal function by introducing slack variables  $\xi_i, \xi_i^*$ . Hence we have the following optimisation problem for SVR [82]:

$$\min_{w, b, \xi_i, \xi_i^*} \frac{1}{2}\|w\|^2 + C \sum_{i=1}^N (\xi_i + \xi_i^*) \quad (3.4.3)$$

$$\text{subject to } \begin{cases} y_i - \langle w, x_i \rangle - b & \leq \epsilon + \xi_i^* \\ \langle w, x_i \rangle + b - y_i & \leq \epsilon + \xi_i \\ \xi_i, \xi_i^* & \geq 0 \end{cases}$$

where  $\xi_i$  is the lower training error ( $\xi_i^*$  is the upper) subject to the  $\epsilon$ -insensitive tube. SVR avoids under-fitting and over-fitting of the training data by minimising the regularisation term  $\frac{1}{2}\|w\|^2$  as well as the training error  $C \sum_{i=1}^N (\xi_i + \xi_i^*)$ . The  $\epsilon$ -insensitive loss function which introduced by Vapnik [82] enforces the distance between the approximation function and the examples no more that  $\epsilon$ .

This loss function is shown in Fig. 3.2 in which the slope is determined by  $C$  and defined as follows:

$$|\xi|_\varepsilon = \begin{cases} 0 & \text{if } |\xi| < \varepsilon \\ |\xi| - \varepsilon & \text{otherwise} \end{cases} \quad (3.4.4)$$

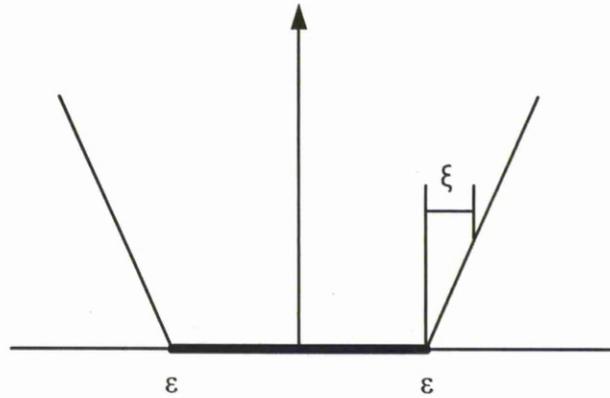


Figure 3.2: The  $\varepsilon$ -insensitive loss function.

### 3.4.2 Primal and dual optimisation

The Lagrange multipliers technique can be applied to solve equation (3.4.3) as follows [84]:

$$\begin{aligned} L = & \frac{1}{2} \|w\|^2 + C \sum_{i=1}^N (\xi_i + \xi_i^*) - \sum_{i=1}^N \alpha_i (\varepsilon + \xi_i - y_i + \langle w, x_i \rangle + b) \\ & - \sum_{i=1}^N \alpha_i^* (\varepsilon + \xi_i^* + y_i - \langle w, x_i \rangle - b) - \sum_{i=1}^N (\eta_i \xi_i + \eta_i^* \xi_i^*) \end{aligned} \quad (3.4.5)$$

Here  $L$  is the Lagrangian and  $\alpha_i, \alpha_i^*, \eta_i$  and  $\eta_i^*$  are Lagrange multipliers with:

$$\alpha_i, \alpha_i^*, \eta_i, \eta_i^* \geq 0. \quad (3.4.6)$$

Equation (3.4.5) is known as the primal objective function. The solution of this function is obtained by solving the dual objective function when the gradient of  $L$  with respect to  $w, b, \xi_i$  and  $\xi_i^*$  is equal to 0, therefore, we have

$$\frac{\partial L}{\partial b} = \sum_{i=1}^N (\alpha_i^* - \alpha_i) = 0 \quad (3.4.7)$$

$$\frac{\partial L}{\partial w} = w - \sum_{i=1}^N (\alpha_i - \alpha_i^*) x_i = 0 \quad (3.4.8)$$

$$\frac{\partial L}{\partial \xi_i} = C - \alpha_i - \eta_i = 0 \quad (3.4.9)$$

$$\frac{\partial L}{\partial \xi_i^*} = C - \alpha_i^* - \eta_i^* = 0 \quad (3.4.10)$$

By substituting equations (3.4.7)-(3.4.10) into (3.4.5), the dual optimisation function can be obtained as follows:

$$\max_{\alpha_i, \alpha_i^*} \begin{cases} -\frac{1}{2} \sum_{i,j=1}^N (\alpha_i - \alpha_i^*) (\alpha_j - \alpha_j^*) \langle x_i, x_j \rangle \\ -\varepsilon \sum_{i=1}^N (\alpha_i + \alpha_i^*) + \sum_{i=1}^l y_i (\alpha_i - \alpha_i^*) \end{cases} \quad (3.4.11)$$

$$\text{subject to } \begin{cases} \sum_{i=1}^N (\alpha_i - \alpha_i^*) = 0 \\ \alpha_i, \alpha_i^* \in [0, C] \end{cases}$$

In deriving equation (3.4.11) the dual variables  $\eta_i, \eta_i^*$  were eliminated through conditions (3.4.9) and (3.4.10) ( $\eta_i = C - \alpha_i$  and  $\eta_i^* = C - \alpha_i^*$ ), as these variables did not appear in the dual objective function anymore but only were presented in the dual feasibility conditions. Therefore, the support vector expansion comes from equation (3.4.8) which can be rewritten as follows [88]:

$$w = \sum_{i=1}^N (\alpha_i - \alpha_i^*) x_i \quad \text{thus} \quad \hat{f}(x) = \sum_{i=1}^N (\alpha_i - \alpha_i^*) \langle x_i, x \rangle + b \quad (3.4.12)$$

In order to calculate  $b$ , the Karush-Kuhn-Tucker (KKT) conditions can be used [94, 95]. These conditions will be stated in the SVR optimisation subsection. Allowing inequality constraints, the KKT approach to nonlinear programming generalises the method of Lagrange multipliers, which have allowed only equality constraints. The KKT conditions state that the product between dual variables and constraints has to vanish at the optimal solution.

$$\alpha_i (\varepsilon + \xi_i - y_i + \langle w, x_i \rangle + b) = 0 \quad (3.4.13)$$

$$\alpha_i^* (\varepsilon + \xi_i^* - y_i - \langle w, x_i \rangle - b) = 0 \quad (3.4.14)$$

and

$$(C - \alpha_i)\xi_i = 0 \quad (3.4.15)$$

$$(C - \alpha_i^*)\xi_i^* = 0 \quad (3.4.16)$$

Consequently, when  $\alpha_i, \alpha_i^* \in (0, C)$ , we have  $\xi_i, \xi_i^* = 0$ . This allows us to conclude that

$$b = y_i - \langle w, x_i \rangle - \varepsilon \quad \text{for } \alpha_i \in (0, C) \quad (3.4.17)$$

$$b = y_i - \langle w, x_i \rangle + \varepsilon \quad \text{for } \alpha_i^* \in (0, C) \quad (3.4.18)$$

From equation (3.4.13) and equation (3.4.14), the Lagrange multipliers are nonzero only if  $|\hat{f}(x_i) - y_i| > \varepsilon$ . Therefore we have a sparse expansion of  $w$  in terms of  $x_i$ . The examples correspond to the nonzero Lagrange multipliers are called support vectors (SV). The support vector expansion can be rewritten as follows:

$$\hat{f}(x) = \sum_{i=1}^N (\alpha_i - \alpha_i^*) \langle x_i, x \rangle + b = \sum_{i \in SV} (\alpha_i - \alpha_i^*) \langle x_i, x \rangle + b \quad (3.4.19)$$

The equivalence between SVR and sparse approximation has been pointed out by Girosi [96] where the same solution can be obtained from both SVR and sparse approximation by solving the same quadratic programming problem.

### 3.4.3 Nonlinear SVR

The next step is to make the support vector regression algorithm nonlinear. This could be achieved by using a nonlinear mapping ( $\phi$ ) to map the low dimensional input space into a high dimensional feature space ( $\mathcal{F}$ ) (Fig. 3.3) via a function,  $\phi : \mathbb{R}^d \rightarrow \mathcal{F}$ . Then the following estimate function is used to make linear regression in that feature space [65] as:

$$f(x) = \langle w, \phi(x) \rangle + b \quad (3.4.20)$$

where  $\phi(x)$  denotes the high dimensional feature space which is nonlinearly mapped from the input space,  $w$  contains the coefficients that have to be estimated from the data and  $b$  is a real constant.

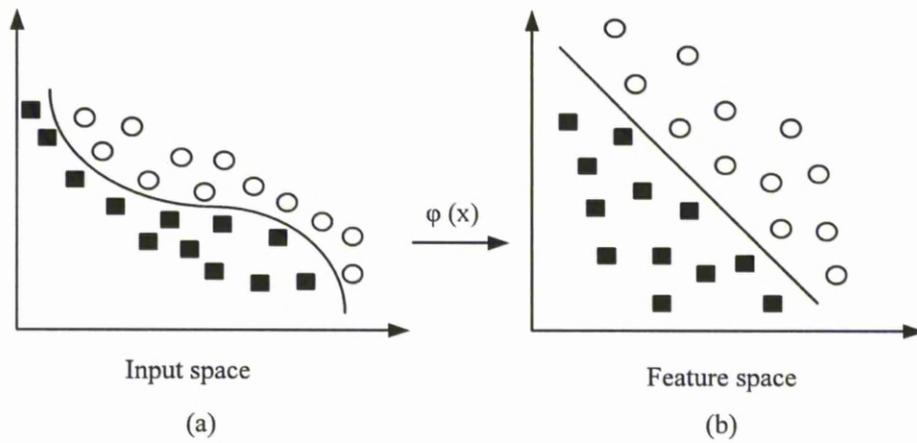


Figure 3.3: Transformation process illustration of a SVR model. A nonlinear mapping function  $\phi(x)$  defined to mapping a nonlinear problem in two dimensional input space (a) to linear problem in two dimensional feature space (b).

Figure 3.4 shows an example of a nonlinear regression function with an  $\varepsilon$ -insensitive loss function. The variables  $\xi_i, \xi_i^*$  measure the cost of the errors on the training points. These variables are equal to zero for all points inside the  $\varepsilon$  tube [65].

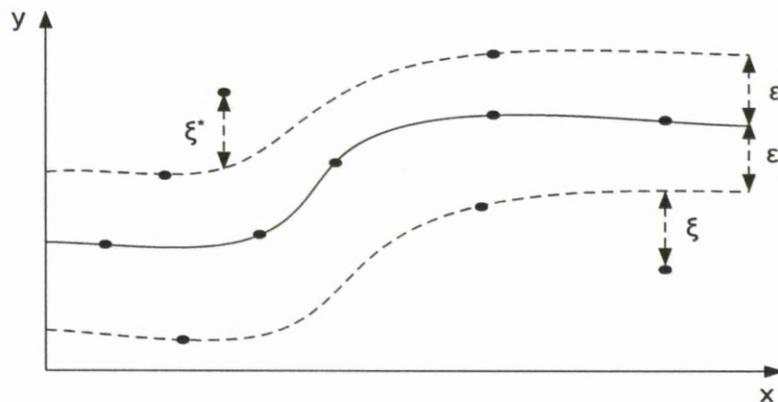


Figure 3.4: The  $\varepsilon$ -insensitive loss function for a nonlinear regression function. The solid line is the approximation function and the dashed line is the contour of the margin. The points lying on or outside the  $\varepsilon$  tube are support vectors.

The nonlinear SVR solution based on  $\varepsilon$ -insensitive loss function is given by [43]:

$$\begin{aligned} \max_{\alpha_i, \alpha_i^*} \left\{ \begin{array}{l} -\frac{1}{2} \sum_{i,j=1}^N (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) Q(x_i, x_j) \\ -\varepsilon \sum_{i=1}^N (\alpha_i + \alpha_i^*) + \sum_{i=1}^l y_i (\alpha_i - \alpha_i^*) \end{array} \right. \quad (3.4.21) \\ \text{subject to } \left\{ \begin{array}{l} \sum_{i=1}^N (\alpha_i - \alpha_i^*) = 0 \\ \alpha_i, \alpha_i^* \in [0, C] \end{array} \right. \end{aligned}$$

where  $Q(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle$  is the kernel function that is the inner product of the points  $\phi(x_i)$  and  $\phi(x_j)$  mapped into feature space.

The two parameters  $C$  and  $\varepsilon$  are free parameters selected by the user. The complexity of SVR method depends on these parameters, therefore they must be tuned simultaneously. The regression output takes the following form [43]:

$$\hat{f}(x) = \sum_{i=1}^N (\alpha_i - \alpha_i^*) Q(x_i, x) + b = \sum_{i \in SV} (\alpha_i - \alpha_i^*) Q(x_i, x) + b \quad (3.4.22)$$

The  $\varepsilon$ -insensitive loss function is attractive because unlike to quadratic and Huber cost functions, where all the data points will be support vectors, the algorithm solution can be sparse [97].

We can briefly review the basic properties of the SVR algorithm for regression as described so far. First, the input low dimensional input space (for which a prediction is to be made) is mapped into a high feature space by a map  $\phi$ . Then dot products are computed with the images of the training patterns under the map  $\phi$ . This equivalent to calculating kernel functions. Finally, the dot products are added up using the weights. This, plus the constant term  $b$  yields the final prediction output.

### 3.4.4 Kernel function

As stated, the nonlinear SVR utilises the fact that the kernel  $Q(x_i, x_j)$  becomes a dot product on the feature space  $\mathcal{F}$  in contrast to a dot product of the input space of the linear case. The dot product on the feature space is denoted as  $\langle \phi(x_i), \phi(x_j) \rangle_{\mathcal{F}}$ . The idea of the kernel function is to enable operations to be performed in the input space rather than the potentially high dimensional

feature space. Therefore, the inner product does not need to be evaluated in the feature space [97].

The following theorem of functional analysis which based upon Reproducing Kernel Hilbert Spaces [96, 98] shows that an inner product in feature space has an equivalent kernel in input space (equation 3.4.23) which provided certain conditions hold.

$$Q(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle \quad (3.4.23)$$

If  $Q$  is a symmetric positive definite function, which satisfies Mercers Conditions [82, 99],

$$Q(x_i, x_j) = \sum_{z=1}^{\infty} \alpha_z \phi_z(x_i) \phi_z(x_j), \quad \text{where } \alpha_z \geq 0 \quad (3.4.24)$$

and

$$\int \int Q(x_i, x_j) \psi(x_i) \psi(x_j) dx_i dx_j > 0, \quad \text{where } \int \psi^2(x) d(x) < \infty \quad (3.4.25)$$

then the kernel represents a legitimate inner product in feature space.

There are different types of kernel functions that satisfy Mercers conditions for SVR. They can be defined as follows [83, 97]:

- Linear kernel: The linear kernel is the simplest function of all kernel functions.

$$Q(x_i, x_j) = \langle x_i, x_j \rangle \quad (3.4.26)$$

- The Gaussian radial basis function kernel:

$$Q(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right) \quad (3.4.27)$$

- The polynomial kernel: A polynomial kernel of degree  $p$  is defined as:

$$Q(x_i, x_j) = (\langle x_i, x_j \rangle + 1)^p \quad (3.4.28)$$

- The hyperbolic tangent kernel:

$$Q(x_i, x_j) = \tanh(\text{scale} \langle x_i, x_j \rangle + \text{offset}) \quad (3.4.29)$$

- The Laplace radial basis function kernel:

$$Q(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|}{2\sigma^2}\right) \quad (3.4.30)$$

- The linear splines kernel:

$$\begin{aligned} Q(x_i, x_j) = & 1 + \langle x_i, x_j \rangle + \langle x_i, x_j \rangle \min(x_i, x_j) - \frac{x_i + x_j}{2} (\min(x_i, x_j))^2 \\ & + \frac{1}{3} (\min(x_i, x_j))^3 \end{aligned} \quad (3.4.31)$$

- The Additive kernels: More complicated kernels can be obtained by forming summing kernels, since the sum of two positive definite functions is positive definite.

$$Q(x_i, x_j) = \sum_z Q_z(x_i, x_j) \quad (3.4.32)$$

The Gaussian and Laplace radial basis function kernels are general-purpose kernels used when there is no prior knowledge about the data. Whereas, The linear kernel is useful when dealing with large sparse data vectors as is usually the case in text categorisation. In addition, the polynomial kernel is popular in image processing [100].

### 3.4.5 SVR optimisation

In the  $\varepsilon$ -insensitive loss function SVR algorithm, a large quadratic programming (QP) problem (equation 3.4.21) must be solved which gives a unique global minimum. This QP problem can be expressed in matrix notation as follows:

$$\min_{\alpha_i, \alpha_i^*} \frac{1}{2} \beta^T Q \beta + c^T \beta \quad (3.4.33)$$

$$\text{subject to } \begin{cases} A\beta = 0 \\ \alpha_i, \alpha_i^* \in [0, C], \quad i = 1, \dots, N \end{cases}$$

where

$$\beta = \begin{bmatrix} \alpha_i \\ \alpha_i^* \end{bmatrix}, \quad c = \begin{bmatrix} \varepsilon + y \\ \varepsilon - y \end{bmatrix} \quad (3.4.34)$$

$$Q = \begin{bmatrix} Q & -Q \\ -Q & Q \end{bmatrix}, \quad A = \underbrace{\{1, \dots, 1\}}_N, \underbrace{\{-1, \dots, -1\}}_N \quad (3.4.35)$$

$y = [y_1, \dots, y_N]^T$  and  $Q = Q(x_i, x_j)$  for  $i, j = 1, \dots, N$ .

According to Fletcher [101], when  $Q$  is a positive definite matrix and the constraints of equation (3.4.21) are linear, the solution of equation (3.4.21) must satisfy the KKT conditions as follows [94, 95]:

$$\begin{aligned} \beta_i = 0 &\Leftrightarrow y_i \hat{f}_i \geq 1, \\ 0 < \beta_i < C &\Leftrightarrow y_i \hat{f}_i = 1, \\ \beta_i = C &\Leftrightarrow y_i \hat{f}_i \leq 1, \end{aligned} \quad (3.4.36)$$

where  $\hat{f}_i = \hat{f}(x_i)$  is the output of the SVM for the  $i_{th}$  training example.

Two algorithms will be introduced to solve the large quadratic programming problem. They are Decomposition Algorithm (DA) [102] and Sequential Minimal Optimisation (SMO) [103, 88].

### Decomposition algorithm

The Decomposition Algorithm (DA) solves a sequence of small quadratic programming sub-problems instead of solving the large quadratic programming problem at once [102]. It is based on the observations that a sequence of quadratic programming sub-problems which at least always contains one example violating the KKT conditions will eventually converge to the optimal solution [104]. Osuna [104] suggested keeping a constant size matrix for every quadratic programming sub-problem, which implies adding and deleting the same number of examples in each iteration.

In the DA algorithm, the index of the training set is partitioned into two sets. The first one is called a working set ( $G$ ) while the second one is called a correcting set ( $E$ ). So,  $\beta, y, c$  and  $Q$  from equation (3.4.33) can be arranged properly as follows:

$$\beta = \begin{bmatrix} \beta_G \\ \beta_E \end{bmatrix}, \quad c = \begin{bmatrix} c_G \\ c_E \end{bmatrix}, \quad y = \begin{bmatrix} y_G \\ y_E \end{bmatrix}, \quad Q = \begin{bmatrix} Q_{GG} & Q_{GE} \\ Q_{EG} & Q_{EE} \end{bmatrix} \quad (3.4.37)$$

The dual objective function can be rewritten involving the two sets  $G$  and  $E$  as follows:

$$\min_{\alpha_i, \alpha_i^*} \frac{1}{2} \beta_G^T Q_{GG} \beta_G - \beta_G^T (c_G Q_{GE} \beta_E) \quad (3.4.38)$$

subject to  $\begin{cases} \langle y_G, \beta_G \rangle + \langle y_E, \beta_E \rangle = 0 \\ 0 \leq \beta_G \leq \mathbf{C} \end{cases}$

where  $\mathbf{C}$  is a column vector with all elements equal to  $C$ .

At each step,  $n$  elements exchange between set  $G$  and set  $E$  where at least one variable violating the KKT conditions is moved from  $E$  to  $G$ . Then the sub-problem (3.4.38) which involving the new working set is solved. The cycle repeats until no example violates the KKT conditions. Note that  $n$  is arbitrary.

### Sequential minimal optimisation

Sequential Minimal Optimisation (SMO) is a special case of the DA. The SMO was derived in [103] and applied to text categorisation problems. Then, Smola *et al.* [88], has generalised SMO for solving the regression problems.

The SMO is derived by taking the idea of DA to its extreme and optimising a minimal set of just two points at each iteration ( $n = 2$ ). The advantage of SMO comes from the fact that the optimisation problem for two data points admits an analytical solution, eliminating the need to use an iterative QP solver, which is hard to program, as a part of algorithm.

Unfortunately, keeping the size of working set equal to 2 leads to more sub-problems which need to be solved by using SMO, but each sub-problem can be solved very quickly due to the existence of analytical solutions. So, the overall training time of the SMO is less than the DA. In addition, the SMO does not require extra matrix storage due to the fact that the QP sub-problem can be solved analytically.

There are two main components of SMO [103]. They are the analytical solution for the sub-problem and a heuristic strategy for choosing which two examples are to be optimised which corresponds to exchanging two examples from  $G$  to  $E$  and reverse. The implementation of the SMO is straightforward. The details and pseudocode of the SMO for regression can be found in [88].

## 3.5 Gaussian process (GP)

The Gaussian process (GP) [105] has provided a promising nonlinear Bayesian approach particularly suited to regression problems since in these circumstances the posterior distribution of the parameters can be computed analytically [106, 27, 107]. The important advantage of GP over other non-Bayesian models is its explicit probabilistic formulation, which gives the ability to infer model parameters such as those that control the kernel shape and the noise level [75]. The Bayesian analysis of forecasting models is difficult because a simple prior distribution over parameters implies a complex prior distribution over functions. GP is flexible enough to represent a wide variety of interesting model structures, many of which would have a large number of parameters if they were formulated in more classical fashion [27]. In this section, the probabilistic approach to solve the regression problem is reviewed [63, 108].

### 3.5.1 Definition of Gaussian process

The idea of using Gaussian processes directly was inspired by investigations by Neal [109] into priors over weights for neural networks. Suppose there is a dataset  $D_N = \{x_i, y_i\}_{i=1}^N$  where each  $x_i \in \mathfrak{R}^d$  denotes the input space of the sample and has a corresponding target value  $y_i \in \mathfrak{R}$  for  $i = 1, \dots, N$ , where  $N$  corresponds to the number of data points. Consider a RBF network with fixed basis functions given by equation (3.3.1) where the weights are all given zero mean Gaussian priors ( $\mathcal{N}(0, \sigma_w^2 I)$ ) and the outputs are all given zero mean Gaussian noise.

In general, regression can be regarded as the conditional probability of the weights ( $w$ ) on the data (*i.e.*  $P(w|D_N)$ ). According to the Bayesian method, the posterior distribution of the weights is given by [75]:

$$P(w|D_N) = \frac{P(D_N|w)P(w)}{P(D_N)} \quad (3.5.1)$$

where  $P(D_N|w)$  is the maximum likelihood of the data,  $P(w)$  is the prior probability of  $w$  and  $P(D_N)$  is a normalisation constant  $P(D_N) = \int P(D_N|w)P(w)dw$ . Therefore we have,

$$P(w|D_N) \propto P(D_N|w)P(w) \quad (3.5.2)$$

When the distribution of the  $w$  is normal with zero mean, we have:

$$P(w) = \left( \frac{1}{2\pi\sigma^2} \right)^{\frac{-N}{2}} \exp \left( \frac{-\|w\|^2}{2\sigma^2} \right) \quad (3.5.3)$$

Substituting equation (3.5.3) into equation (3.5.2), we have [72]:

$$P(w|D_N) \propto \left( \frac{1}{2\pi\sigma^2} \right)^{\frac{-N}{2}} \exp \left( \frac{-(\|y_N - \hat{f}(X_N)\|^2 + \|w\|^2)}{2\sigma^2} \right) \quad (3.5.4)$$

where  $X_N = \{x_1^T, \dots, x_N^T\}$ ,  $y_N = \{y_1, \dots, y_N\}$  and  $\hat{f}$  is an approximation function with coefficients ( $w$ ). Therefore, the maximum of the logarithm posterior probability distribution is equivalent to maximising  $\|y_N - \hat{f}(X_N)\|^2 + \|w\|^2$ .

It is convenient to rewrite the problem (3.3.1) in matrix form as in equation (3.3.10), where the fixed basis function is given as:

$$\Phi_{ij} = \phi_j(x_i) \quad \text{for } i = 1, \dots, N \quad \text{and } j = 1, \dots, M \quad (3.5.5)$$

where  $\phi_j$  is the  $j$ th fixed basis function,  $N$  is the number of data points and  $M$  is the number of basis functions. Let  $y_d = \{y_{di}\}_{i=1}^N$  is the desired output (target) vector which are generated by Gaussian noise ( $\epsilon$ ) of variance ( $\sigma^2$ ) added to the model output ( $y_N$ ) and  $\epsilon \in \mathcal{N}(0, \sigma^2)$ . Then the distribution of  $y_d$  is normal with zero mean and the covariance matrix of  $y_d$  is [72]:

$$Q_N = \langle y_d^T y_d \rangle = \langle \Phi w w^T \Phi^T \rangle = \sigma_w^2 \Phi \Phi^T \quad (3.5.6)$$

From the above analysis, it can be seen that the probability distribution of  $y_N$  is:

$$P(y_d) = \mathcal{N}(0, Q_N + \sigma^2 I) = \mathcal{N}(0, C_N) \quad (3.5.7)$$

where  $C_N = Q_N + \sigma^2 I$ .

The GP employs a covariance matrix which is equivalent to using an infinite number of basis functions investigated in [109]. Therefore, the GP has a further merit compared with the MLP and RBF networks because these networks can only employ a limited number of neuron functions.

### 3.5.2 Predicting with Gaussian process

The prediction using the GP is specified by a conditional probability distribution  $P(y_{N+1}|x_{N+1}, D_N)$  of output  $y_{N+1}$  given a test input  $x_{N+1}$  and a set

of  $N$  training points ( $D_N$ ). The prediction of  $y_{N+1}$  is denoted by  $\hat{y}_{N+1}$  and is described by:

$$P(y_{N+1}|x_{N+1}, D_N) \propto \exp\left(-\frac{1}{2}(y_{N+1}C_{N+1}^{-1}y_{N+1} - y_N C_N^{-1}y_N)\right) \quad (3.5.8)$$

where  $C_{N+1}^{-1}$  is the inverse of the  $(N+1) \times (N+1)$  covariance matrix of  $y_{N+1}$  ( $y_{N+1} = \{y_1, \dots, y_N, y_{N+1}\}$ ).

The covariance matrix is always in the full rank due to the additive term  $\sigma^2 I$ . To overcome the computational time problem of calculating the inverse of  $(N+1) \times (N+1)$  matrix, the covariance matrix  $C_{N+1}$  and its inverse  $C_{N+1}^{-1}$  can be partitioned as follows [75]:

$$C_{N+1} = \begin{bmatrix} C_N & \eta \\ \eta^T & \gamma \end{bmatrix}, \quad C_{N+1}^{-1} = \begin{bmatrix} \tilde{C} & \zeta \\ \zeta^T & \kappa \end{bmatrix} \quad (3.5.9)$$

where  $C_N$  denotes the  $N \times N$  covariance matrix of the training data,  $\eta$  denotes the  $N \times 1$  covariance between the training data and  $y_{N+1}$  and  $\gamma$  denotes the variance of  $y_{N+1}$ .

According to the partitioned inverse equations, the elements of  $C_{N+1}^{-1}$  may be written as follows [110]:

$$\begin{aligned} \kappa &= (\gamma - \eta^T C_N^{-1} \eta)^{-1} \\ \zeta &= -\kappa C_N^{-1} \eta \\ \tilde{C} &= C_N^{-1} + \frac{1}{\kappa} \zeta \zeta^T \end{aligned} \quad (3.5.10)$$

Using equation (3.5.9) and equation (3.5.10), the prediction of the GP is in the recursive form:

$$P(y_{N+1}|x_{N+1}, D_N) = \frac{1}{Z} \exp\left(-\frac{(y_{N+1} - \hat{y}_{N+1})^2}{2\sigma_{\hat{y}_{N+1}}^2}\right) \quad (3.5.11)$$

and

$$\hat{y}_{N+1} = \eta C_N^{-1} y_N \quad \sigma_{\hat{y}_{N+1}}^2 = \gamma - \eta^T C_N^{-1} \eta \quad (3.5.12)$$

where  $Z$  is a normalised constant.

In contrast to classical methods, by using the GP, we obtain not only a point prediction but a predictive distribution. This advantage can be used to obtain the prediction intervals that describe a degree of belief of the predictions.

The covariance matrix of the GP, which is a kind of kernel function, only needs to be positive definite. Therefore, the GP approach is more flexible for selecting kernel function than SVR in which the kernel function should satisfy the Mercer condition. The advantage of the GP is that the covariance matrix being positive definite is not sufficient for the existence of the high dimensional feature space. Consequently, the solution with the covariance matrix will not give a sparse approximation to the target function.

### 3.5.3 Covariance function

The covariance function plays a key role to construct GP. It is chosen such that the correlation between the different training examples is expressed. In this thesis, the squared exponential covariance function is used which can be defined as follows [75]:

$$C(x^{(i)}, x^{(j)}) = \nu_0 \exp \left( -\frac{1}{2} \sum_{l=1}^d a_l (x_l^{(i)} - x_l^{(j)})^2 \right) + b \quad (3.5.13)$$

where  $d$  is the dimension of the input variables,  $a_l$ ,  $\nu_0$  and  $b$  are the hyper-parameters of the covariance function which are determined using the maximum likelihood method.  $b$  represents the bias that controls the vertical offset of the GP, while  $\nu_0$  controls the vertical scale of the process. The  $a_l$  parameters allow a different distance measure for each dimension. many other choices of covariance functions are available. They have been reviewed in [75, 111, 112].

## 3.6 Conclusions

In this chapter the basics of linear regression method and its limitations are discussed. These limitations can be overcome using various nonlinear regression methods. The curse of dimensionality which is the main practical problem of nonlinear regression estimation and how to overcome it are also discussed in this chapter. In addition, the mathematical foundation for the three nonlinear regression methods utilised in this thesis, *i.e.* RBF, SVR and GP, are introduced.

The implementation of SRM principle makes SVR superior to the others that employ ERM principle during the training, such as RBF. Moreover, in contrast

---

with RBF, the SVR's solution is always unique and global optimal. In addition, the GP method can achieve better performance than SVR in noisy environment. The reason is that the GP is based on Bayesian modelling and assumes that the parameters of the regression model are determined according to a probability distribution while the SVR is basically a point prediction method.

# Chapter 4

## Local Prediction Method for Short Term Load Forecasting

### 4.1 Introduction

Accurate forecasting of electricity load is one of the most important issues in the electricity industry. It is essential part of an efficient power system planning and operation. This chapter presents and compares three local predictors for STLF. They are based on the existing RBF networks, SVR and GP, and employ the coordinate delay method (CD) for time series analysis and data pre-processing. Local prediction makes use of similar historical data patterns in the reconstructed phase space to train the regression algorithm.

Three real world datasets are used to compare the performance of the proposed three local predictors with their global versions and simple two benchmark methods. It is demonstrated that the proposed local prediction framework significantly enhances the STLF accuracy.

The rest of this chapter is organised as follows: In Section 4.2 the basics of time series reconstruction are reviewed. Section 4.3 presents the local predictor framework for short term load forecasting. Experimental results obtained for STFL problems and their comparisons with other methods are presented in Section 4.4. Finally, this chapter is concluded in Section 4.5.

## 4.2 Phase space reconstruction of time series based on CD method

Nonlinear time series analysis and prediction has become a reliable tool for the study of complicated dynamical environments and measurements. Phase space reconstruction is the first step in nonlinear time series analysis.

With further scrutiny, one can notice the complexity of the historical load data and the uncertainty of the influencing factors such as weather, economical, and random factors. This encourages us to apply the time series phase space reconstruction method to the power load forecasting.

A commonly used phase space reconstruction method in the analysis of the nonlinear time series is the CD method which is based on the embedding theorem developed by Takens [113] and Sauer *et al.* [114]. The theorem looks at the one dimensional nonlinear time series as compressed information of higher dimension and in this way its features can be extracted by extending one dimensional time series to higher dimensional one.

Takens embedding theorem [113] gives theoretical foundation for analysis of time series which is generated by nonlinear deterministic dynamical systems. Then Sauer *et al.* [114] shows a phase space can be reconstructed from an univariate nonlinear time series.

For the univariate time series, the theorem regards an 1-dimensional time series  $x(t)$  for  $t = 1, 2, \dots, N$ , where  $N$  is the length of the dataset, as compressed higher dimensional information and, thus, its features can be extracted by extending  $x(t)$  to a vector  $X(t)$  in a  $d$ -dimensional space as follows:

$$X(t) = [x(t), x(t+m), x(t+2m), \dots, x(t+(d-1)m)]^T \quad (4.2.1)$$

where  $d$  is called the embedding dimension of the system and  $m$  is the delay constant. In order to obtain an appropriate model reconstruction, it is necessary to estimate  $d$  and  $m$ .

The phase space reconstruction gives not only a picture of the phase space trajectory in the embedded space similar to the actual trajectory of the true system, but preserves the original attractor's dynamics such as the Lyapunov

exponents of the attractor and geometrical invariants such as the eigenvalues of fixed points and the fractal dimension of an attractor.

In the following two subsections, a method for calculating the embedding dimension and time delay constant is reviewed.

### 4.2.1 Estimate the embedding dimension

Let a scalar time series is a projection from the  $d$ -dimensional state space of a system. The states of the original system are overlapped due to the projection from the states to the time series. A multi-dimensional reconstructed state space which is an image of the original system is defined to unfold the overlapped states of the original system from a given time series. According to Sauer's theorem [114], all overlapped states of the trajectory are eliminated when the embedding dimension  $d > 2d_0$ , where  $d_0$  is the box-counting dimension of the attractor.

In practice, there are several methods of estimating the embedding dimension such as correlation dimension method [115] singular value decomposition method [116] and false nearest neighbour method [117]. In this thesis, the correlation dimension method is used to determine the embedding dimension. It is the most popular method for determining the embedding dimension because of its computational simplicity.

The correlation dimension  $D_2$ , which is an estimation of the attractor dimension  $d_0$ , is defined in the following way. Firstly, the correlation integral is given by:

$$C(r) = \frac{2}{\widehat{N}(\widehat{N}-1)} \sum_{t=1}^{\widehat{N}-1} \sum_{p=t+1}^{\widehat{N}} \theta(r - \|x_t - x_p\|) \quad (4.2.2)$$

where  $\widehat{N} = N - (d-1)m$  is the number of embedded data,  $N$  is the number of data points,  $x_t, x_p$  are vectors in the multidimensional space,  $r$  is a radius within which distance between  $x_t$  and  $x_p$  are tested for proximity and  $\theta$  is the Heaviside step function:

$$\theta[x] = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{otherwise} \end{cases} \quad (4.2.3)$$

It is shown that [115] for sufficiently small  $r$  and sufficiently large  $N$ ,  $D_2$  can be

evaluated from the slop of the curve  $\ln C(r)$  versus  $\ln r$  in a linear scaling region as:

$$D_2 = \lim_{r \rightarrow 0} \frac{\ln C(r)}{\ln r} \quad (4.2.4)$$

From equation (4.2.4), we can notice that the correlation dimension depends on the size of  $r$ . When  $r$  is small, the behaviour of the correlation dimension is dominated by the characteristics of the noise, which has infinite dimension [118]. Suppose that  $D_2(r_{min}, r_{max})$  denote a range of correlation dimensions corresponding to a range of radii  $(r_{min}, r_{max})$ . The correlation dimension is determined such that  $D_2(r_{min}, r_{max})$  is a constant over a number of embedding dimensions exceeding  $[2D_2(r_{min}, r_{max}) + 1]$  [119].

### 4.2.2 Estimate the time delay constant

In order to use the coordinate delay method, it is necessary to choose the time delay constant ( $m$ ). A natural choice of  $m$  is the first minimum of the autocorrelation function such that each coordinate is linearly independent. In contrast to the autocorrelation function which measures only the linear dependence, the mutual information measures the general (linear and nonlinear) dependence of each coordinate. Therefore the mutual information provides a better criterion for the selection of  $m$ .

The mutual information from a time series proposed by Fraser *et al.* [120] can be defined as follows:

$$I_d(m) = dH_0 - H_d(m) \quad (4.2.5)$$

where

$$H_d(m) = \frac{1}{\widehat{N}} \sum_{i=1}^{\widehat{N}} \ln P_r(x_i(m)) \quad (4.2.6)$$

where  $d$  is the embedding dimension,  $\widehat{N} = N - (d - 1)m$  is the number of embedded data,  $N$  is the number of data points,  $P_r(x_i(m))$  is the probability to find other  $d$ -dimensional  $x_j$  within a sphere of radius  $r$  centred around the state vector  $x_i$  and  $H_0$  is equal to  $H_d$  when  $d = 1$ .

The Fraser's algorithm is difficult to implement and has been applied to a two-dimensional case only. So, a better algorithm has been proposed by Liebert

*et al.* [121] as follows: the first minimum of  $I_d(m)$  is equal to the first minimum of  $\ln(C_1^d)$ , where  $C_1^d = \lim_{q \rightarrow 1} C_q^d$  and  $C_q^d$  is in turn defined as:

$$C_q^d = \left( \frac{1}{\widehat{N}} \sum_{i=1}^{\widehat{N}} P_r^{q-1}(x_i) \right)^{\frac{1}{q-1}} \quad (4.2.7)$$

where  $\widehat{N}$  is the number of embedded data and  $P_r(x_i)$  is the probability to find other  $d$ -dimensional  $x_j$  within a sphere of radius  $r$  centred around the state vector  $x_i$  and can be defined as follows:

$$P_r(x_i) = \frac{1}{\widehat{N}} \sum_{j=1}^{\widehat{N}} \theta [r - |x_i - x_j|] \quad (4.2.8)$$

### 4.3 Local prediction method

In global predictors, a prediction model is trained based on the entire data history and used to predict the load at a specific time with a fixed data window. To overcome the drawbacks of the global predictors, the local predictors can be used [89].

In last few decades, the local predictor approach has interested many researchers to solve the nonlinear time series prediction problem such as [122, 123]. McNames, *et al.* [124] introduced the local averaging model for time series prediction. This method which can be used with smaller neighbourhoods, is more stable and often more accurate than local linear model for very short dataset. Therefore, this model was used in [124] to generate the winning entry of the K. U. Leuven time series prediction competition. Lau, *et al.* [89] combined the strength of SVR and local predictor. The proposed algorithm, gave a better prediction results than other local models when it is applied to nonlinear time series prediction.

Local prediction is concerned with predicting the future based only on a set of  $K$  nearest neighbours in the reconstructed embedded space without considering the historical instances which are distant and less relevant. Predictions of this kind are to establish a curve for the most recent data, and then make predictions based on the established curve. Local prediction constructs the true function by

subdivision of the function domain into many subsets (neighbourhoods). Therefore the dynamics of time series can be captured step by step locally in the phase space and the drawbacks of global methods can be overcome.

In general, the proposed algorithm consists of four stages. The first stage reconstructs the time series using the embedding dimension and the time delay constant. The second stage finds the  $K$  closest vectors, or nearest neighbours, of observed variables in the dataset for each query vector. The model is constructed in the third stage using only the  $K$  nearest neighbours, and the fourth stage evaluates the model using the query vector as the input to estimate the process output,  $\hat{y}$ .

These stages can be described in detail as follows:

- Stage 1: Upload a time series dataset  $S_N = \{x(t), t = 1, \dots, N\}$ , and set the parameters used for the prediction method, RBF, SVR or GP algorithms. Using the correlation dimension method and the mutual information method, calculate the embedding dimension ( $d$ ) and the time delay constant ( $m$ ) for the time series data set. Then, reconstruct the time series dataset  $\tilde{S}_N$  based on  $d$  and  $m$ .
- Stage 2: Choose the Euclidian distance as the distance metric in the phase space,  $d_E(x, q) = \sqrt{\sum_{j=1}^d [x(t_1 - (j-1)m) - q(t_2 - (j-1)m)]^2}$  between  $q$  (the query point) and each  $x$  in  $\tilde{S}_N$  (corresponding to two reconstructions of  $x(t_1)$  and  $x(t_2)$ ) and finding the  $K$  nearest neighbours  $\{x_q^1, x_q^2, \dots, x_q^K\}$ ;
- Stage 3: Regarding each neighbour  $\{x_q^l\}_{l=1}^K$  as a point in the domain and  $\{x(q_l + T)\}_{l=1}^K$  as the target value where  $T$  is the prediction step, and training the prediction method RBF, GP or SVR. For example training SVR to obtain support vectors and corresponding weight coefficients.
- Stage 4: Calculate the prediction value  $x(t+T)$  of the query point  $q$  based on the prediction method used. Then, the stages 2 to 4 can be repeated until the future values of different query vectors are all acquired.

Fig. 4.1 presents the computation procedure of the proposed method.

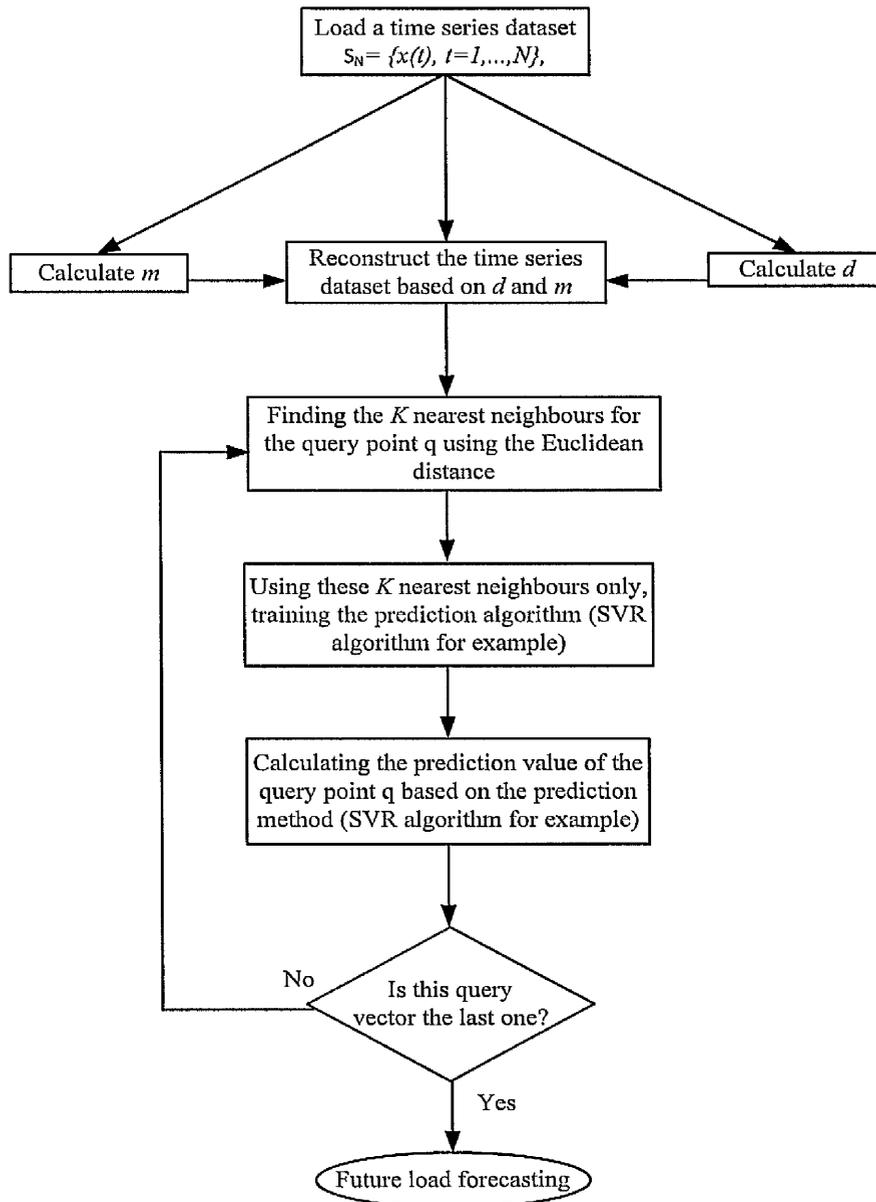


Figure 4.1: Flowchart of the local predictor.

Overall, the accuracy of the local predictor is better than the global predictor, because the accuracy of the latter is corrupted by distant patterns that have weak relationship to the current query. Another advantage of local predictor is that the training set for each point on the reconstructed trajectory is much smaller than the global predictor which requires use of all the available training examples, and it can save computational time and memory space through decomposing the prediction problem to several smaller ones.

Two important aspects should be concerned in the local predictor algorithm. The first one is how to choose suitable neighbour points. In this thesis, the Euclidean distance (as the most common metric in the literatures) is used to choose the nearest patterns by calculating the distance between each query vector and each vector in the training data vectors. The second is how long into the predicted series we can trust, in other words what is the number of the nearest neighbours. In general, the number of the nearest neighbours must be larger than the embedding dimension of the time series. However, if the number is too large, some far away points may be taken into account and this could reduce accuracy. So choosing  $K$  is very important step in order to establish the local predictor.

There are some methods used in literatures to find this parameter such as cross validation [125] and bootstrap [126]. This parameter should be high for low density datasets while it should be low for high density ones. So, in this thesis, we calculate  $K$  by designing a systematic method as follows:

$$K = \text{round} \left( \frac{\alpha}{N \times k_{\max} \times D_{\max}} \sum_{i=1}^N \sum_{k=1}^{k_{\max}} D_k(x_i) \right) \quad (4.3.1)$$

where  $N$  is the number of training points,  $k_{\max}$  is the maximum number of nearest neighbours,  $D_k(x_i)$  is the distance between each training point  $x$  and its nearest neighbours while  $D_{\max}$  is the maximum distance,  $\frac{1}{N \times k_{\max}} \sum_{i=1}^N \sum_{k=1}^{k_{\max}} D_k(x_i)$  is the average distance around the points which is inversely proportional to the local densities and  $\alpha$  is a constant. The two constants  $k_{\max}$  and  $\alpha$  are very low sensitivity parameters.  $k_{\max}$  can be chosen as a percentage of the number of training points ( $N$ ) for efficiency while  $\alpha$  can be chosen as a percentage.

## 4.4 Experimental results

### 4.4.1 Datasets

To evaluate the performance of the proposed local predictors and compare them with the corresponding global analogues, we used three different datasets from June 2005 to March 2006. The first two are half-hourly load datasets for England-Wales [127] and France [128] respectively, while the third is the hourly load dataset for Greece [129]. Two load types, namely the seasons of summer and winter, have been selected to validate the performance of the presented local predictors. For England-Wales and France 5,040 data and 4,032 data are used for summer and winter, respectively, whereas for Greece 2,520 data and 2,016 data are used for the two seasons. The training and testing periods as well as the number of training and testing samples for each load type of the three datasets are summarised in Table 4.1.

Certain characteristics can be reported about the datasets before evaluating our methods. For the England-Wales dataset, as one example from the three datasets, the load demand data are half-hourly recorded. Figure 4.2 illustrates the electricity demand of England-Wales from 1<sup>st</sup> July 2005 to 31<sup>st</sup> July 2005 for summer period, while Fig. 4.3 illustrates the electricity demand from 26<sup>th</sup> December 2005 to 25<sup>th</sup> January 2006 for winter period.

According to Figs. 4.2 and 4.3, the load has some seasonal patterns: the electricity demand in winter period is higher than the electricity demand in summer period. This pattern indicates the relation between the load usage and weather conditions in different seasons. Also, we can observe that the load has daily and weekly periodicity.

Load demand in weekdays (Monday through Friday) is usually higher than that of weekend. In addition, electricity demand in Sunday is a little lower than that on Saturday. Moreover, holidays also affect the load demand. On some major holidays such as Christmas or New Year, the electricity demand may be affected more compared with other holidays.

Table 4.1: Load data used for load forecasting

Load type	Set type	Period	Number of samples		
			England-Wales	France	Greece
Summer	Training data	06/06/2005 - 21/08/2005	3,696	3,696	1,848
	Testing data	22/08/2005 - 18/09/2005	1,344	1,344	672
Winter	Training data	26/12/2005 - 26/02/2006	3,024	3,024	1,512
	Testing data	27/02/2006 - 19/03/2006	1,008	1,008	504

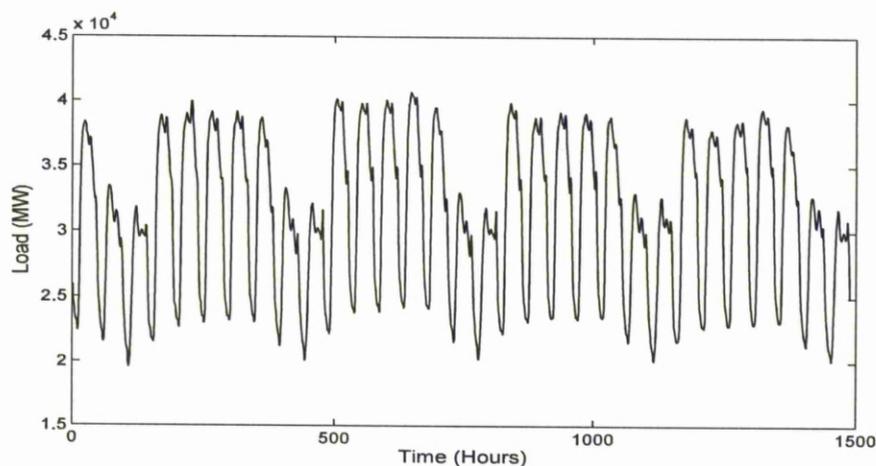


Figure 4.2: Half-hourly electricity demand in England-Wales from 1<sup>st</sup> July 2005 to 31<sup>st</sup> July 2005 for summer period.

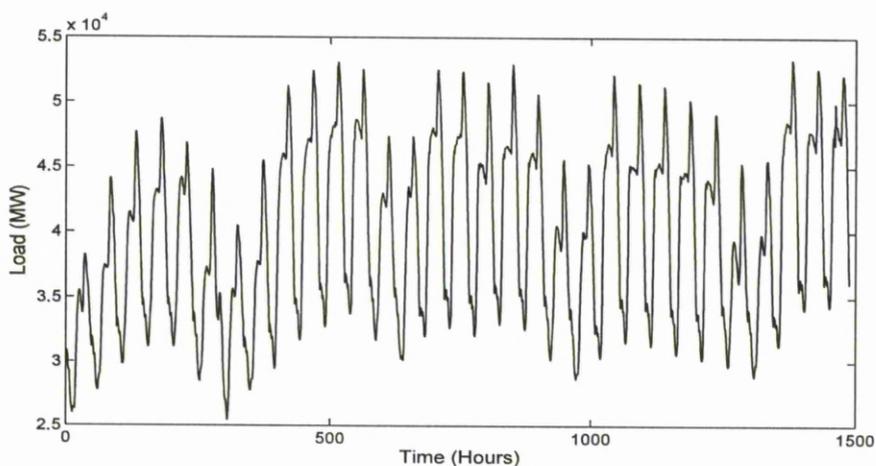


Figure 4.3: Half-hourly electricity demand in England-Wales from 26<sup>th</sup> December 2005 to 25<sup>th</sup> January 2006 for winter period.

#### 4.4.2 Parameters

To implement a good model, there are some important parameters to choose including embedding dimension  $d$ , time delay constant  $m$ , number of nearest neighbours  $K$  and the parameters of each algorithm (RBF, SVR or GP). Choosing the proper values of the  $d$  and  $m$  is a critical step in the algorithm. These parameters are calculated as described in Section 4.2.

The embedding dimension is the integer greater than or equal to  $2D_2 + 1$ , where  $D_2$  is the correlation dimension. Figures 4.4-4.6 show  $D_2$  for each dataset, where  $D_2$  can be evaluated from the slope of the curve  $\ln C(r)$  versus  $\ln r$  in a linear region.

From Figs. 4.4-4.6,  $D_2$  are 1.25, 1.57 and 1.44 for the England-Wales, France and Greece datasets, respectively. The optimal values of  $d$  and  $m$  for each dataset are shown in Table 4.2.

Table 4.2: Phase reconstruction parameters for each dataset

Data sets	Embedding dimension $d$	Time delay constant $m$
England-Wales	4	12
France	5	9
Greece	4	6

Also, choosing  $K$  is very important step in local prediction model. To calculate the best value of  $K$  for each dataset, equation (4.3.1) is used. In addition, the parameters  $k_{\max}$  and  $\alpha$  are fixed for all test cases in this Chapter at 30% of  $N$  and 60, respectively.

For all test cases, the Gaussian basis function (equation 3.3.3) is used for RBF network while the Gaussian radial basis function kernel (equation 3.4.27) is used for SVR. In addition, the squared exponential function (equation 3.5.13) is used for GP.

To train the RBF and GP algorithms, there are some parameters to choose, which, in order to get a good model, need to be selected carefully. To do this, the training data is divided into two subsets. One of them is used to train the model while the other, called the validation set, is used for select the model. According to their performance on the validation set, we infer the proper values of the parameters.

In addition, there are some key parameters for SVR, which are  $C$ ,  $\varepsilon$  and  $\sigma$  in the Gaussian kernel function. The selection of these parameters is important to the generalisation of the forecasting. Therefore, in order to get these parameters,

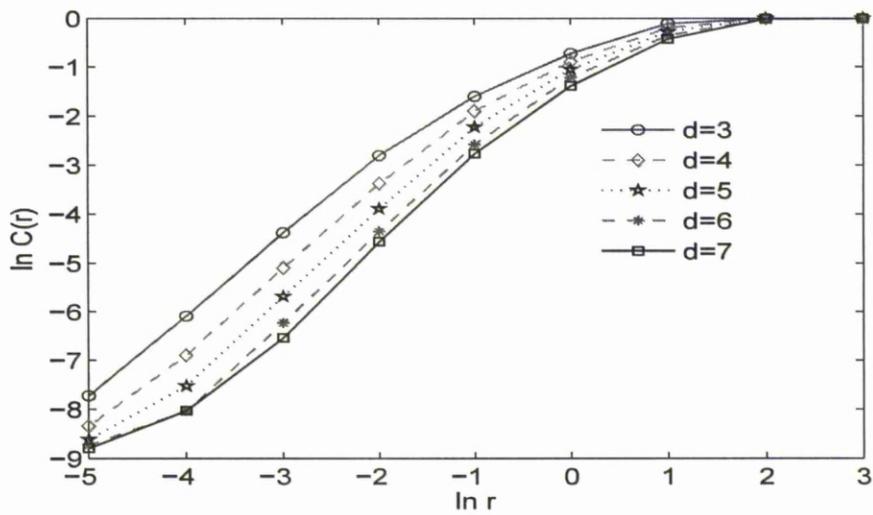


Figure 4.4: Correlation dimension for the England-Wales dataset.

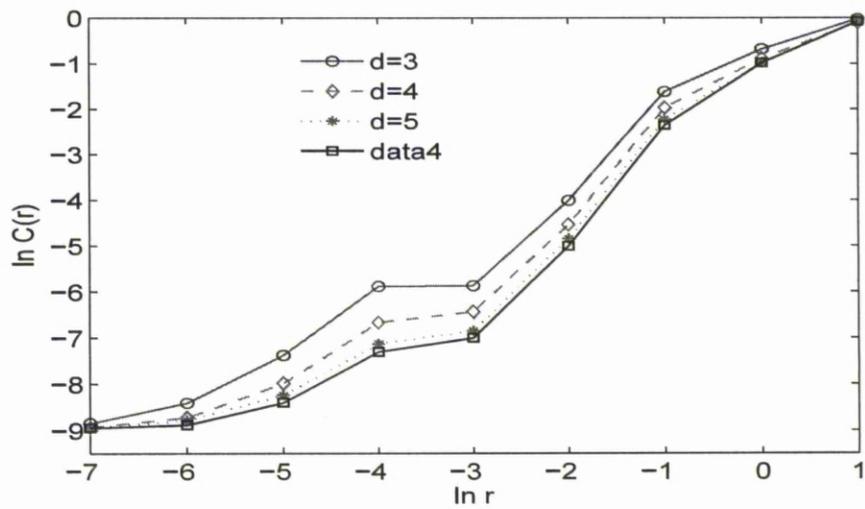


Figure 4.5: Correlation dimension for the France dataset.

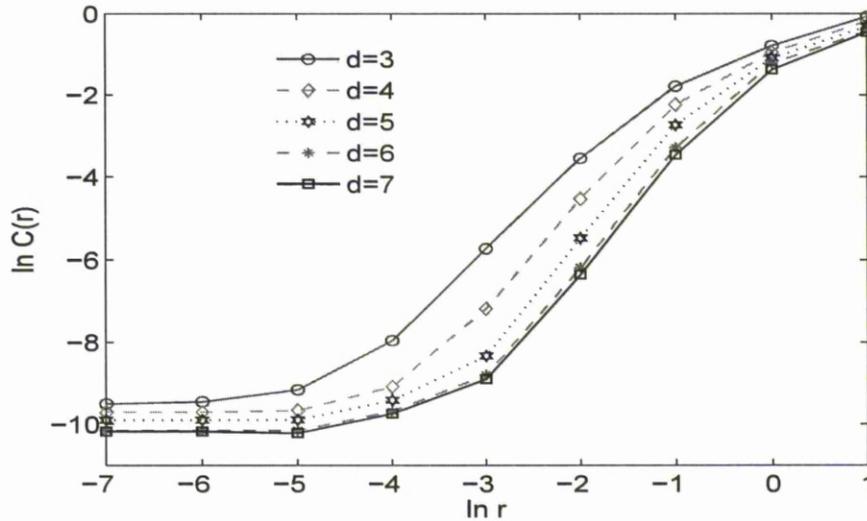


Figure 4.6: Correlation dimension for the Greece dataset.

the training data is divided into two subsets (training set and validation set). Based on this partition, the suitable parameters are chosen using the following procedures [130]:

- Set initial values of  $C$  and  $\varepsilon$ . Then, adjust the value of  $\sigma$  till a minimum validation error is achieved.
- Fix the value of  $\varepsilon$  and use the value of  $\sigma$  as calculated in the previous step. Then, adjust the value of  $C$  till a minimum validation error is achieved.
- Use the values of  $\sigma$  and  $C$  as calculated in the previous two steps. Then, adjust the value of  $\varepsilon$  till a minimum validation error is achieved.

Table 4.3 shows the number of RBF's centers ( $c$ ) and the values of SVR's key parameters for each dataset.

### 4.4.3 Results

To evaluate the performance of the proposed local predictors, numerical simulations comparing with their global versions and two benchmark methods are conducted. The two benchmark methods are Holt-Winters exponential smoothing and seasonal ARIMA model.

Table 4.3: The parameters used for each dataset

Data sets	Parameter	Summer	Winter
England-Wales	$c$	9	10
	$\sigma$	1.16	0.91
	$C$	35	20
France	$c$	10	10
	$\sigma$	1.05	1.21
	$C$	50	10
Greece	$c$	9	7
	$\sigma$	1	1.18
	$C$	25	18

The seasonal ARIMA model belongs to a family of flexible linear time series models that can be used to model many different types of seasonal as well as nonseasonal time series. Thus, each observation can be explained as a linear function of its past values, but with some errors. ARIMA models can also adjust for seasonality in the data, in which case the model is denoted by ARIMA  $(p, d, q) \times (P, D, Q)_S$ , where  $(p, d, q)$  are the nonseasonal autoregressive order, differencing order and moving average order, respectively and  $(P, D, Q)$  are the seasonal autoregressive order, differencing order and moving average order, respectively while  $S$  is the seasonal order.

Holt-Winters exponential smoothing model is exponential smoothing model that usually used for forecasting trend and seasonal time series. The standard Holt-Winters method is capable of handling a series with level, trend and a single cycle (seasonal pattern). The initial smoothed values for the level and seasonal components are estimated by averaging the early observations. The parameters are estimated in a single procedure by minimising the sum of squared one step-ahead in-sample errors. These parameters lie between zero and one. For the Holt-Winters exponential smoothing and the seasonal ARIMA model, the Minitab software package [131] is used in this thesis.

For all experiments, we quantified the prediction performance with mean absolute percentage error (MAPE) and normalised mean square error (NMSE). They can be defined as follows:

$$MAPE = \frac{1}{N} \sum_{i=1}^N \frac{|A_i - F_i|}{A_i} \times 100 \quad (4.4.1)$$

$$NMSE = \frac{1}{\Delta^2 N} \sum_{i=1}^N (A_i - F_i)^2, \quad (4.4.2)$$

$$\Delta^2 = \frac{1}{N-1} \sum_{i=1}^N (A_i - \bar{A})^2$$

where  $A_i$ ,  $F_i$  and  $\bar{A}$  are the actual value, the forecasted value and the mean of the actual values, respectively,  $N$  is the testing dataset size, and  $i$  denotes the test instance index.

To objectively compare the algorithms, the hourly load from 1 up to 24 hour (steps) ahead is predicted for the hourly dataset and the half-hourly load from 1 up to 48 half-hour (steps) ahead is predicted for the half-hourly datasets for each method for both load types. Tables 4.4-4.6 show the overall performance of each method for the entire testing period for both load types. These results are depicted in Figs. 4.7-4.12.

Table 4.4: Error of the entire testing period for the England-Wales dataset

Prediction method		Summer		Winter	
		MAPE	NMSE	MAPE	NMSE
Seasonal ARIMA		2.15	0.068	2.24	0.072
Holt-Winters Exp. Sm.		1.98	0.059	2.13	0.063
Global	RBF	1.95	0.057	2.09	0.060
	GP	1.69	0.045	1.82	0.053
	SVR	1.78	0.051	1.89	0.056
Local	RBF	1.58	0.043	1.69	0.046
	GP	1.26	0.031	1.37	0.037
	SVR	1.34	0.036	1.42	0.041

Table 4.5: Error of the entire testing period for the France dataset

Prediction method		Summer		Winter	
		MAPE	NMSE	MAPE	NMSE
Seasonal ARIMA		2.17	0.069	2.30	0.074
Holt-Winters Exp. Sm.		2.05	0.058	2.19	0.069
Global	RBF	2.06	0.059	2.21	0.070
	GP	1.75	0.050	1.91	0.057
	SVR	1.83	0.054	1.95	0.058
Local	RBF	1.73	0.047	1.86	0.056
	GP	1.43	0.041	1.52	0.043
	SVR	1.38	0.038	1.49	0.042

Table 4.6: Error of the entire testing period for the Greece dataset

Prediction method		Summer		Winter	
		MAPE	NMSE	MAPE	NMSE
Seasonal ARIMA		2.43	0.077	2.44	0.079
Holt-Winters Exp. Sm.		2.31	0.074	2.33	0.075
Global	RBF	2.28	0.072	2.27	0.071
	GP	1.97	0.058	2.00	0.062
	SVR	2.08	0.060	2.09	0.063
Local	RBF	1.90	0.055	1.94	0.058
	GP	1.46	0.042	1.48	0.043
	SVR	1.56	0.044	1.60	0.046

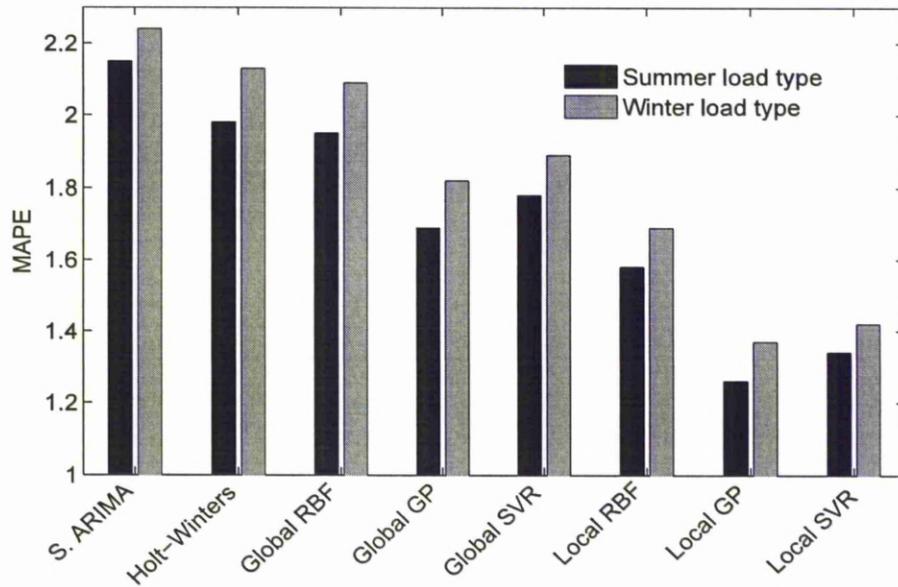


Figure 4.7: Comparison among all methods using MAPE for the England-Wales dataset.

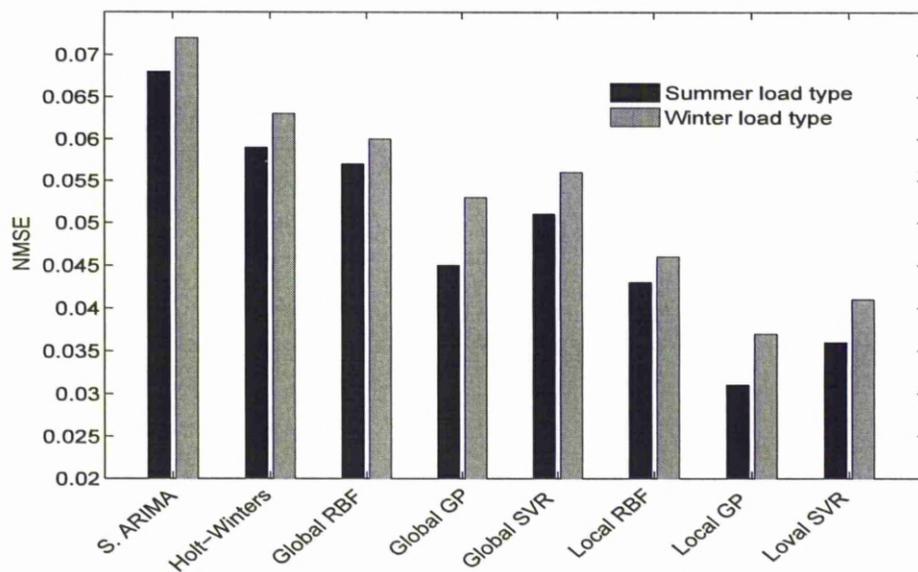


Figure 4.8: Comparison among all methods using NMSE for the England-Wales dataset.

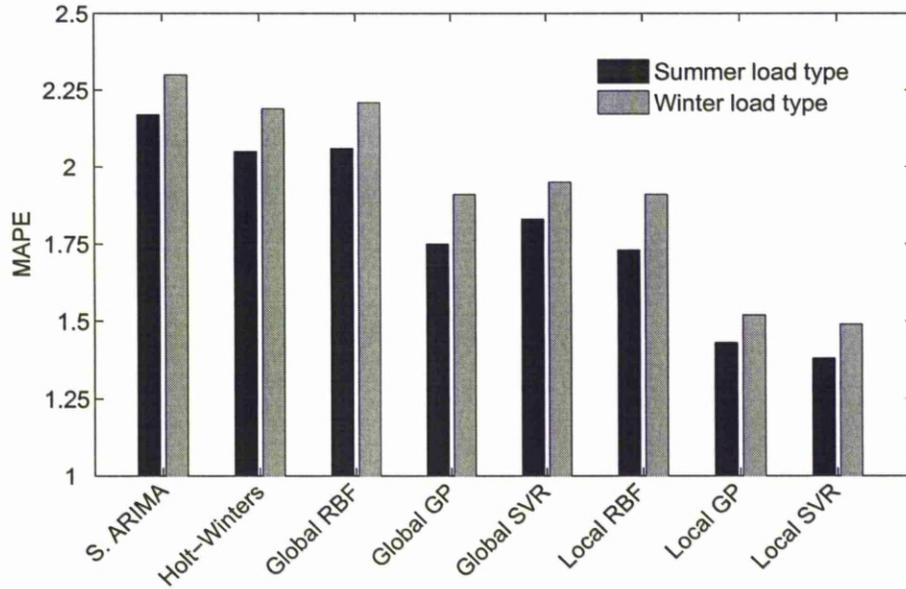


Figure 4.9: Comparison among all methods using MAPE for the France dataset.

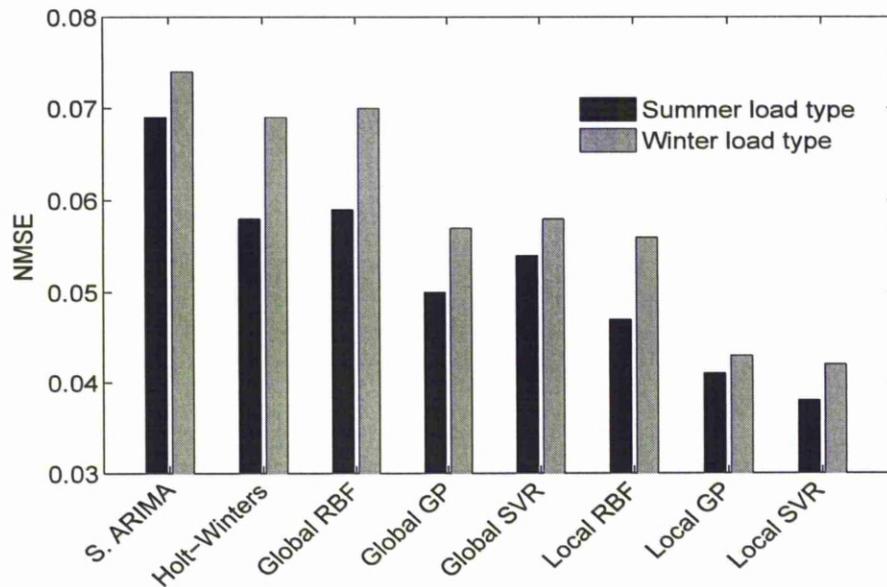


Figure 4.10: Comparison among all methods using NMSE for the France dataset.

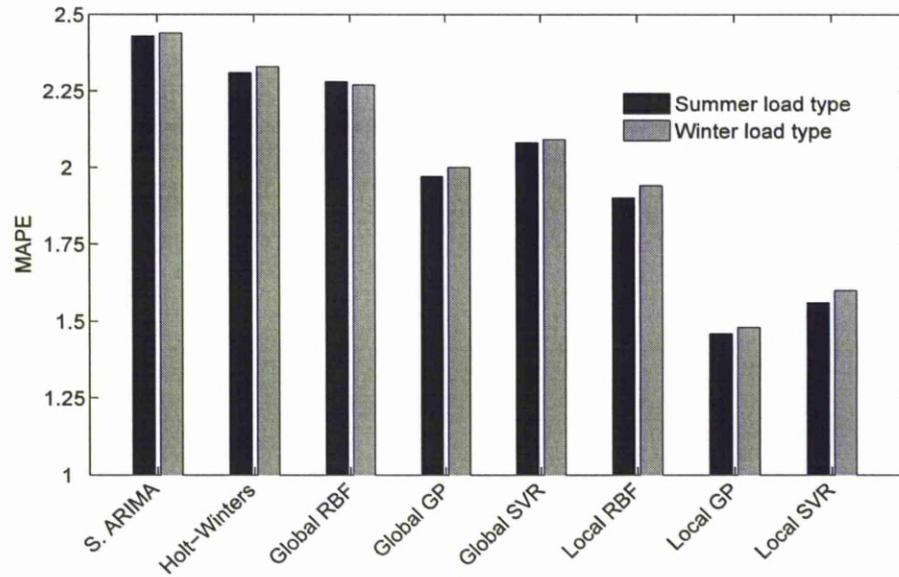


Figure 4.11: Comparison among all methods using MAPE for the Greece dataset.

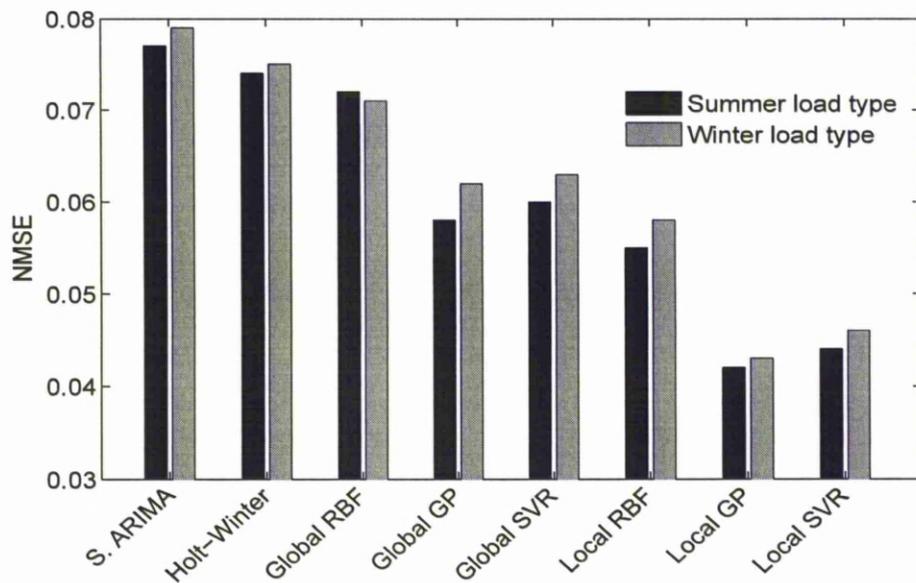


Figure 4.12: Comparison among all methods using NMSE for the Greece dataset.

It can be seen from these results that the local RBF, local GP and local SVR methods outperform their global versions, seasonal ARIMA and Holt-Winters exponential smoothing method for all datasets and for both load types. For the England-Wales dataset, the local RBF improved MAPE forecasting accuracy over the global RBF, seasonal ARIMA and Holt-Winters method by 18.97%, 26.51% and 20.20%, respectively for summer period and 19.14%, 24.55% and 20.66%, respectively for winter period. The local SVR improved MAPE forecasting accuracy over the global SVR, seasonal ARIMA and Holt-Winters exponential smoothing method by 24.72%, 37.67% and 32.32%, respectively for summer period and 24.87%, 36.61% and 33.33%, respectively for winter period. Also, the improvements of the local GP over the global GP, seasonal ARIMA and Holt-Winters exponential smoothing method were of 25.44%, 41.40% and 36.36%, respectively for summer period and 24.73%, 35.68% and 38.84%, respectively for winter period.

The results of France and Greece datasets also ascertain that the local RBF, local GP and local SVR methods give better performance than their global counterparts, seasonal ARIMA and Holt-Winters exponential smoothing for all cases. Moreover, the local GP gives the best performance amongst local methods for most cases. For summer period, the local GP improves the MAPE accuracy over the local RBF by 20.25%, 17.34%, and 23.16% for England-Wales, France and Greece, respectively. Also, it gives slightly better performance than the local SVR by 5.97% and 6.41% for England-Wales and Greece, respectively, while the local SVR gives slightly better performance than the local GP by 3.50% for France. The analysis of the winter period gives similar accuracy than the summer period analysis.

The results show that the performance of all methods in the summer period is better than their performance in winter periods for half-hourly load datasets, while they give almost the same performance for hourly load data set. This is probably owed to seasonality characteristics pertinent to each dataset.

Figs. 4.13-4.15 present one example for each dataset for both load types. They show the one day ahead forecasted load versus the actual load of 22/08/2005 and 27/02/2006 as an example of weekdays of summer and winter load types,

respectively and 27/08/2005 and 04/03/2006 as an example of weekends of summer and winter load types, respectively. These figures are drawn using the local predictor which gives the best performance of each dataset (local GP for England-Wales and Greece datasets and local SVR for France dataset). These results show that our forecasted values are very close to the actual values.

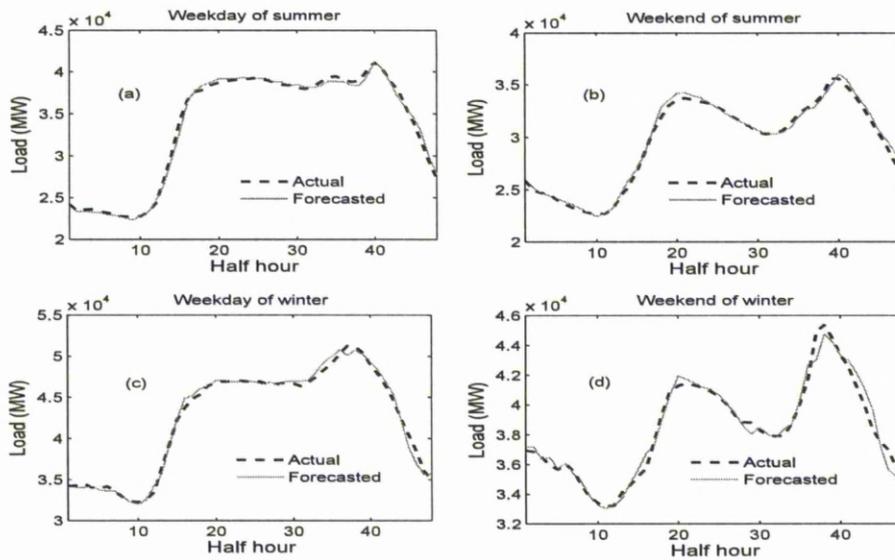


Figure 4.13: England-Wales' one day ahead forecasted and actual load of (a) 22/08/2005, (b) 27/08/2005, (c) 27/02/2006 and (d) 04/03/2006 using local GP method.

By calculating the MAPE for each method for both load types (summer and winter) at each lead time, then each method's performance is summarised by averaging these MAPE values across the three datasets. For the half-hourly dataset, MAPE values are calculated for 48 half-hour lead times, while, for the hourly dataset, the MAPE values are calculated for 24 hour lead times. By focusing only on the 24 hour lead times, these MAPE values are averaged across the three datasets for both load types and the resulting MAPE values are presented in Fig. 4.16.

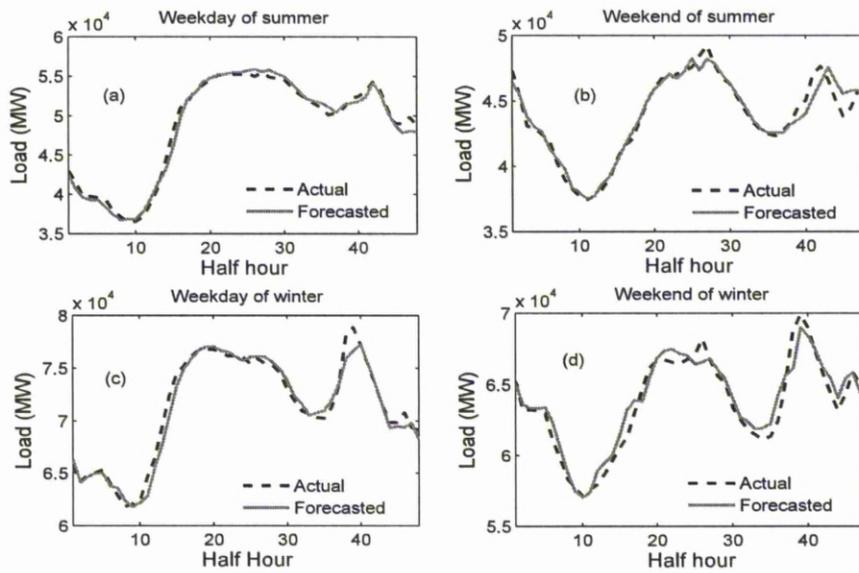


Figure 4.14: France's one day ahead forecasted and actual load of (a) 22/08/2005, (b) 27/08/2005, (c) 27/02/2006 and (d) 04/03/2006 using local SVR method.

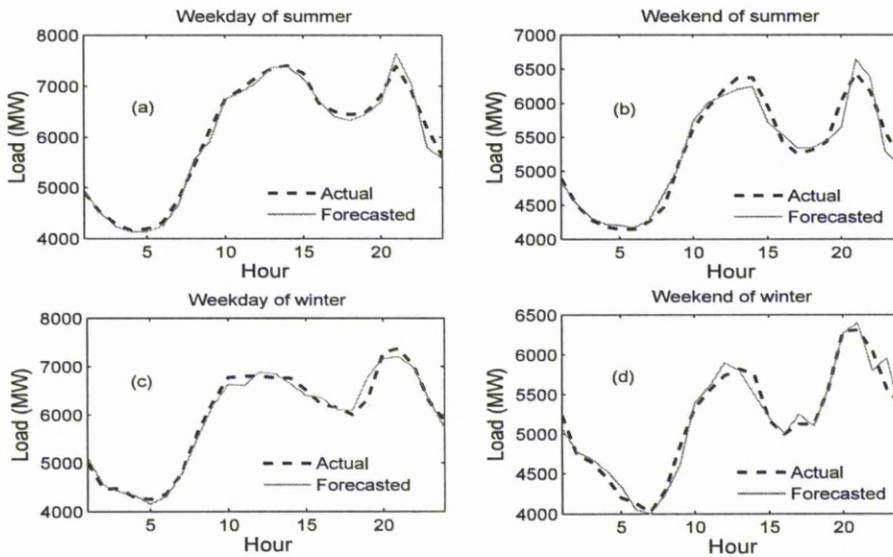


Figure 4.15: Greece's one day ahead forecasted and actual load of (a) 22/08/2005, (b) 27/08/2005, (c) 27/02/2006 and (d) 04/03/2006 using local GP method.

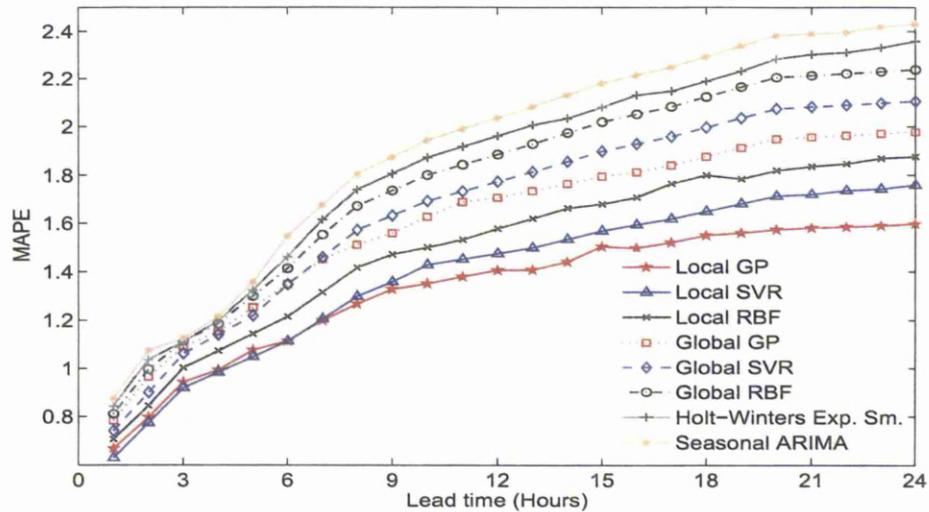


Figure 4.16: Mean MAPE plotted against lead time for the three datasets for both load types.

Fig. 4.16 shows that the local RBF, local GP and local SVR methods give better performance than their global counterparts, seasonal ARIMA and Holt-Winters exponential smoothing method for all lead times. The second principle observation is that the local GP gives the best performance followed by the local SVR and then the local RBF, however by zooming the first few hours in this figure as shown in Fig. 4.17, it can be noticed that the local SVR gives the best performance followed by the local GP and then the local RBF in the first few hours lead times.

The reason is that the GP is based on Bayesian modelling and assumes that the parameters of the regression model are determined according to a probability distribution (equation 3.5.11) while the SVR is basically a point prediction method (equation 3.4.22). Therefore, the local GP method can achieve better performance than local SVR in noisy environment. This explains why the local GP gives better performance than local SVR in a multi step ahead prediction as the noise is increasing with the increases of lead times.

Finally, as it is expected we can notice that the error is increasing proportionally with the lead time. This happens because uncertainty makes it more

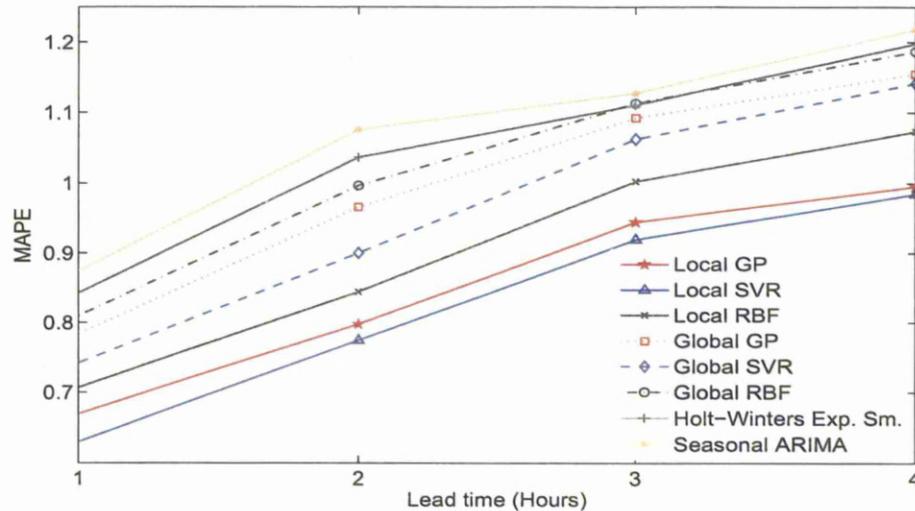


Figure 4.17: Mean MAPE plotted against lead time for the three datasets for both load types for the first few hours.

difficult to predict more distant events.

The MAPE of the whole testing period for each day during the week is summarised in Table 4.7 and depicted in Fig 4.18. First, the MAPE of each day during the testing period for each dataset for both load types is calculated. Then the average MAPE value across the three datasets of each day of the week (Monday to Sunday) during summer and winter testing periods can be calculated. The table also summarises in the last column the overall mean performance for each method. The improvements of our three local predictors over their global version, seasonal ARIMA and Holt-Winters exponential smoothing method are shown in Tables 4.8-4.10.

From these results, we can notice that the overall MAPE of the local methods is better than other methods. This confirms the superiority of the proposed local method over their global versions, seasonal ARIMA model and Holt-Winters exponential smoothing. In addition, the improvement in STLF's accuracy of local GP (in a range of 24.19%–38.43%) and local SVR (in a range of 24.2%–35.81%) over other methods is very worthy regarding to the literatures.

Table 4.7: MAPE of each day during testing period of the three datasets of both load types

Prediction method	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday	Overall Mean Error
Seasonal ARIMA	2.31	2.16	2.29	2.07	2.18	2.47	2.54	2.29
Holt-Winters Exp. Sm.	2.21	2.10	2.06	1.89	2.03	2.39	2.51	2.17
RBF	2.17	2.06	2.08	1.87	2.06	2.34	2.41	2.14
GP	1.79	1.71	1.81	1.62	1.77	2.09	2.23	1.86
SVR	1.98	1.78	1.89	1.77	1.75	2.15	2.24	1.94
RBF	1.77	1.59	1.74	1.57	1.64	2.10	2.14	1.79
GP	1.34	1.19	1.44	1.13	1.12	1.79	1.87	1.41
SVR	1.43	1.31	1.45	1.14	1.17	1.88	1.93	1.47

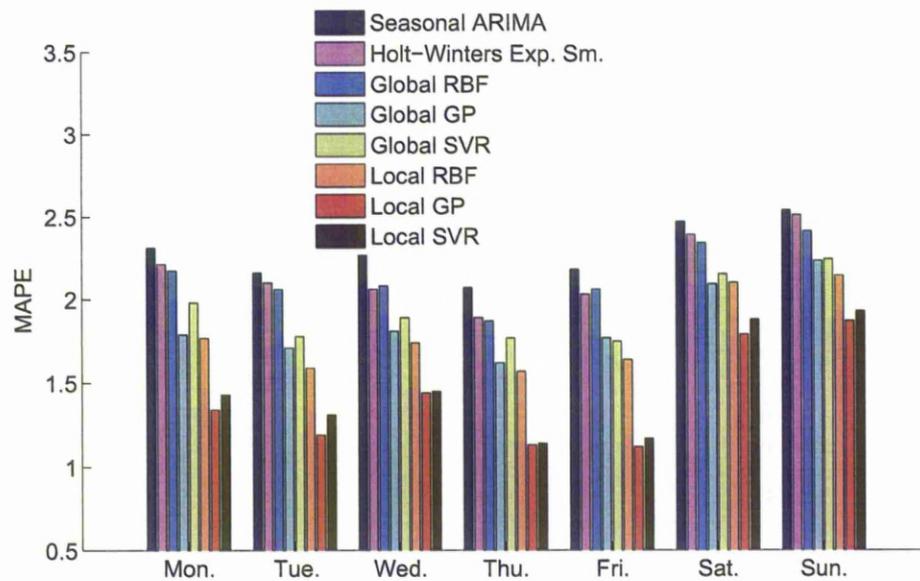


Figure 4.18: Average prediction MAPE of every day of the week for the three datasets for both load types.

Table 4.8: Improvement of the local RBF method over other methods

Prediction method	Overall MAPE	Improvement
Local RBF	1.79	—
Global RBF	2.14	16.36%
Holt-Winter Exp. Sm.	2.17	17.51%
Seasonal ARIMA	2.29	21.83%

Table 4.9: Improvement of the local SVR method over other methods

Prediction method	Overall MAPE	Improvement
Local SVR	1.47	—
Global SVR	1.94	24.23%
Holt-Winter Exp. Sm.	2.17	32.26%
Seasonal ARIMA	2.29	35.81%

Table 4.10: Improvement of the local GP method over other methods

Prediction method	Overall MAPE	Improvement
Local GP	1.41	—
Global GP	1.86	24.19%
Holt-Winter Exp. Sm.	2.17	35.02%
Seasonal ARIMA	2.29	38.43%

## 4.5 Conclusions

In this chapter, the local prediction method has been extended using RBF, SVR and GP. The presented three methods start with the data preprocessing where the CD method is used to calculate the embedding dimension and the time delay constant for the power load data. Then the continuous power load data are used for the phase space reconstruction. After that, the local model can be set up using only the neighbouring points of the point under prediction.

The experimental cases with prediction up to day ahead have employed to consistently confirm the superiority of the local predictors over the global ones, seasonal ARIMA and Holt-Winters methods. Moreover, amongst the three proposed local predictors, the local GP gives better performance than local RBF in all test cases. Also, it gives slightly better performance in most cases in comparison with the local SVR, especially in noisy environment. The local SVR and local GP can achieve 1.47% and 1.41% MAPE level, respectively for one day-ahead load forecasting for all three datasets. These obtained accuracy is very good for the practical application in comparison with published literature.

## Chapter 5

# Short Term Load Forecasting Based on Locally Weighted Support Vector Regression

### 5.1 Introduction

In the SVR algorithm, the regularisation parameter is constant so that all training data contribute to the accuracy of the model to the same extent. However, in many cases, the effects of the training points are different. Therefore, each training data should have a weight according to its importance.

To achieve this goal, some trials are made to modify the standard SVR by weighting the SVR's regularisation parameter. Tay *et al.* [132] introduced a modified support vector machine (SVM) for financial time series forecasting which is called C-ascending SVM. In this method, each data is weighted using one of two designed weighting functions. They are linear weight function and exponential weight function. Then, the recent historical data points have larger weights than the distant historical data. Lee *et al.* [133] proposed the weighted SVM for quality estimation in the polymerisation process. The proposed method combines the support vector machine and locally weighted regression (LWR). Each data is weighted according to its distance to the current prediction point. Hu *et al.* [134] introduced the weighted SVM based fuzzy C-mean clustering algorithm to solve STLF problem. In this approach, the training samples are

clustered into several subsets with consideration of homogenous characteristics. In addition, according to the time, each data is weighted. The older data points have smaller weights than the new ones. Unfortunately, the above trials were used as global predictors except [133].

In this chapter, a modified version of the support vector regression is presented to solve the load forecasting problem. The proposed model is derived by modifying the risk function of the support vector regression algorithm with use of locally weighted regression while keeping the regularisation term in its original form. In addition, instead of using the weighting function's bandwidth as a fixed value, the weighted distance algorithm based on the Mahalanobis distance is proposed to optimise this bandwidth. This leads to improve the accuracy of the algorithm. The performance of the new method is evaluated with two real world datasets and compared with the local support vector regression and some published methods using the same datasets. The results show that the proposed model exhibits superior performance to that of other methods.

The rest of this chapter is organised as follows: In Section 5.2 the basics of multivariate time series reconstruction are reviewed. Section 5.3 outlines the basics of locally weighted regression method. Section 5.4 presents the proposed locally weighted support vector regression. Then The weighted distance algorithm is presented in Section 5.5. Experimental results obtained for STFL problems and their comparisons with other methods are presented in Section 5.6. Finally, Section 5.7 concludes this Chapter of the work.

## 5.2 Phase space reconstruction of multivariate time series

The first step in the local predictor is time series reconstruction. In Section 4.2 the basics of reconstruction univariate time series based on CD method is discussed, while this section reviews the reconstruction of the multivariate time series based on CD method.

For the multivariate time series, assuming there are  $n$  time series, they are  $\{x_i(t)\}$ , ( $i = 1, 2, \dots, n$  and  $t = 1, \dots, N$ ) where  $N$  is the length of the dataset.

According to the embedding theorem developed by Takens [113] and Sauer *et al.* [114], the reconstructed vector of multivariate time series in the phase space could be denoted as [135]:

$$z_i(t) = [x_i(t), x_i(t + m_i), x_i(t + 2m_i), \dots, x_i(t + (d_i - 1)m_i)] \quad (5.2.1)$$

where  $i = 1, 2, \dots, n$ ;  $t = 1, 2, \dots, L$ .  $d_i$  and  $m_i$  are the selected embedding dimension and time delay constant of the  $i^{\text{th}}$  time series, respectively.  $L$  is the length of the embedded points generated in the phase space which can be computed using this formula,  $L = N - \max_{i=1, \dots, n} [(d_i - 1)m_i]$ .  $z(t)$  is now a  $L \times D_t$  matrix and  $D_t = \sum_{i=1}^n d_i$ . The details of how to choose the proper values of  $d$  and  $m$  using the correlation dimension method and mutual information method have been discussed in subsections 4.2.1 and 4.2.2, respectively.

### 5.3 Locally weighted regression (LWR)

LWR [136] is a kind of locally weighted learning methods. LWR forms a local model around a point of interest whereby only training data that is closest to that point will be used in handling each query, instead of using all training data [137]. After answering the query the aforesaid local model is discarded. To answer a new query, a new local model is created, which means that every set of training and generation period is unique and independent to others. LWR is a method for estimating a regression surface through multivariate smoothing: the response variable is smoothed dynamically, as a function of the predictor variables [136]. LWR consists of developing a moving local model to a set of nearest neighbours.

Locally weighted regression is derived from standard linear regression. This algorithm fits a surface to “local” points using distance-weighted regression. LWR is based on the (assumption) that the neighbouring values of the predictor variables are the best indicators of the response variable in that range of predictor values [136].

To estimate the value of the function  $\hat{f}(q)$  at any value of query vector ( $q$ ) in the  $d$ -dimensional space ( $q = x_q \in \mathbb{R}^d$ ), the  $K$  (neighbourhood size) data points whose  $x_i$  values are closest to  $q$  are used ( $1 < K \ll N$  and  $N$  is the number

of training points). Each point in the neighbourhood is weighted according to its distance from  $q$ . The Euclidean distance is usually used to calculate these distances. The points that are close to  $q$  have large weights, and the points far from  $q$  have small weights [133].

Suppose we have the input data  $X = \{x_i\}_{i=1}^N$  where each  $x_i \in \mathfrak{R}^d$  and the output values  $y = \{y_i\}_{i=1}^N$  where each  $y_i \in \mathfrak{R}$  and  $N$  is the number of data points.

Each row of  $X$  and  $y$  is multiplied by the corresponding weight  $w_i$ . Thus:

$$U = WX \quad (5.3.1)$$

$$V = Wy \quad (5.3.2)$$

where  $W$  is a diagonal matrix with diagonal elements  $W_{ii} = \sqrt{w_i}$  and zeros elsewhere.

For the linear regression which can be expressed as follows (It will be assumed that the constant 1 has been appended to all the input vectors to include a constant term in the regression):

$$U\beta = V \quad (5.3.3)$$

where  $\beta = [\beta_1, \beta_2, \dots, \beta_d]^T$  are the unknown parameters to be estimated from the data. Equation (5.3.3) is solved for  $\beta$  as follows [137]:

$$U^T U \beta = U^T V \quad (5.3.4)$$

and

$$\beta = (U^T U)^{-1} U^T V \quad (5.3.5)$$

Formally, this gives us an estimator for each query point of the form:

$$\hat{y} = \hat{f}(q) = q^T (U^T U)^{-1} U^T V \quad (5.3.6)$$

### 5.3.1 Weighting functions

The weighting function should have a maximum value at zero distance while it should decay smoothly as the distance increase. Discontinuities in weighting

functions lead to discontinuities in the predictions since training points cross the discontinuity as the query changes. The weighting functions that go to zero at a finite distance allow faster implementations, since points further from the query than that distance can be ignored with no error whereas the infinite weighting functions allow exact interpolation of the stored data [137].

Many weighting functions are proposed by the researchers [136, 138]. The most commonly used weighting functions are:

- Gaussian weighting function: This function has infinite extent and can be truncated when it becomes smaller than a threshold value to ignore data further from a particular radius from the query.

$$W(d_E) = e^{-\left(\frac{d_E}{h}\right)^2} \quad (5.3.7)$$

where  $h$  is the bandwidth parameter which plays an important role in local modelling. The optimisation of this parameter will be discussed in Section 5.5. This weighting function is used in this thesis.

- Quadratic weighting function: This function has finite extent where it ignores the data further than a radius of 1 from the query.

$$W(d_E) = \begin{cases} (1 - d_E^2) & \text{if } |d_E| < 1 \\ 0 & \text{otherwise} \end{cases} \quad (5.3.8)$$

- Tricube weighting function: This function has also finite extent.

$$W(d_E) = \begin{cases} (1 - d_E^3)^3 & \text{if } |d_E| < 1 \\ 0 & \text{otherwise} \end{cases} \quad (5.3.9)$$

The weight of the data point  $(x_i, y_i)$  then is:

$$w_i = W(d_E(q, x_i)) = W\left(\sqrt{(q - x_i)^T(q - x_i)}\right) \quad (5.3.10)$$

Thus  $w_i$  has its maximum value when  $x_i$  is closest to  $q$ , and decreases as  $x_i$  increases in distance from  $q$ .

## 5.4 Locally weighted support vector regression (LWSVR)

The work presented in this section extends our previous work in Section 4.3 by combining the support vector regression and locally weighted regression, which can be called as locally weighted support vector regression (LWSVR). LWSVR is an ameliorated SVR, which endows a weight factor to each train load datum. In the LWSVR, the regularisation constant  $C$  in equation (3.4.3) is computed as a function of the distance between input data points, and the concept of LWR is used. Therefore, each point in the neighbourhood is weighted according to its distance from the current point under prediction. The points that are close to the current point under prediction have larger weights than others.

In the presented approach, the modified risk function can be formulated as follows:

$$\frac{1}{2} \|w\|^2 + \sum_{i=1}^N C_i (\xi_i + \xi_i^*) \quad (5.4.1)$$

and

$$C_i = W_i \times C \quad (5.4.2)$$

where  $W_i$  is the weighting function. Replacing the constant  $C$  in equation (3.4.21) using equation (5.4.2), the dual problem can be written as:

$$\max_{\alpha_i, \alpha_i^*} \begin{cases} -\frac{1}{2} \sum_{i,j=1}^N (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) Q(x_i, x_j) \\ -\varepsilon \sum_{i=1}^N (\alpha_i + \alpha_i^*) + \sum_{i=1}^l y_i (\alpha_i - \alpha_i^*) \end{cases} \quad (5.4.3)$$

$$\text{subject to } \begin{cases} \sum_{i=1}^N (\alpha_i - \alpha_i^*) = 0 \\ \alpha_i, \alpha_i^* \in [0, C_i] \end{cases}$$

By solving this problem, the regression output can be obtained using equation (3.4.22).

### 5.4.1 LWSVR conceptual interpretation

As shown in Fig. 5.1, the standard SVR with fixed  $C$  (shown by dashed line in Fig. 5.1) tries to track all training data with a specific model complexity.

Therefore, the size of the forecasting error ( $\xi_i, \xi_i^*$ ) does not vary greatly. While the LWSVR (shown by solid line in Fig. 5.1) applies a heavy penalty to the points near the query point in an attempt to reduce such errors. LWSVR is expected to give higher forecasting accuracy, because as the shape of the weighting function becomes sharper, the forecasting error around the new input data are expected to decrease. In contrast to standard SVR, the model is retrained when the location of the query point moves.

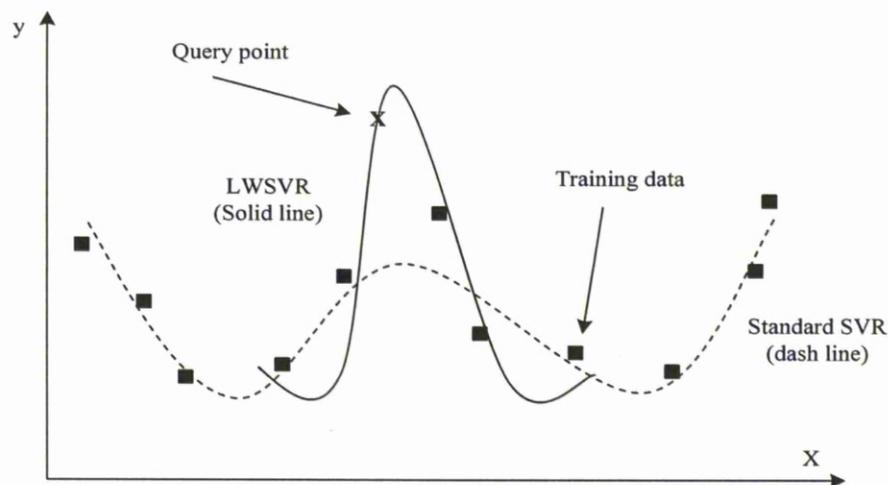


Figure 5.1: LWSVR conceptual interpretation.

## 5.5 Weighted distance algorithm for optimising the bandwidth

A bandwidth parameter  $h$  defines the scale or range over which generalisation is performed. This is a very important parameter which plays an important role in local modelling. If  $h$  is infinite then the local modelling becomes global. On the other hand, if  $h$  is too small, then it is possible that we will not have adequate number of data points in the neighbourhood for a good prediction.

There are several ways to use this parameter like, fixed bandwidth selection where  $h$  is constant, nearest neighbour bandwidth selection where  $h$  is set to be the distance between the query point and the  $K^{th}$  nearest point, global

bandwidth selection where  $h$  is calculated globally by an optimisation process, etc. [137]. In fixed bandwidth selection method,  $h$  is chosen as a constant value, therefore the training data with constant size and shape are used. However, it is the easiest and commonly used way to adjust the radius of the weighting function, its performance is unsatisfactory for nonlinear system as the density and distribution of data points are unlikely to be identical at every place of the data set [139].

In this thesis, the weighted distance algorithm which employs the Mahalanobis distance for optimising the bandwidth ( $h$ ) is proposed in order to improve the accuracy of LWSVR method (*i.e.* the improvement in the accuracy using this method over the nearest neighbour bandwidth selection method is in the range of 2.50%  $\sim$  4.00%).

The Mahalanobis distance based on correlation between variables by which different patterns can be identified and analysed. With this measure, the problem of scale and correlation inherent in Euclidean distance are no longer an issue. In the Euclidean distance, the set of points which have equal distance from a given location is a sphere. The Mahalanobis distance stretches this sphere correct for the respective scales of the different variables and to account for correlation among variables.

The standard Mahalanobis distance can be defined as:

$$MD(x) = \sqrt{(x - \mu)^T S^{-1} (x - \mu)} \quad (5.5.1)$$

where  $x$  is a vector of data,  $\mu$  is a mean and  $S^{-1}$  is inverse covariance matrix.

Defining the Mahalanobis distance between the query vector  $x_q$  and data vector  $x$  as  $MD_q = \sqrt{(x - x_q)^T S^{-1} (x - x_q)}$  where  $x$  belongs to the  $K$  nearest neighbours of the query vector  $x_q$  and  $S^{-1}$  is computed after removing the mean from each column, the bandwidth  $h_q$  is the function of  $MD_q$ :

$$h_q = \Theta(MD_q) \quad (5.5.2)$$

where  $MD_{min} \leq MD_q \leq MD_{max}$  and  $MD_{min}$  is the distance between  $x_q$  and the closest neighbour while  $MD_{max}$  is the distance between  $x_q$  and the farthest neighbour.

According to the LWR method, the point corresponding to  $MD_q = MD_{min}$  is most important that is  $h_{max} = \Theta(MD_{min}) = 1$  while the point corresponding to  $MD_q = MD_{max}$  is the least important, that is  $h_{min} = \Theta(MD_{max}) = \delta$ .  $\delta$  is a real constant. This constant is a low sensitivity parameter. Therefore, after few trials, we fix it to 0.01 which gives the best results.

The bandwidth  $h_q$  can be selected as a function of  $MD_q$  as following [139]:

$$h_q = \Theta(MD_q) = a \left( \frac{1 - bMD_q}{MD_q} \right)^2 + c \quad (5.5.3)$$

where  $a$ ,  $b$  and  $c$  are constants. By applying the boundary conditions, these constants are calculated and get [139]:

$$h_q = \Theta(MD_q) = (1 - \delta) \left( \frac{MD_{min}(MD_{max} - MD_q)}{MD_q(MD_{max} - MD_{min})} \right)^2 + \delta \quad (5.5.4)$$

The Gaussian kernel weighting function which used in this thesis can be written as following:

$$w(MD_q) = \exp - \left( \frac{MD_q}{(1 - \delta) \left( \frac{MD_{min}(MD_{max} - MD_q)}{MD_q(MD_{max} - MD_{min})} \right)^2 + \delta} \right)^2 \quad (5.5.5)$$

Fig. 5.2 presents the computation procedure of the LWSVR method which can be divided to four main stages. The first stage reconstructs the multivariate time series using the embedding dimension and the time delay constant. The second stage finds the  $K$  closest vectors, or nearest neighbours, of observed variables in the data set for each query vector then calculates the bandwidth parameter ( $h$ ) and weighting function of each point in the neighbourhood. After that, calculates the modified risk function of SVR. The third stage trains the SVR with modified risk function using only the  $K$  nearest neighbours. The fourth stage evaluates the model using the query vector as the input to estimate the process output.

Our approach is different from the previous works. First, the phase space of time series is reconstructed using the embedding dimension ( $d$ ) and time delay constant ( $m$ ). Here, the correlation dimension method and mutual information method are used to calculate  $d$  and  $m$  respectively. Secondly, the Euclidian distance is used to find the neighbouring points for each query point. Then,

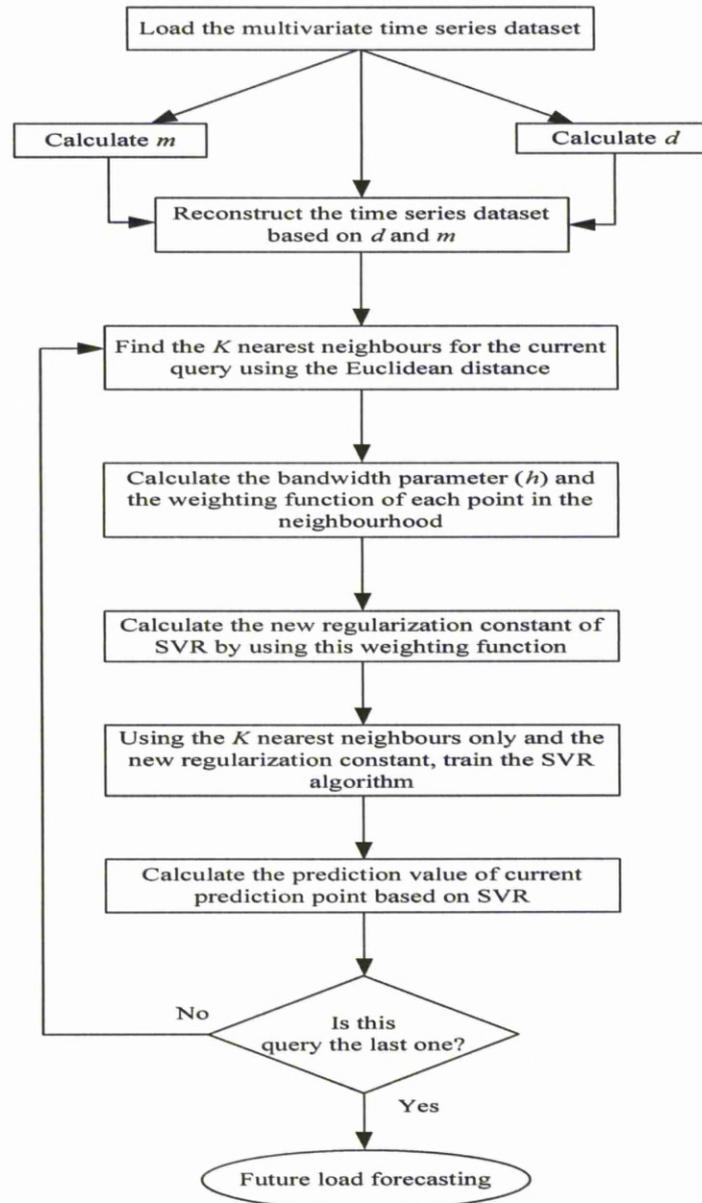


Figure 5.2: Flowchart of the LWSVR algorithm.

each point in the neighbourhood is weighted according to its distance from the query point to calculate the new regularisation parameter of SVR. Moreover, these neighbouring points only is used to train the prediction model instead of using all available training points, so that the drawbacks of global predictors can be overcome. Finally, the weighted distance algorithm is used to optimise the bandwidth of the weighting function so that the disadvantage of using this bandwidth as a fixed value can be overcome.

## 5.6 Case studies

To evaluate the performance of the proposed LWSVR method, two different datasets are used. The first one is the data provided by the EUNITE - the European Network on Intelligent Technologies for Smart Adaptive Systems - network during the daily peak load competition [140], while the second one is the hourly load and temperature from North American electric utility [141].

Regarding to the first dataset, EUNITE organised a world wide competition on methods to accurately predict electricity load [140]. The data provided to the competitors are as follows:

- Half hourly electricity load demand from January 1997 to December 1998.
- Average daily temperature from 1995 to 1998.
- Holidays information from 1997 to 1999.

For the second dataset, the hourly load and temperature from January 1985 to March 1991 are available. Certain characteristics can be reported about both datasets. Figure 5.3 shows the hourly load of EUNITE dataset for January 1998 and July 1998. It illustrates that the load in EUNITE dataset has some seasonal patterns. This figure indicates the relation between wether conditions especially temperature and electricity usage in different seasons where the electricity demand in summer period is lower than the electricity demand in winter period. Also, the load has daily and weekly periodicity. Same characteristics can be also noticed for the North American dataset.

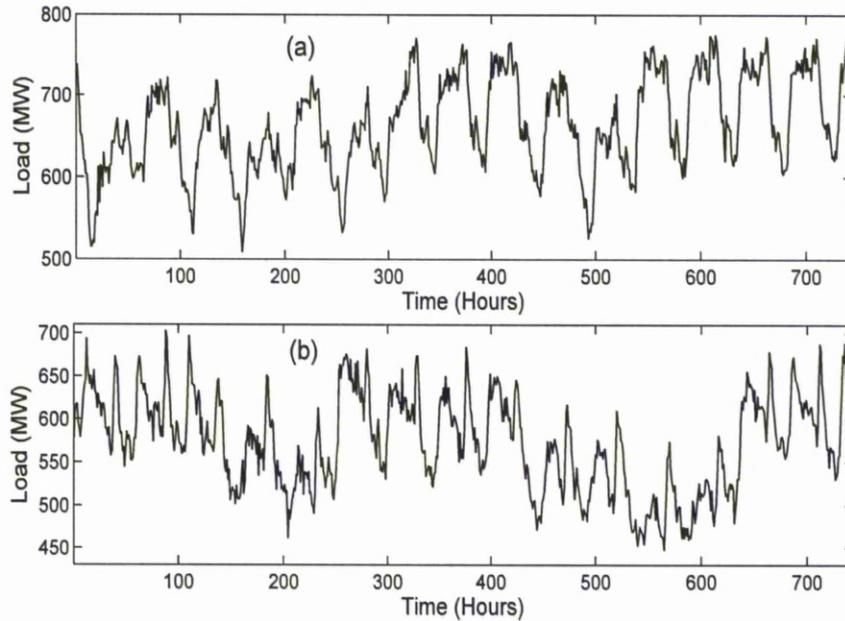


Figure 5.3: Hourly electricity demand for the EUNITE dataset from (a) 1<sup>st</sup> January 1998 to 31<sup>st</sup> January 1998 and (b) 1<sup>st</sup> July 1998 to 31<sup>st</sup> July 1998.

To implement a good model, there are some important parameters to choose. Choosing the proper values of  $d$  and  $m$  is a critical step in the algorithm. The correlation dimension method and the mutual information method are used to selecting  $d$  and  $m$ , respectively as discussed in Chapter 4 and the optimal values of these parameters are shown in Table 5.1. Using the obtained values of  $d$  and  $m$  the multivariate time series can be reconstructed as described in Section 5.2. Also, choosing  $K$  is very important step in order to establish the local prediction model. So, equation (4.3.1) is used to calculate  $K$  as described in Section 4.3 where the parameters  $k_{\max}$  and  $\alpha$  are fixed for all test cases in this Chapter at 30% of  $N$  and 75, respectively.

For all test cases, the squared exponential function (equation 3.5.13) whose coefficients are determined using maximum likelihood method is used for GP. In addition, there are some key parameters for SVR such as  $C$  and  $\sigma$  in the Gaussian kernel function. To choose the suitable values of these parameters, the same procedures which described in subsection 4.4.2 are used. The values of

Table 5.1: Phase reconstruction parameters for each dataset

Dataset	Load time series		Temperature time series	
	$d_1$	$m_1$	$d_2$	$m_2$
EUNITE competition	4	2	4	2
North American electric utility	4	5	3	9

these parameters are  $C = 28$  and  $\sigma = 2.3$  for the EUNITE dataset, while these values are  $C = 0.7$  and  $\sigma = 1.81$  for the North American electric utility dataset.

For all performed experiments, we quantified the prediction performance with mean absolute percentage error (MAPE) and normalised mean square error (NMSE). They are defined by equation (4.4.1) and equation (4.4.2), respectively.

### 5.6.1 Case 1: EUNITE competition dataset

In this case, the dataset provided in EUNITE competition is used. Our goal in this case as well as the goal of the competition is to forecast the maximum daily load for January 1999. To achieve this goal, the load and temperature information of winter data (from January to March and October to December in 1997 and 1998) are used as a training period.

The performance of the LWSVR model is compared with LWR, local SVR and local GP methods. This comparison is shown in Table 5.2 and depicted in Figs. 5.4 and 5.5.

These results show that the LWSVR model outperforms LWR, local SVR and local GP methods. It improves the MAPE over LWR, local SVR and local GP methods by 24.72%, 12.42% and 11.26%, respectively. Moreover, the actual load and forecasted load values using LWSVR of the peak daily load of January 1999 are plotted in Fig. 5.6. The results of Fig. 5.6 show that the LWSVR's prediction values are very close to the actual values.

To further study the superiority of LWSVR over other published methods.

Table 5.2: Comparison of the LWSVR method and other methods using the dataset of EUNITE competition

Prediction method	MAPE	NMSE
LWR	1.78	0.061
Local SVR	1.53	0.050
Local GP	1.51	0.048
LWSVR	1.34	0.032

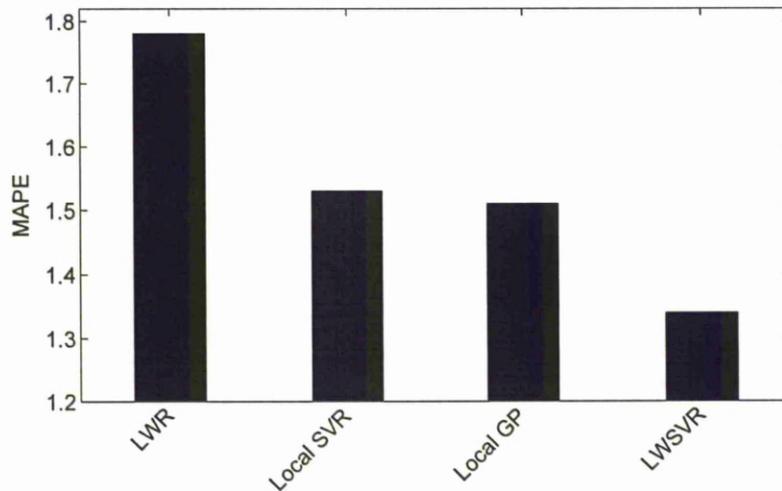


Figure 5.4: Comparison of LWSVR method and other methods using MAPE for the dataset of EUNITE competition.

Its performance is compared with some published methods that employ the dataset of EUNITE competition. These methods are:

- Method A [26]: Support vector machine.
- Method B [142]: Support vector machine optimised by genetic algorithm.
- Method C [143]: Support vector machine based input dimension reduction.
- Method D [144]: ANN with extended Bayesian training method.

All of these methods are global methods. During the EUNITE competition, the real temperature of January 1999 was not provided to the competitors. So

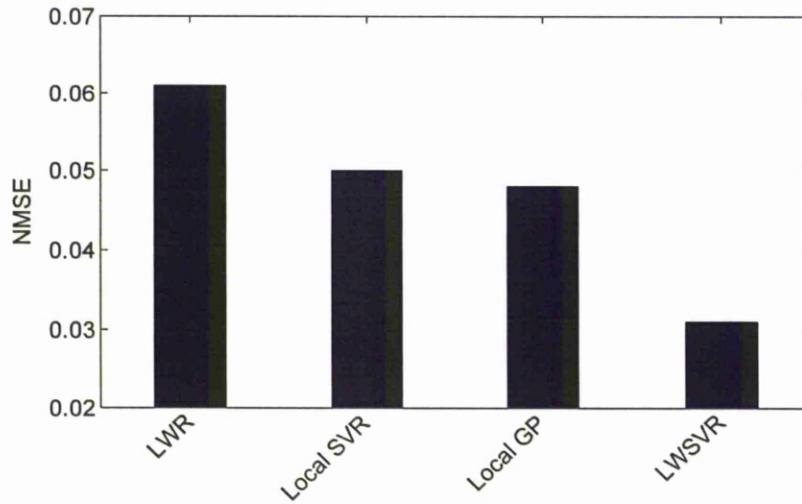


Figure 5.5: Comparison of LWSVR method and other methods using NMSE for the dataset of EUNITE competition.

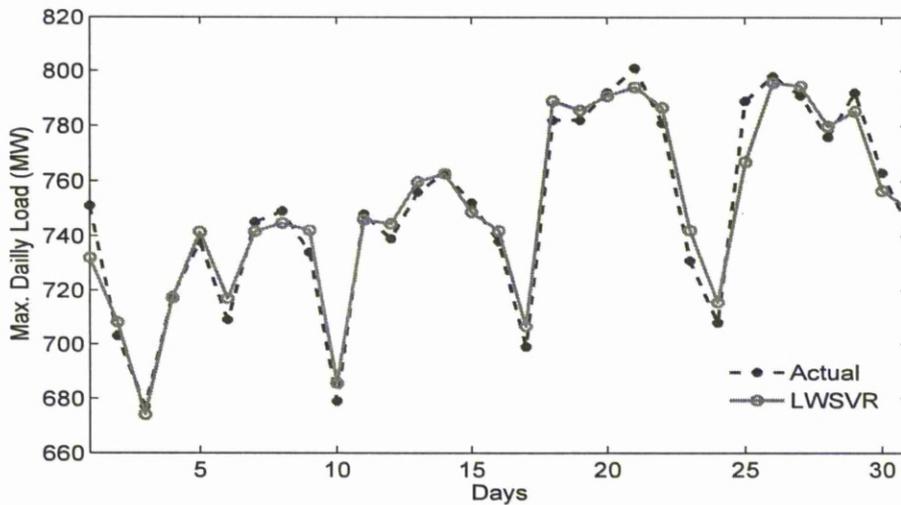


Figure 5.6: Forecasted and actual maximum daily load in January 1999.

that, the load data only was used by some competitors like [26] to forecast the maximum daily load in January 1999. In [26], SVM was employed to generate the winning entry of the EUNITE competition using the load data from January to March and October to December in 1997 and 1998 without using the temperature information. In [142], genetic algorithm was used to choose the optimal values of

the SVM's parameters. A new method was proposed in [143] to load forecasting by establishing the feature selection model and using floating search method to find the feature subset. Then SVM can be used to forecast the load using a small sample of the data. Moreover, after the competition was closed, the real temperature of January 1999 was available. This encourage some researchers like [144] to use the temperature information to forecast the maximum daily load of January 1999. In [144], the ANN with extended Bayesian training method was used and gave better results than that of the competition's winner.

More details about the data used and the training period of each method can be found in Table 5.3. Moreover, the Gaussian radial basis kernel function (3.4.27) not only used in all our experiments but also used in [26, 142] and [143]. To compare LWSVR method with methods A, B, C and D, we used the same experimental setup as used in each method. This comparison is shown in Table 5.3. The results of Table 5.3 show that the LWSVR outperforms other methods. The improvements of LWSVR over methods are shown in Table 5.4.

### 5.6.2 Case 2: North American electric utility dataset

In this case, the dataset of the North American electric utility is used. At this electric power utility, the daily forecasts were made at 8:00 A.M. Forecasts were produced for the entire next day, starting at midnight and through the following midnight and hence they were made from 16 to 40 hours into the future. On Friday, the forecasts were produced for the entire weekend as well as Monday (16 to 88 hours into the future). The objective is to forecast the hourly load, from 16 up to 40 hour (steps) ahead for weekdays and from 16 up to 88 hour (steps) ahead for weekends during the test period which goes from November 1990 to March 1991.

To compare LWSVR method with LWR, local SVR, local GP and some published methods that employ the same dataset, we used the same experimental setup as used in [144] (Method D) and [30] (Method E) which use a multiple regression model called (EGRV). That is, the hourly load and temperature data from the month to be forecasted and from two month earlier, along with the data corresponding to the same window in the previous year are used as a training

Table 5.3: Comparison of the LWSVR method and other methods using the dataset of EUNITE competition (MAPE)

Input data	Training period	MAPE				
		LWSVR	Method A	Method B	Method C	Method D
Load only	Load data of January to March and October to December in 1997 and 1998	1.41	1.95	–	–	–
	Load data for two years (1997 and 1998)	1.44	–	1.93	1.70	–
Load and temperature	Load and temperature data for two years (1997 and 1998)	1.38	–	–	–	1.75

Table 5.4: Improvement of LWSVR over other methods using the dataset of EUNITE competition

Prediction Method	Improvement
LWSVR	—
Method A [26]	27.69%
Method B [142]	25.39%
Method C [143]	15.29%
Method D [144]	21.14%

period. First, the error of each day during the testing period is calculated. Then the average error of each day of the week (Monday to Sunday) during the testing period is calculated. Finally, the overall mean performance for the entire testing period for each model can be calculated. These results are summarised in Table 5.5 and depicted in Figs. 5.7 and 5.8.

Table 5.5: Comparison of LWSVR method and other methods using the dataset of North American electric utility

Prediction method	MAPE	NMSE
Method D [144]	4.88	—
Method E [30]	4.73	—
LWR	4.71	0.131
Local SVR	4.08	0.112
Local GP	4.16	0.119
LWSVR	3.62	0.083

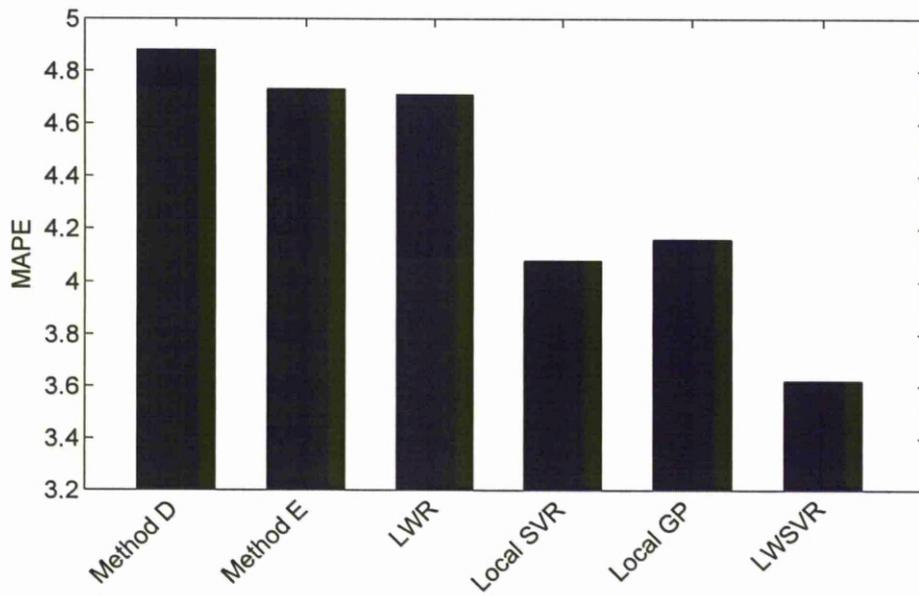


Figure 5.7: Comparison of LWSVR method and other methods using MAPE for the dataset of North American electric utility.

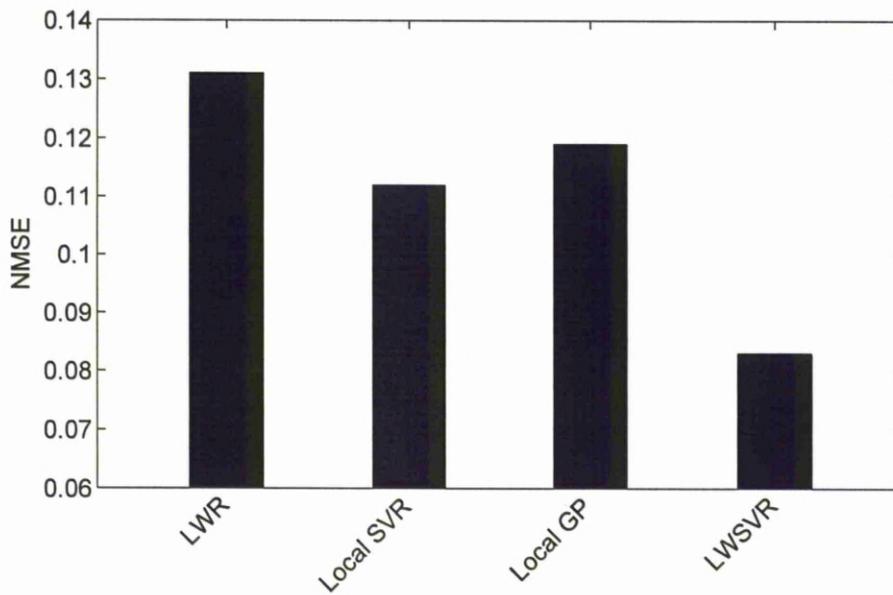


Figure 5.8: Comparison of LWSVR method and other methods using NMSE for the dataset of North American electric utility.

It can be seen from these results that the LWSVR method gives better performance than LWR, local SVR, local GP, method D [144] and method E [30]. It improves the performance (MAPE) over LWR, local SVR, local GP, method D [144] and method E [30] by 23.14%, 11.27%, 12.98%, 25.82% and 23.47%, respectively.

Fig. 5.9 shows the average prediction MAPE of every day of the week (Monday to Sunday) during the testing period. These results confirm the superiority of the LWSVR method over other methods.

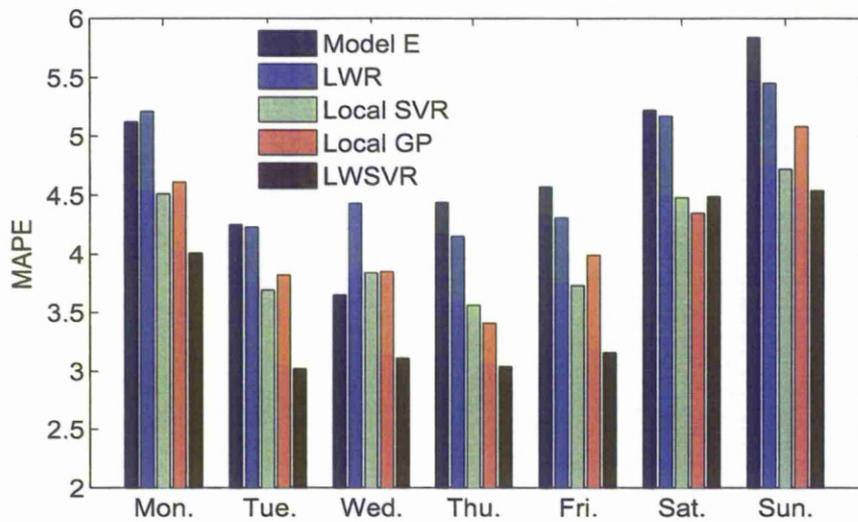


Figure 5.9: Average prediction MAPE of every day of the week during the testing period using the dataset of North American electric utility.

In addition, the MAPE of the whole testing data for the 24 hours is summarised in Table 5.6. From these results, it can be noticed that the LWSVR model exhibits a better performance than LWR, local SVR, local GP and method E [30]. Most MAPE values of LWSVR are under 4 (70.83% of the 24 hours), while in local SVR, half MAPE values are under 4. Also, we can notice that all MAPE values of of method E [30] are over 4 except one value only.

Fig. 5.10 presents one example for the dataset of North American electric utility. It shows one day ahead forecasted load using LWSVR versus the actual load of the period from 27<sup>th</sup> November 1990 to 3<sup>rd</sup> December 1990. These results show that the LWSVR's prediction values are very close to the actual values.

Table 5.6: Comparison of the LWSVR method and other methods using the dataset of North American electric utility (MAPE of the 24 Hours)

Hour	MAPE				
	Method E [30]	LWR	Local SVR	Local GP	LWSVR
1	4.08	4.24	3.67	3.74	3.10
2	4.44	4.57	3.96	4.05	3.04
3	4.84	4.81	4.30	4.38	3.99
4	5.03	5.01	4.46	4.45	4.01
5	5.47	5.23	4.44	4.53	4.43
6	5.66	5.52	4.71	4.80	4.04
7	5.43	5.51	4.80	4.88	4.10
8	4.76	4.66	4.21	4.25	3.67
9	4.22	3.98	3.37	3.55	3.35
10	3.84	4.07	3.54	3.74	3.03
11	4.04	4.12	3.66	3.73	3.17
12	4.12	4.55	3.94	3.93	3.34
13	4.55	4.44	3.86	3.94	3.18
14	5.02	4.87	4.07	4.14	3.97
15	5.31	5.22	4.52	4.63	4.29
16	5.43	5.60	4.87	4.98	4.01
17	5.36	4.88	4.25	4.24	4.21
18	4.94	4.62	4.01	4.02	3.25
19	4.19	4.33	3.75	3.71	3.32
20	4.30	3.86	3.24	3.22	3.20
21	4.42	4.49	3.89	3.97	3.28
22	4.40	4.55	3.97	3.90	3.54
23	4.57	4.89	3.99	4.15	3.39
24	5.04	5.02	4.54	4.79	3.98
Average	4.73	4.71	4.08	4.16	3.62

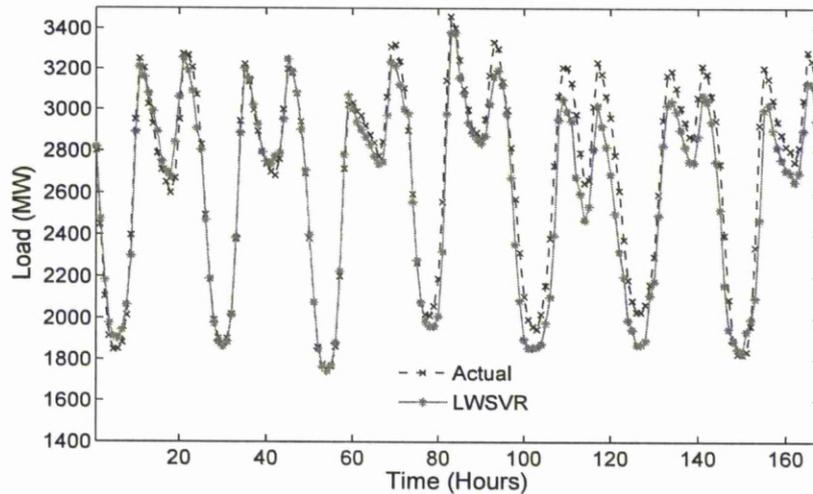


Figure 5.10: One day ahead forecasted and actual load from 27<sup>th</sup> November 1990 to 3<sup>rd</sup> December 1990.

## 5.7 Conclusions

In this chapter, a new approach has been proposed to solve the load forecasting problem. The idea of the proposed approach is to modify the risk function of the support vector regression algorithm with use of locally weighted regression and keeping the regularisation term in its original form. In addition, the weighting functions's bandwidth is optimised using the proposed weighted distance algorithm based Mahalanobis distance.

Two different real world datasets have been used to evaluate the performance of the proposed method (LWSVR). The proposed method has been compared with LWR, local SVR, local GP and some published papers employing the same datasets. The numerical results, achieved based on different measuring errors show the superiority of LWSVR over LWR, local SVR, local GP and other published methods.

## Chapter 6

# Locally Weighted Group Method of Data Handling Based KPCA for STLF

### 6.1 Introduction

Phase space reconstruction is an important step in local prediction methods. The traditional time series reconstruction techniques usually use the coordinate delay (CD) method [113] to calculate the embedding dimension and the time delay constant of the time series [145].

The traditional time series reconstruction techniques have a serious problem. In which there may be correlation between different features in reconstructed phase space. Consequently, the quality of phase space reconstruction and modelling will be affected [11]. In recent years, to process nonlinear time series, the kernel principal component analysis (KPCA) which is one type of nonlinear principal component analysis (PCA) is used [146].

KPCA is an unsupervised technique that is based on performing principal component analysis in the feature space of a kernel. The main idea of KPCA is first to map the original inputs into a high-dimensional feature space via a kernel map, which makes data structure more linear, and then to calculate principal components in the high-dimensional feature space [11].

In Chapter 5 the LWSVR algorithm which is derived by combining SVR and

LWR is proposed to solve the STLTF problem and gives a very good performance. Nevertheless, some limitations exist in the SVR approach [147]. First, the most serious limitation of SVR algorithm is uncertain in choice of a kernel. The best choice of kernel for a given problem is still a research issue. The second limitation is the selection of SVR parameters due to the lacking of the structural methods for confirming the selection of parameters efficiently. Finally, the SVR algorithm is computationally slower than the artificial neural networks.

To overcome such limitations, a new method is proposed in this chapter using an alternative machine learning technique which is called group method of data handling (GMDH). GMDH is a self-organising method that was firstly developed by Ivakhnenko [9] as a multivariate analysis method for modelling and identification of complex systems. GMDH has been applied to solve many prediction problems with success [148, 10, 149]. The main idea of GMDH is to build an analytical function in a feedforward network based on a quadratic node transfer function whose coefficients are obtained using a regression technique [150].

The proposed method is derived by combining the GMDH with the local regression method and weighted least squares regression and employing the weighted distance algorithm which uses the Mahalanobis distance to optimise the weighting function's bandwidth. In the proposed model, the phase space is reconstructed based on KPCA method, so that the problem of the traditional time series reconstruction techniques can be avoided. The LWGMDH method has been evaluated using two real world datasets and compared with some published methods using the same datasets. The results show that the proposed method exhibits superior performance to that of other methods.

The rest of this chapter is organised as follows: Section 6.2 describes the time series reconstruction based on KPCA method. Section 6.3 reviews the GMDH algorithm. The LWGMDH method is introduced in Section 6.4. Then the experimental results and comparisons with other methods are presented in Section 6.5. Finally, Section 6.6 concludes this chapter of our work.

## 6.2 Time series reconstruction based on KPCA

### 6.2.1 Introduction to KPCA

The PCA is a well-known method for feature extraction [151]. It involves the computations in the input (data) space so it is a linear method in nature. KPCA is an unsupervised technique that is based on performing principal component analysis in the feature space of a kernel. KPCA can be used to reconstruct the time series, on the basis of which some kernel principal components are chosen according to their correlative degree to the model output to form final phase space of the nonlinear time series.

In KPCA the computations are performed in a feature space that is nonlinearly related to the input space. This feature space is that defined by an inner product kernel in accordance with Mercer's theorem [43]. Due to the nonlinear relationship between the input space and feature space the KPCA is nonlinear. However, unlike other forms of nonlinear PCA, the implementation of KPCA relies on linear algebra by mapping the original inputs into a high-dimensional feature space via a kernel map, which makes data structure more linear.

Figure 6.1 illustrates the basic idea of KPCA [43]. The feature space shown in Fig. 6.1(b) is nonlinearly related to the input space (Fig. 6.1(a)) via the transformation  $\phi(x)$ . The contour lines shown in Fig. 6.1(b) represent constant projections onto a principle eigenvector (drawn as an arrow). The transformation  $\phi(x)$  has been chosen in such away that the images of the data points induced in the feature space congregate themselves essentially along the eigenvector. Figure 6.1(a) shows the nonlinear contour lines in the input space that correspond to those in the feature space.

### 6.2.2 KPCA implementation

The basic idea of KPCA is to map the data  $x$  into a high dimensional feature space  $\phi(x)$  via a nonlinear mapping, and perform the linear PCA in that feature space as:

$$Q(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle \quad (6.2.1)$$

where  $x_i, x_j$  are variables in input space and  $Q(x_i, x_j)$  is called kernel function.

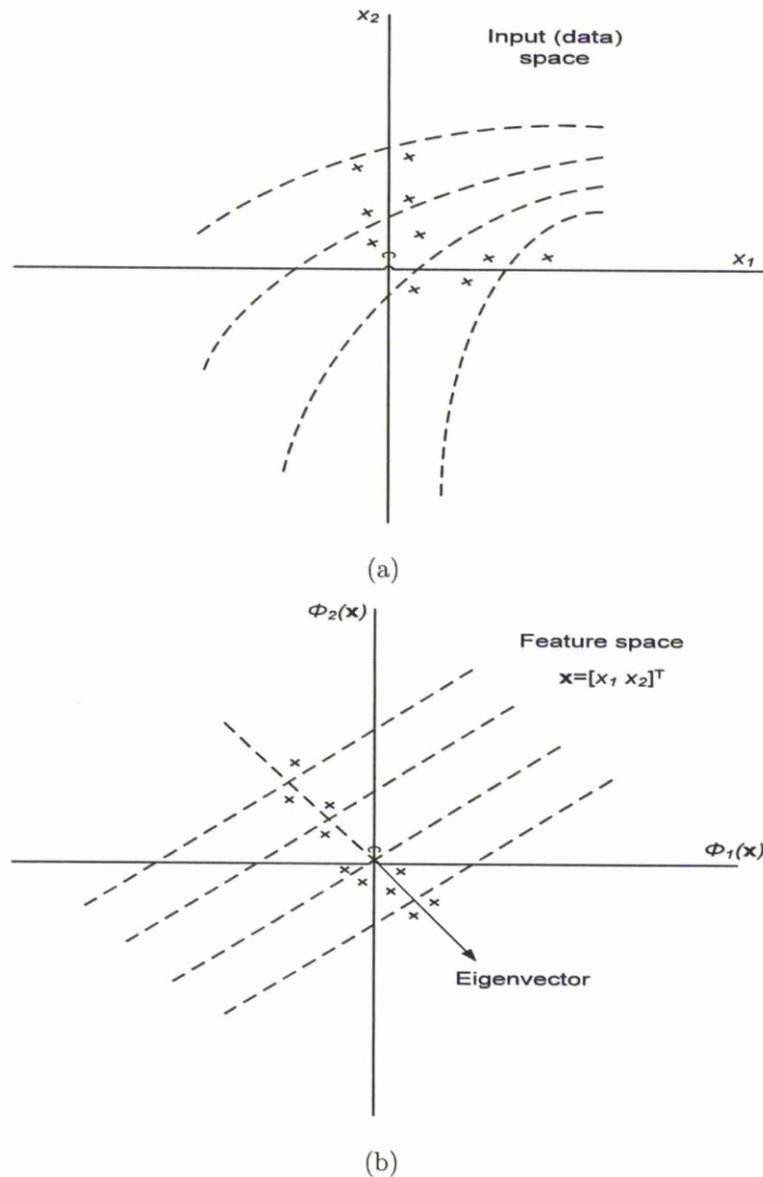


Figure 6.1: Illustration of the basic idea of KPCA: (a) Two dimensional input space, displaying a set of data points, and (b) Two dimensional feature space, displaying the induced images of the data points congregating around a principle eigenvector.

Given a set of data  $\{x_i\}_{i=1}^N$  where each  $x_i \in \mathfrak{R}^d$  and  $N$  is the number of data points, we have a corresponding set of feature vector  $\{\phi(x_i)\}_{i=1}^N$ . Accordingly, the sample covariance matrix of  $\phi(x_i)$  can be defined as follows:

$$\tilde{C} = \frac{1}{N} \sum_{i=1}^N \phi(x_i) \phi^T(x_i) \quad (6.2.2)$$

As in PCA method, we have to ensure that the set of feature vectors  $\{\phi(x_i)\}_{i=1}^N$  have zero mean [43]:

$$\frac{1}{N} \sum_{i=1}^N \phi(x_i) = 0 \quad (6.2.3)$$

Proceeding then on the assumption that the feature vectors have been centred, KPCA solves equation (6.2.4) for the eigenvalues:

$$\lambda_i v_i = \tilde{C} v_i, \quad i = 1, 2, \dots, N \quad (6.2.4)$$

where  $\lambda_i$  is one of the non-zero eigenvalues of  $\tilde{C}$  and  $v_i$  is the corresponding eigenvector.

Because the eigenvectors  $v_i$  lie in the plane which is composed of  $\phi(x_1), \phi(x_2), \dots, \phi(x_N)$ , therefore [11]:

$$\lambda_i \phi(x_i) \cdot v_i = \phi(x_i) \cdot \tilde{C} v_i \quad i = 1, 2, \dots, N \quad (6.2.5)$$

and there exist coefficients  $\alpha$  meet:

$$v = \sum_{j=1}^N \alpha_j \phi(x_j) \quad (6.2.6)$$

Substituting equation (6.2.2) and equation (6.2.6) into equation (6.2.4) and defining an  $N \times N$  matrix  $Q$  which is defined by equation (6.2.1), the following formula can be got [43]:

$$\sum_{i=1}^N \sum_{j=1}^N \alpha_j \phi(x_i) Q(x_i, x_j) = N \lambda \sum_{j=1}^N \alpha_j \phi(x_j) \quad (6.2.7)$$

Equation (6.2.7) can be written in the compact matrix form [43]:

$$N \lambda \alpha = Q \alpha \quad (6.2.8)$$

Assuming the eigenvectors of  $\phi(x_i)$  is of unit length  $v_i \cdot v_i = 1$ , each  $\alpha_i$  must be normalised using the corresponding eigenvalue by  $\tilde{\alpha}_i = \frac{\alpha_i}{\sqrt{N\lambda_i}}$ ,  $i = 1, 2, \dots, N$ .

Finally the principal component for  $x_i$ , based on  $\tilde{\alpha}_i$ , can be calculated as following:

$$p_t(i) = v_i^T \phi(x_i) = \sum_{j=1}^N \tilde{\alpha}_i(j) Q(x_j, x_i), \quad i = 1, 2, \dots, N \quad (6.2.9)$$

From equation (6.2.9), one can notice that the maximal number of principal components that can be extracted by KPCA is  $N$ . The dimension of  $p_t$  can be reduced in KPCA by considering the first several eigenvectors that is sorted in descending order of the eigenvalues. In this thesis, the commonly used Gaussian radial basis function kernel (equation 3.4.27) is employed.

### 6.3 Group method of data handling (GMDH)

Suppose that the original dataset consists of  $d$  columns of the values of the system input variables that is  $X = (x_1(t), x_2(t), \dots, x_d(t))$ , ( $t = 1, 2, \dots, N$ ) and a column of the observed values of the output and  $N$  is the length of the dataset.

The connection between inputs and output variables can be represented by an infinite Volterra-Kolmogorov-Gabor (VKG) polynomial of the form:

$$y = a_0 + \sum_{i=1}^N a_i x_i + \sum_{i=1}^N \sum_{j=1}^N a_{ij} x_i x_j + \sum_{i=1}^N \sum_{j=1}^N \sum_{k=1}^N a_{ijk} x_i x_j x_k + \dots \quad (6.3.1)$$

where  $N$  is the number of the data points,  $A(a_0, a_i, a_{ij}, a_{ijk}, \dots)$  and  $X(x_i, x_j, x_k, \dots)$  are vectors of the coefficients and input variables of the resulting multi-input single-output system, respectively.

It is showed by Ivakhnenko that the VKG series can be used as a cascade of second order polynomials using only pairs of variables [9]. The corresponding network as shown in Fig. 6.2 can be constructed from simple polynomial. As the learning procedure evolves, branches that do not contribute significantly to the specific output can be deleted, this allows only the dominant causal relationship to evolve.

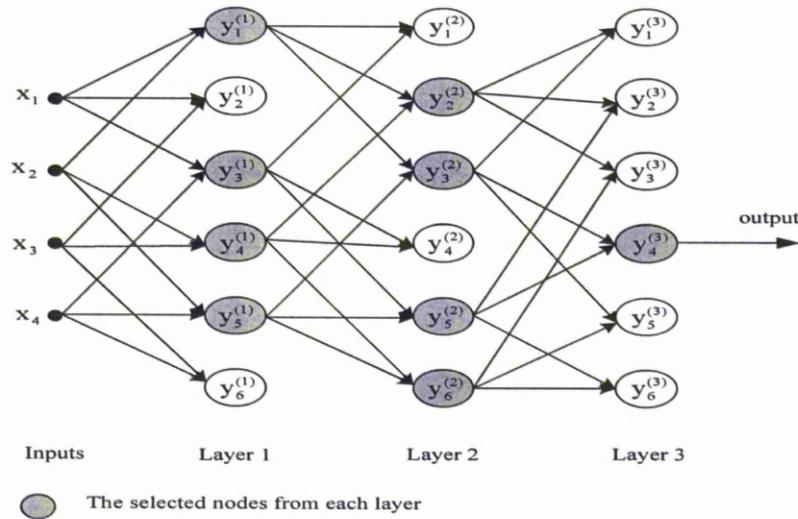


Figure 6.2: GMDH network.

### 6.3.1 GMDH layers

When constructing the GMDH network, all combinations of the system input variables  $X = \{x_1, x_2, \dots, x_d\}$  are generated and used as inputs of the first layer of the network. The outputs from this layer are then classified then the best nodes are selected for input into the next layer with all combinations of the selected outputs being sent into layer 2. This process is continued until the current layer is found to not be as good as the previous one. Therefore, the previous layer best neurons is then used as the final solution.

### 6.3.2 GMDH nodes

Each layer consists of nodes generated to take a specific pair of the combination of inputs as its source. The total number of nodes (polynomials) that can be constructed in each layer is equal to  $\frac{d(d-1)}{2}$ . Each node (polynomial) produces a set of coefficients  $A(a_0, a_1, a_2, a_3, a_4, a_5)$  such that equation (6.3.2) is estimated using the set of training data.

$$\hat{y} = a_0 + a_1x_i + a_2x_j + a_3x_ix_j + a_4x_i^2 + a_5x_j^2 \quad (6.3.2)$$

This equation is tested for fit by determining the mean square error (MSE)

of the predicted  $\hat{y}$  and actual  $y$  values using the set of testing data as follows:

$$e = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2 \quad (6.3.3)$$

The least square solution of (6.3.3) is given by:

$$A = (X^T X)^{-1} X^T Y \quad (6.3.4)$$

where  $Y = [y_1, y_2, \dots, y_N]^T$ ,

$$X = \begin{bmatrix} 1 & x_{1P} & x_{1Q} & x_{1P}x_{1Q} & x_{1P}^2 & x_{1Q}^2 \\ 1 & x_{2P} & x_{2Q} & x_{2P}x_{2Q} & x_{2P}^2 & x_{2Q}^2 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 1 & x_{NP} & x_{NQ} & x_{NP}x_{NQ} & x_{NP}^2 & x_{NQ}^2 \end{bmatrix} \text{ and } P, Q \in \{1, 2, \dots, d\}$$

By using these coefficients and processing the set of testing data in equation (6.3.2), the node then computes its error using equation (6.3.3). The error is the measure of fit that this node achieved.

### 6.3.3 GMDH connections

The nodes of the GMDH layer are sorted based on the error produced, then the best  $n$  nodes are saved. The generated  $y$  values of each node become one set of inputs which can be used by the next layer when it combines all outputs from the previous layers nodes assigning them to the new layers nodes (See Fig. 6.2). The layer must remember the nodes that were saved so that other data submitted to the network will follow the same generated path to the output [152]. When the GMDH network is completed, there is a set of original inputs that filtered through the layers to the optimal output node. This is the computational network that is to be used in computing predictions.

### 6.3.4 GMDH network and its advantages

The GMDH network training algorithm procedures can be summarised as follows:

- GMDH network begins with only input nodes and all combinations of different pairs of them are generated using a quadratic polynomial and sent into the first layer of the network. The advantage of using pairs of input is that only six weights (coefficients) have to be computed for each node.
- Use least squares regression to compute the optimal parameters for the function in each node to make it best fit the training data.
- Compute the mean squared error for each node.
- Sort the nodes in order of increasing error.
- Select the best nodes which give the smallest error from the candidate set to be used as input into the next layer with all combinations of different pairs of them being sent into second layer.
- This process is repeated until the current layer is found to not be as good as the previous one. Therefore, the previous layer best node is then used as the final solution as shown in Fig. 6.3.

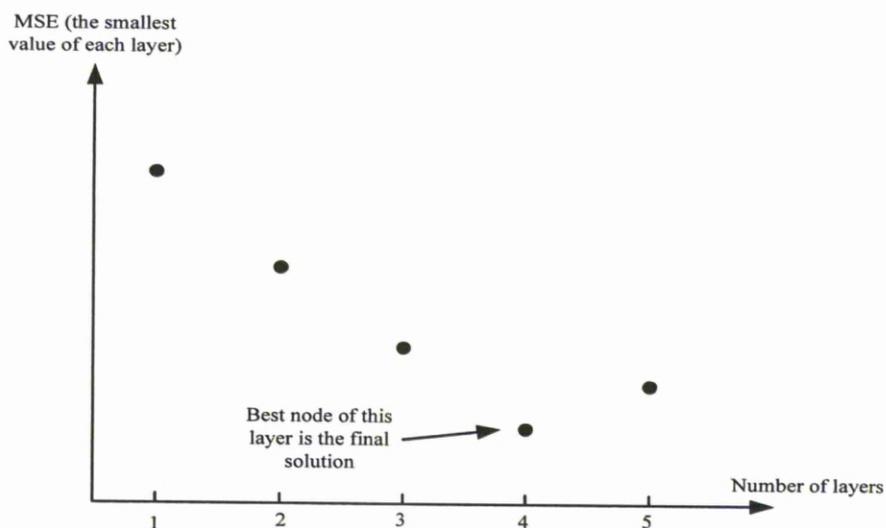


Figure 6.3: Stopping criterion.

There could be summarised that the GMDH networks influence the contemporary artificial neural network algorithms with several advantages [153]:

- The GMDH networks offer adaptive network representations that can be tailored to the given task. The number of layers, model structure and other optimal parameters are determined automatically.
- The GMDH networks learn the weights rapidly in a single step by standard least square regression which guarantees finding locally good weights due to the reliability of the regression technique.
- The GMDH networks feature sparse connectivity which means that the best discovered networks can be trained fast.

More details about the GMDH and its different applications have been reported in [150, 154].

## 6.4 Locally weighted group method of data handling (LWGMDH)

The LWGMDH method is derived by combining the GMDH with the local regression method and WLS regression. To predict the output values  $\hat{y}$  for each query point ( $x_q$ ) belongs to the testing set, the GMDH will be trained using the  $K$  nearest neighbours only ( $1 < K \ll N$ ,  $N$  is the number of training points) of this  $x_q$ . The coefficient parameters is calculated using WLS regression where each point in the neighbourhood is weighted according to its distance from the  $x_q$ . The points that are close to  $x_q$  have large weights, and the points far from  $x_q$  have small weights.

Overall, the framework of the design procedure of the LWGMDH comes as a sequence of the following steps.

1. Reconstruct the time series: Load the multivariate time series dataset  $X = (x_1(t), x_2(t), \dots, x_n(t))$ , ( $t = 1, 2, \dots, N$ ). Using the KPCA method to calculate the number of principal components of each dataset (we set

the time delay constant of all datasets equal to 1). Then, reconstruct the multivariate time series using these values.

2. Form a training and validation data: The input dataset after reconstruction  $\tilde{X}$  is divided into two parts, that is a training  $\tilde{X}_{\text{tr}}$  dataset and validation  $\tilde{X}_{\text{va}}$  dataset. The size of the training dataset is  $N_{\text{tr}}$  while the size of the validation dataset is  $N_{\text{va}}$ .
3. For each query point  $x_q$ , choosing the  $K$  nearest neighbours of this query point using the Euclidian distance between  $x_q$  and each point in  $\tilde{X}_{\text{tr}}$  ( $1 < K \ll N_{\text{tr}}$ ).
4. Create the first layer: Using the  $K$  nearest neighbours only, all combinations of the inputs are generated based on (6.3.2) and sent into the first layer of the network.
5. Estimate the coefficients parameters of each node: The vector of coefficients  $A$  is derived by minimising the locally weighted mean squared error

$$e = \frac{1}{K} \sum_{i=1}^K w_i^2 (y_i - \hat{y}_i)^2 \quad (6.4.1)$$

where  $w$  is the weighting function. Many weighting functions are proposed by the researchers [136]. In this work, the Gaussian kernel weighting function is used as following:

$$w_i = \sqrt{\exp\left(-\frac{\|x_i - x_q\|^2}{h^2}\right)} \quad (6.4.2)$$

where  $x_q$  is the query point,  $x_i$  is a data point belongs to the nearest neighbours points of  $x_q$  and  $h$  is the bandwidth parameter which plays an important role in local modelling. An optimisation method for the bandwidth is proposed in Section 5.5.

The weighted least square solution of (6.4.1) is given by:

$$A = ((WX)^T(WX))^{-1}(WX)^T(Wy) \quad (6.4.3)$$

where  $W$  is the diagonal matrix with diagonal elements  $W_{ii} = \sqrt{w_i}$  and zeros elsewhere [137],  $Y = [y_1, y_2, \dots, y_K]^T$ ,  $A(a_0, a_1, a_2, a_3, a_4, a_5)$ ,  $X$  is defined in the last section but with number of rows equal to  $K$  (the number

of the nearest neighbours). This procedure is implemented repeatedly for all nodes of the layer.

6. Select the nodes with the best predictive capability to create the next layer: Each node in the current layer is evaluated using the training and validation datasets. Then the nodes which gives the best predictive performance for the output variable are chosen for input into the next layer with all combinations of the selected nodes based on equation (6.3.2) being sent into next layer. In this thesis, a predetermined number of these nodes is used. The coefficients parameters of each node in this layer can be estimated using the same procedures in step 5.
7. Check the stopping criterion: The modelling can be terminated when:

$$e_{l+1} \geq e_l \quad (6.4.4)$$

where  $e_{l+1}$  is a minimal identification error of the current layer while  $e_l$  is a minimal identification error of the previous layer. So that the previous layer ( $l$ ) best node is then used as the final solution of the current query point. If the stopping criterion is not satisfied, the model has to be expanded. The steps 6 to 7 can be repeated until the stopping criterion is satisfied.

8. Then, the steps 3 to 7 can be repeated until the future values of different query points are all acquired.

Figure 6.4 presents the computation procedure of the proposed method.

## 6.5 Short term load forecasting results

To evaluate the performance of the proposed LWGMDH method, two different datasets are used. The first one is the hourly load and temperature from North American electric utility [141] as presented in Section 5.6, while the second one is the historical load, temperature and price data (for two years 2002 and 2003) from the Victorian electricity market in Australia.

In the second dataset, the historical loads and prices in half hourly basis was collected from Australian Energy Market Operator (AEMO) [155], while historical weather data (temperature) was collected from the Bureau of Meteorology,

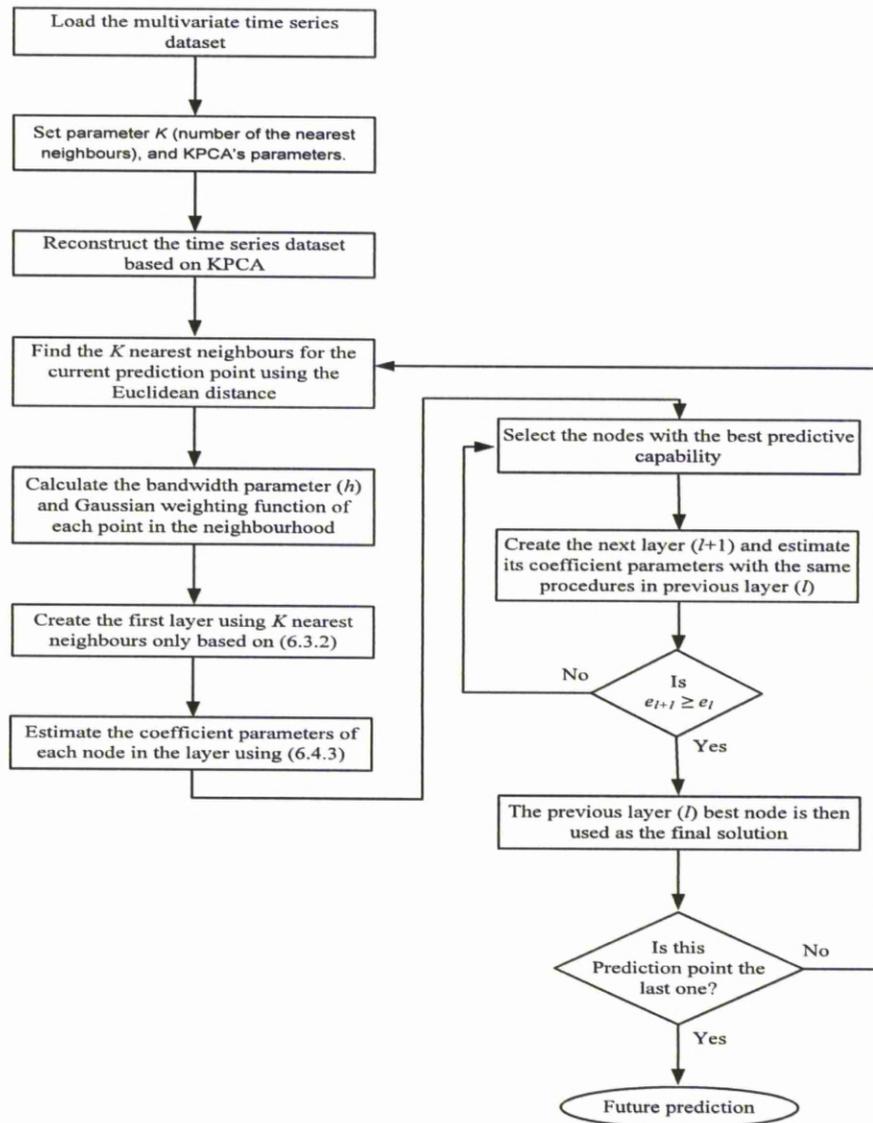


Figure 6.4: Flowchart of the LWGMDH method.

VIC Climate and Consultancy Section. These data are transformed to an hourly basis by averaging two half hours. Fig. 6.5 illustrates the electricity demand of Victoria for two months (February 2003 and June 2003).

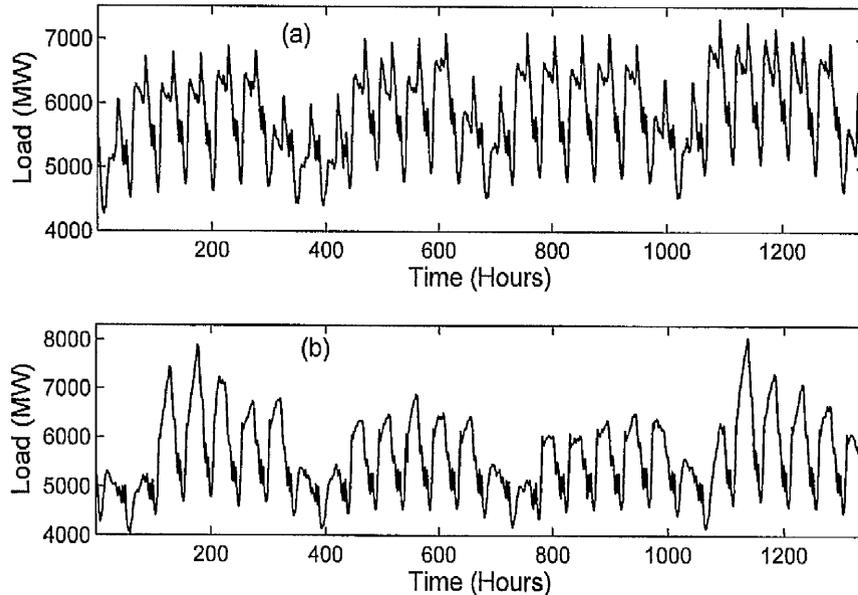


Figure 6.5: Hourly electricity demand in Victoria city from (a) June 2003 (b) February 2003.

As the previous datasets, Fig. 6.5 indicates that the load in Victoria has multiple seasonal patterns, corresponding to a daily and weekly periodicity, respectively. In addition, Fig. 6.5 indicates that the electricity demand in June 2003 (winter period in Australia) is lower than the electricity demand in February 2003 (summer period in Australia). This shows the relation between the load and the temperature (the maximum and minimum temperatures for February 2003 were recorded to be nearly 35 and 10  $C^\circ$ , respectively, while these values are 20 and 3  $C^\circ$ , respectively for June 2003).

To show the effectiveness of the proposed LWGMDH method, numerical simulations comparing with conventional GMDH (as a global method), local SVR, LWSVR based CD and LWSVR based KPCA method are conducted. There are two important parameters in the KPCA algorithm, which are the number of principal components ( $n_c$ ) and  $\sigma^2$  in the Gaussian kernel function. The optimal

values of these parameters which computed using cross validation method are shown in Table 6.1. For the CD method parameters, the correlation dimension method and the mutual information method are used to selecting the embedding dimension ( $d$ ) and the time delay constant ( $m$ ), respectively. The optimal values of these parameters for each dataset are shown in Table 6.1. Also, choosing  $K$  is very important step in order to establish the local prediction model. So, equation (4.3.1) is used to calculate  $K$  as described in Section 4.3 where the parameters  $k_{\max}$  and  $\alpha$  are fixed for all test cases in this Chapter at 30% of  $N$  and 75, respectively.

Table 6.1: Phase reconstruction parameters for each dataset

Dataset	Coordination delay method parameters						KPCA parameters	
	Load time series		Temperature time series		Price time series		$\sigma^2$	$n_c$
	$d_1$	$m_1$	$d_2$	$m_2$	$d_3$	$m_3$		
North American Electric Utility	4	5	3	9	–	–	1.05	14
Victoria dataset	4	4	5	5	4	3	0.9	19

For all performed experiments, we quantified the prediction performance with mean absolute percentage error (MAPE) and normalised mean square error (NMSE). They are defined by equation (4.4.1) and equation (4.4.2), respectively.

### 6.5.1 Forecasting results using North American electric utility dataset

In this case, the dataset of the North American electric utility is used. The objective is to forecast the hourly load, from 16 up to 40 hour (steps) ahead for weekdays and from 16 up to 88 hour (steps) ahead for weekends during the test period which goes from November 1990 to March 1991.

To compare our proposed LWGMDH method with GMDH, local SVR, LWSVR based CD, LWSVR based KPCA and some published methods that employ the same dataset, we used the same experimental setup as used in [144] (Method D) and [30] (Method E) which use a multiple regression model called (EGRV). That is, the hourly load and temperature data from the month to be forecasted and from two month earlier, along with the data corresponding to the same window in the previous year are used as a training period.

To implement the SVR algorithm, there are some key parameters to calculate. These parameters are calculated as described in subsection 4.4.2. The values of these parameters are 0.70 and 1.81 for  $C$  and  $\sigma$ , respectively. In addition, for the GMDH method, the maximum number of nodes in each layer is chosen as 16 while the number of layers of the network is chosen automatically.

As in Chapter 5, the error of each day during the testing period is calculated. Then the average error of each day of the week (Monday to Sunday) during the testing period is calculated. Finally, the overall mean performance for the entire testing period for each model can be calculated. These results are summarised in Table 6.2.

Table 6.2: Comparison of LWGMDH method and other methods using the dataset of North American electric utility

Prediction method	MAPE	NMSE
Method D [144]	4.88	—
Method E [30]	4.73	—
GMDH	4.60	0.141
Local SVR	4.08	0.112
LWSVR based CD	3.62	0.083
LWSVR based KPCA	3.26	0.080
LWGMDH	2.91	0.071

These results illustrate that the LWGMDH method has a better prediction performance than than GMDH, local SVR, LWSVR based CD method, LWSVR based KPCA method, method D [144] and method E [30]. By using the LWGMDH, the performance (MAPE) is improved over method D [144], method E [30], GMDH, local SVR, LWSVR based CD and LWSVR based KPCA by 40.37%, 38.48%, 36.74%, 28.68%, 19.61% and 10.74%, respectively.

The average prediction MAPE of every day of the week (Monday to Sunday) during the whole testing period for each method are shown in Fig. 6.6. In addition, the MAPE of the whole testing period for the 24 hours is summarised in Table 6.3. It can be seen from these results that, the LWGMDH method gives better performance than other methods in all cases.

There are two peak periods in each day. The first one is the A.M. peak period from 7 A.M. to 9 A.M., while the second one is the P.M. peak period from 4 P.M. to 7 P.M. Figures. 6.7 and 6.8 show the average prediction MAPE for each method during these two peak periods for the whole testing period. From these figures, we can notice that the LWGMDH gives the best performance amongst all methods for both A.M. peak period and P.M. peak period. These results confirm the superiority of the LWGMDH method over other methods.

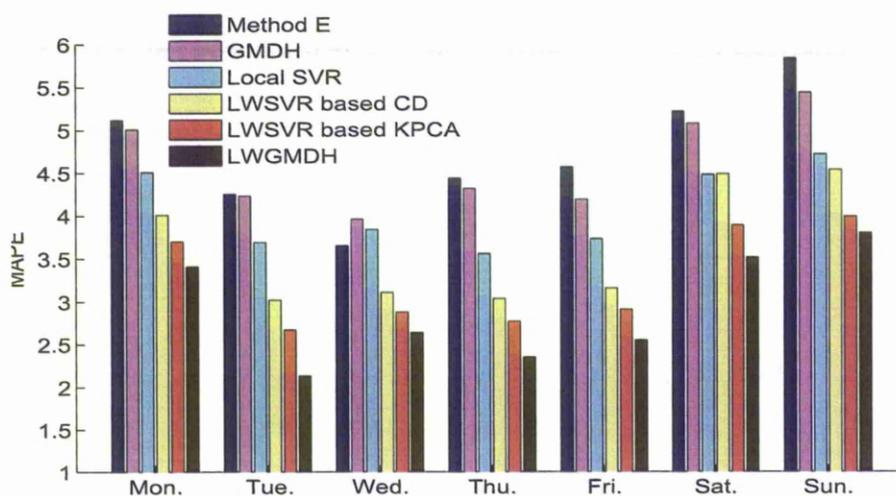


Figure 6.6: Average prediction MAPE of every day of the week during the whole testing period.

Table 6.3: Comparison of the LWGMDH method and other methods using the dataset of the North American electric utility (MAPE of the 24 Hours)

Hour	MAPE					
	Method E [30]	GMDH	Local SVR	LWSVR + CD	LWSVR + KPCA	LWGMDH
1	4.08	4.07	3.67	3.10	2.80	2.51
2	4.44	4.39	3.96	3.04	2.33	2.17
3	4.84	4.75	4.30	3.99	3.11	2.15
4	5.03	4.92	4.46	4.01	3.08	2.25
5	5.47	4.95	4.44	4.43	4.05	3.52
6	5.66	5.27	4.71	4.04	4.62	4.09
7	5.43	5.35	4.80	4.10	3.57	3.08
8	4.76	4.66	4.21	3.67	4.09	3.59
9	4.22	3.87	3.37	3.35	3.14	2.65
10	3.84	4.21	3.54	3.03	2.77	2.76
11	4.04	4.28	3.66	3.17	2.91	2.52
12	4.12	4.39	3.94	3.34	3.52	2.97
13	4.55	4.26	3.86	3.18	2.55	3.09
14	5.02	4.51	4.07	3.97	3.21	3.14
15	5.31	5.08	4.52	4.29	4.12	3.58
16	5.43	5.48	4.87	4.01	3.46	3.10
17	5.36	4.80	4.25	4.21	3.62	2.74
18	4.94	4.47	4.01	3.25	2.31	2.07
19	4.19	4.16	3.75	3.32	2.46	2.45
20	4.30	3.61	3.24	3.20	3.02	2.84
21	4.42	4.35	3.89	3.28	3.18	3.15
22	4.40	4.62	3.97	3.54	3.14	3.22
23	4.57	4.69	3.99	3.39	3.40	2.83
24	5.04	5.16	4.54	3.98	3.80	3.49
Avg.	4.73	4.60	4.08	3.62	3.26	2.91

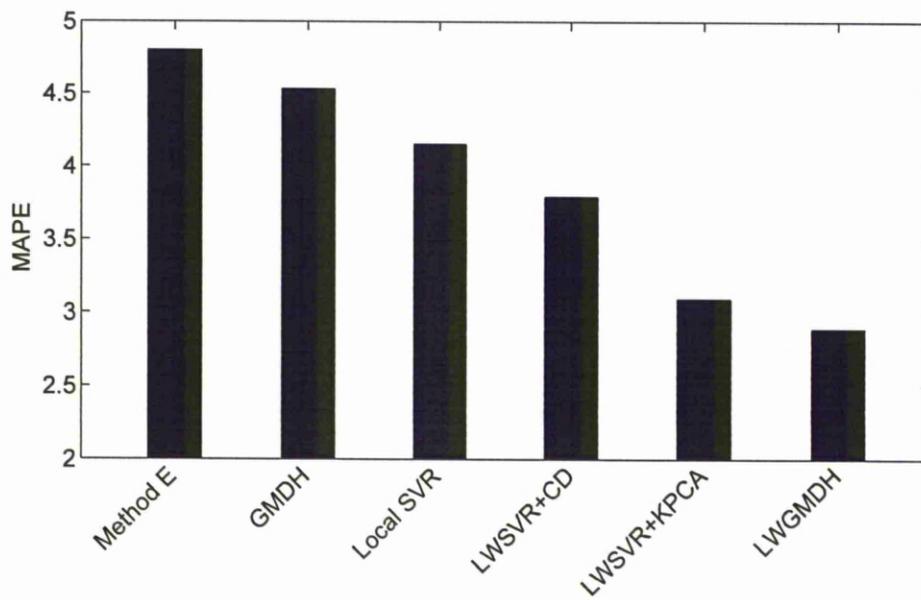


Figure 6.7: Average prediction MAPE of each method at A.M. peak period during the whole testing period.

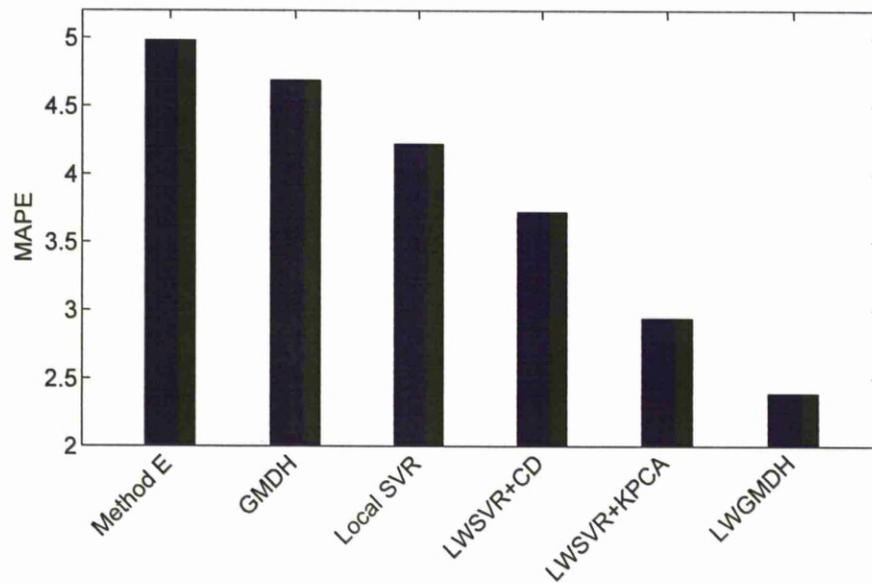


Figure 6.8: Average prediction MAPE of each method at P.M. peak period during the whole testing period.

### 6.5.2 Forecasting results using Victoria dataset

In this case, the publicly available load, price and temperature data (for two years 2002 and 2003) of the Victorian electricity market, Australia are used.

In this case the values of SVR's key parameters are also chosen as described in subsection 4.4.2. The values of these parameters are 10 and 1.10 for  $C$  and  $\sigma$ , respectively. Moreover, the maximum number of nodes in each layer of the GMDH network is chosen as 23 while the number of layers of the network is chosen automatically.

To verify the predictive ability of the proposed method, its performance is compared with GMDH, local SVR, LWSVR based CD, LWSVR based KPCA and two published methods. These methods are:

- Method D [144]: ANN with extended Bayesian training method.
- Method F [15]: ANN based similar days approach.

To make results comparable with the literature, we used the same experimental setup as used in [15] and [144]. That is the hourly load and temperature data from the month to be forecasted and from two month earlier, along with the data corresponding to the same window in the previous year are used as a training set.

First, the MAPE at each lead time (from 1 up to 6 hours ahead) is calculated for each method during the period from Monday, 1<sup>st</sup> September 2003 to Sunday 7<sup>th</sup> September 2003. These results are shown in Fig 6.9.

Fig. 6.9 shows that our proposed method gives the best performance amongst all methods. Moreover, as it is expected we can notice that the error is increasing proportionally with the lead time. This happens because uncertainty makes it more difficult to predict more distant events.

To further study the adaptiveness of the proposed method, the simulations for the whole September month (from 1<sup>st</sup> to 30<sup>th</sup> September) are also performed. The MAPE value for one hour and six hours ahead is obtained as 0.54 and 1.19, respectively. Table 6.4 shows the MAPE value of each method for all the simulated cases.

Table 6.4: Comparison among methods using the Victoria dataset (MAPE)

Hour ahead	Period	MAPE of each prediction method (%)						
		Model F [15]	Model D [144]	GMDH	Local SVR	LWSVR based CD	LWSVR based KPCA	LWGMMDH
1	01-07/09	0.56	0.49	0.48	0.47	0.45	0.40	0.38
2	01-07/09	0.83	0.72	0.70	0.68	0.65	0.57	0.54
3	01-07/09	1.00	0.82	0.80	0.77	0.72	0.65	0.62
4	01-07/09	1.15	0.91	0.91	0.86	0.79	0.71	0.67
5	01-07/09	1.20	0.99	0.97	0.91	0.82	0.77	0.69
6	01-07/09	1.30	1.07	1.06	0.98	0.90	0.79	0.72
1	01-30/09	0.77	-	0.73	0.68	0.64	0.57	0.54
6	01-30/09	2.06	-	1.91	1.63	1.48	1.30	1.19

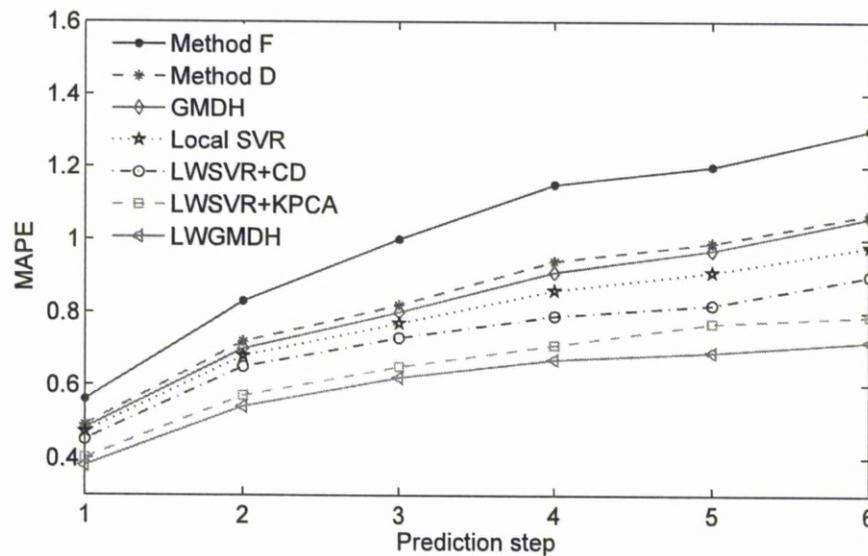


Figure 6.9: MAPE plotted against lead time for each method during the period from Monday, 1<sup>st</sup> September 2003 to Sunday, 7<sup>th</sup> September 2003.

These results show that our proposed LWGMDH method outperforms other methods. For the first week of September 2003, the LWGMDH method improves the accuracy over LWSVR based KPCA, LWSVR based CD, local SVR, GMDH, method D [144] and method F [15] by 5.00%, 15.56%, 19.15%, 20.83%, 22.45% and 32.14%, respectively for one hour ahead forecasting, while these improvements are 8.86%, 20.00%, 26.53%, 32.08%, 32.71% and 44.62%, respectively for six hour ahead forecasting. In addition, for September 2003, our proposed method improves the accuracy over LWSVR based KPCA, LWSVR based CD, local SVR, GMDH and method F [15] by 5.26%, 15.63%, 20.59%, 26.03% and 29.87%, respectively for one hour ahead forecasting, while these improvements are 8.46%, 19.59%, 26.99%, 37.70% and 42.23%, respectively for six hours ahead forecasting.

To further study the superiority of LWGMDH over other methods, the hourly load for one day ahead (from 1 up to 24 hour ahead) is predicted during two test months. These months are July 2003 (Winter season in Australia) and December 2003 (Summer season in Australia). The performance of the LWGMDH method is compared with LWSVR based KPCA, LWSVR based CD, local SVR, GMDH

methods. We calculate the MAPE of each day during each testing period. Then the average MAPE value of each day of the week (Monday to Sunday) during each testing period can be calculated. These results are shown in Figs. 6.10-6.11. Table 6.5 also summarises the overall mean performance of each method for each testing period.

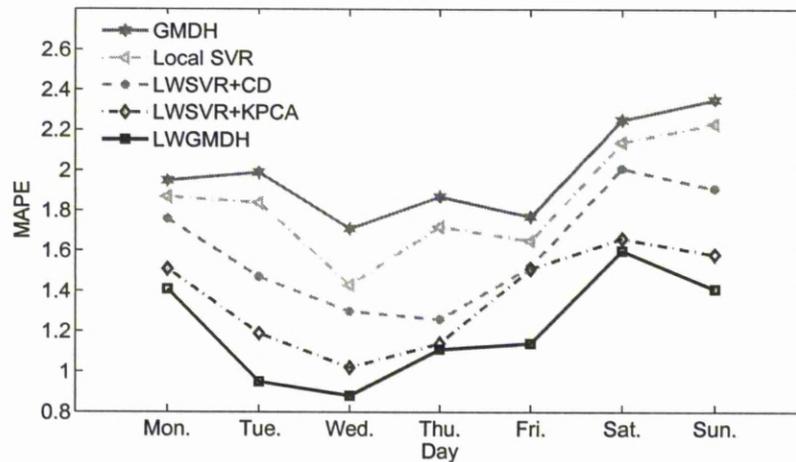


Figure 6.10: Average prediction MAPE of every day of the week during July 2003.

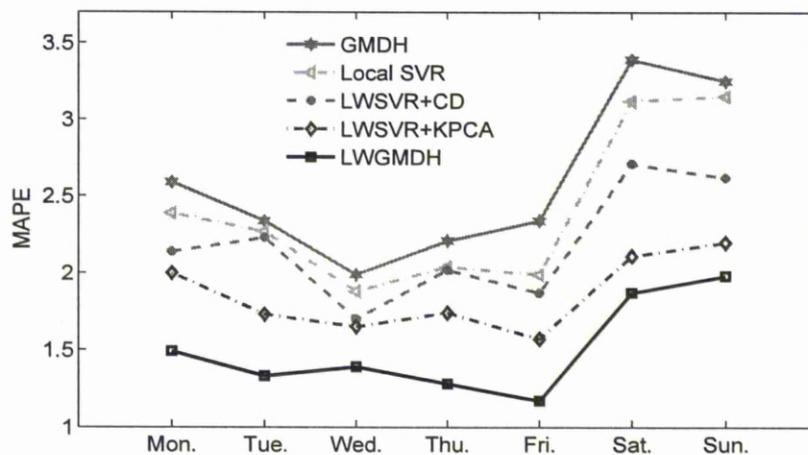


Figure 6.11: Average prediction MAPE of every day of the week during December 2003.

Table 6.5: Overall mean performance of each method for each testing period using the Victoria dataset

Prediction method	July 2003		December 2003	
	MAPE	NMSE	MAPE	NMSE
GMDH	2.04	0.060	2.25	0.070
Local SVR	1.84	0.051	2.15	0.064
LWSVR based CD	1.60	0.042	1.92	0.056
LWSVR based KPCA	1.37	0.036	1.70	0.049
LWGMDH	1.21	0.027	1.52	0.041

It can be seen from these results that the LWGMDH method gives better performance than GMDH, local SVR, LWSVR based CD and LWSVR based KPCA methods in all days. Moreover, the results show the superiority of the proposed method over other methods. The MAPE improvements of LWGMDH over methods are shown in Table 6.6.

Table 6.6: MAPE improvements of LWGMDH over other methods using the Victoria dataset

Prediction method	Improvements	
	July 2003	December 2003
LWGMDH	–	–
GMDH	40.69%	32.44%
Local SVR	34.24%	29.30%
LWSVR based CD	24.38%	20.83%
LWSVR based KPCA	11.68%	10.59%

Another observation from these results is that the MAPE of December is higher than July. This happens because December is unstable month in respect to load behaviour. This is due to the increase in power consumption because of

a gradual rise in temperature (the maximum and minimum temperatures were recorded to be nearly 40 and 12 °C, respectively.) and the celebration activities (Christmas and New Year).

## 6.6 Conclusions

In this chapter, a LWGMDH method for STLF has been proposed where the phase space is reconstructed using the KPCA method. Then, the neighbouring points are presented by Euclidian distance for each query point. These neighbouring points only can be used to train the GMDH where the coefficient parameters are calculated using WLS regression. Therefore, the most important points that are close to the query vector have larger weights than others. To improve the performance of the proposed method, the weighted distance algorithm is also used to optimise the weighting function's bandwidth.

According to the numerical results based on two different real world datasets, the proposed LWGMDH method gives a promising results for STLF. The numerical results show the superiority of the proposed LWGMDH method over the GMDH, local SVR, LWSVR based CD, LWSVR based KPCA and other published methods.

# Chapter 7

## Evolutionary Design of the Generalised Locally Weighted GMDH for STL<sub>F</sub>

### 7.1 Introduction

While providing a useful systematic design procedure, conventional GMDH also has some drawbacks [8]. First, it tends to generate quite complex polynomial for relatively simple systems (data). Second, it also tends to produce an exceedingly complex network when it comes to highly nonlinear systems due to its limited generic structure (quadratic two-variable polynomial) [156].

To overcome these drawbacks, a number of researchers have attempted to hybridise GMDH with some evolutionary optimisation techniques such as genetic algorithm (GA) [157], particle swarm optimisation [158], genetic programming [159], etc. Onwubolu [152] proposed a hybrid of differential evolution and GMDH and clearly showed that the proposed framework performs better than conventional GMDH method.

Iba *et al.* [160] proposed a hybrid of genetic programming and GMDH and showed that this hybrid method gives better performance than the conventional GMDH method. Zadeh *et al.* [161] proposed a new GMDH-type neural network where GA is deployed to design the whole architecture of the GMDH-type neural networks. In this method, the connectivity configuration is not limited to the

adjacent layers as in the conventional GMDH but the neurons in any layers can be connected to each other.

Moreover, the self-organising polynomial neural network (SOPNN) which is GMDH type algorithm was developed by Oh *et al.* [8]. SOPNN is more flexible than the conventional GMDH as each node in the network can have a different number of input variables as well as a different polynomial order. GA based SOPNN is proposed in [156] for nonlinear systems modelling and prediction. In [156], GA is used for each layer of the network to get the preferred nodes and their optimal design parameters (the number of input variables, polynomial order and input variables).

An evolutionary algorithm (EA) is a computer program that attempt to solve complex problems by mimicking the processes of Darwinian evolution. It operates within a population of individuals which are randomly generated. These individuals represent a particular solutions to a particular problem. The initial population evolves towards better solutions by using some operators. These operators are reproduction, crossover and mutation. The fitness value of an individual gives a measure of its performance for the given problem [162, 163].

In order to solve the STLTF problem, a new design approach of generalised LWGMDH (G-LWGMDH) based EA is introduced in this chapter. In the proposed model, the phase space is reconstructed based on KPCA method. In addition, the EA is used to design the whole architecture of the G-LWGMDH network, *i.e.*, the value of the weighting function's bandwidth, how many input variables are chosen to each node, which input variables are optimally chosen among input variables and what is the best type of the polynomials in each node. The G-LWGMDH method has been evaluated using two real world datasets and compared with some published methods. The results show that the proposed model exhibits superior performance to that of other methods.

The rest of this chapter is organised as follows: Section 7.2 reviews the EA algorithm. The implementation of G-LWGMDH based EA method is introduced in Section 7.3. Then the experimental results and comparisons with other methods are presented in Section 7.4. Finally, this chapter is concluded in Section 7.5.

## 7.2 Evolutionary algorithm (EA)

The EA is started with an initial population of individuals (generation) which are generated randomly. Every individual (chromosome) encodes a single possible solution to the problem under consideration. The fittest individuals are chosen by ranking them according to a pre-defined fitness function, which is evaluated for each member of this population. The individuals with high fitness values therefore represent better solution to the problem than individuals with lower fitness values. Following this initial process, the crossover and mutation operations are used where the individuals in the current population produce the children (offspring). These children are assigned fitness scores. A new population of individuals (generation) is then formed from the individuals in current population and the children. This new population becomes the current population and the iterative cycle is repeated until a termination condition is reached [164]. Fig. 7.1 shows the main flowchart that describes every EA applied for function optimisation.

EA posses a number of features as follows:

- EA is population based. It processes a whole collection of candidate solutions simultaneously.
- EA uses recombination to mix information of more candidate solution into a new one.
- EA is stochastic.

EA has a number of components or operators that must be specified in order to define a particular EA. the most important components, shown in Fig. 7.1, are representation, fitness function, selection method, crossover, mutation and termination. Each of these components will be discussed in the following subsections.

### 7.2.1 Representation and evaluation

The first design step is called representation as it amounts to specifying a mapping from a phenotypes (possible solutions within the original problem) onto

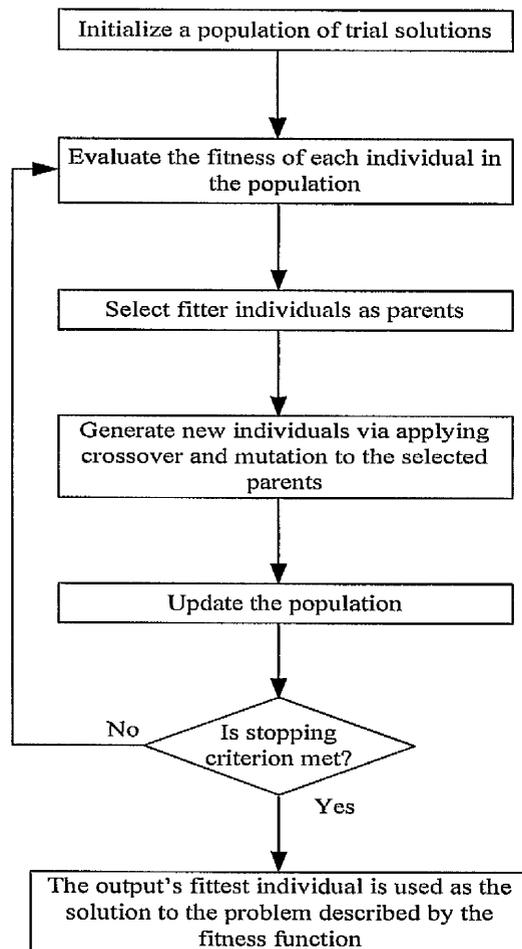


Figure 7.1: The main flowchart of EA.

a set of genotypes (the individuals within EA) which represent these phenotypes.

It is important to understand that the whole evolutionary search takes place in the genotype space. Then the solution, a good phenotype, can be obtained by decoding the best genotype after termination.

Many individual solutions (chromosomes) are randomly generated initially to form an initial population. There are no specific rules for the number of individuals in a population. The population size depends on the nature of the problem. The population covers the entire range of possible solutions (the search space).

In general, the individuals can be encoded based on bits, labels, integers, real numbers, logic based rules and any other finite alphabet adequate to encode the solution space supplied with an en/decoder function.

Once a population is generated, a fitness function is used to evaluate the individuals and sort them according to their fitness score. A fitness function is a function that assigns a quality measure to the individuals (chromosomes). Therefore, a particular individual may be ranked against all the other individuals. The individuals which are more optimal, are allowed to breed and mix their data by any of several techniques, producing a new generation that will hopefully be even better.

### 7.2.2 Selection

Selection is the next step of EA in which the better individuals based on their quality are chosen from a population to become parents of the next generation. Parent selection is responsible for pushing quality improvements. The parent selection is typically probabilistic. Probabilistic selection of individuals is repeatedly despatched in order to locate the candidates applicable to the EA operations. Thus, high quality individuals have a higher chance to become parents than those which have low quality.

There are various selection algorithms reported in [165]. The well known schemes are stochastic sampling with replacement, stochastic sampling with no or partial replacement, remainder stochastic sampling with replacement, etc.

In this thesis, the stochastic sampling with replacement scheme which is

widely used in literature under the name of “roulette wheel selection” is employed. This selection scheme may be implemented as follows [165]:

- The fitness function is evaluated for each individual in the population providing fitness values. These values are then normalised by dividing the fitness value of each individual by the sum of all fitness values, so that the sum of all resulting fitness values equals 1.
- The individuals are sorted by descending fitness values.
- Accumulated normalised fitness values are computed (the accumulated fitness value of an individual is the sum of its own fitness value plus the fitness values of all the previous individuals). The accumulated fitness of the last individual should of course be 1.
- A number  $r$  between 0 and 1 is chosen randomly.
- The selected individual is the first one whose accumulated normalised value is greater than  $r$ .
- This procedure is repeated until there are enough selected individuals

### 7.2.3 Crossover

The idea behind the crossover operator is to combine useful segments of different parents to form an offspring that benefits from advantageous bit combinations of both parents.

Once the parents are selected using the selection scheme, the strings are picked at random to form pairs with probability  $p_c$ . These pairs then exchange some of their data creating offspring by randomly selecting a position on the strings and exchanging each sub-string to one side of the position.

In discrete crossover, there are many types of crossover operator. they are one point, two point, uniform binomial crossover. The simplest crossover operator is the one point crossover [166] in which a crossing site is selected at random and then the sub-string that follows the site are exchanged. For example, two

possible parents could be:

Parent a:  $p_{a1}$   $p_{a2}$   $p_{a3}$   $p_{a4}$   $p_{a5}$   
 Parent b:  $p_{b1}$   $p_{b2}$   $p_{b3}$   $p_{b4}$   $p_{b5}$

One point crossover could possible yield the following spring:

Child a:  $p_{a1}$   $p_{a2}$  ||  $p_{b3}$   $p_{b4}$   $p_{b5}$   
 Child b:  $p_{b1}$   $p_{b2}$  ||  $p_{a3}$   $p_{a4}$   $p_{a5}$

In real valued crossover, there are different types of crossover operator such as intermediate arithmetical, line arithmetical, heuristic arithmetical [162]. In this thesis, the commonly used line arithmetical crossover operator is employed [164]. Where a random number  $r \in [0, 1]$  is generated. Then two offspring  $c_i, c_j$  are produced through the following linear combinations of two parental vectors  $p_i, p_j$ :

$$\begin{aligned} c_i &= r \times p_i + (1 - r) \times p_j \\ c_j &= (1 - r) \times p_i + r \times p_j \end{aligned} \quad (7.2.1)$$

This crossover produces offspring lying along the line segment joining the two parent points in the solution space.

#### 7.2.4 Mutation

By mutation, individuals are randomly altered. These variations (mutation steps) are mostly small. They will be applied to the variables of the individuals with a low probability  $p_m$  (mutation probability) which is set to small number to avoid disruption of the useful schemata. Normally, offspring are mutated after being created by crossover. It is intended to prevent premature convergence and loss of genetic diversity.

In case of binary encoding, the 0 or 1 is flipped to 1 or 0, respectively with a probability  $p_m$ . If the encoding is not binary but still based on a discrete alphabet, then a different value is chosen randomly from this alphabet to replace the current one with probability  $p_m$ .

In real valued encoding, there some different types of mutation such as uniform mutation, Gaussian mutation and adaptive non-uniform mutation. In this

thesis, a Gaussian mutation method is employed for each  $k^{\text{th}}$  gene  $p[k]$  as follows [162]:

$$c[k] = \mathcal{N}(p[k], \sigma^2) \quad (7.2.2)$$

### 7.2.5 Termination

After crossover and mutation, each of the population generated goes through a series of evaluation, selection, crossover and mutation. These procedures are repeated until a termination condition is reached. Different termination heuristics can be applied such as:

- Fixed number of generations reached.
- A solution is found that satisfies minimum criteria.
- The highest ranking solution's fitness is reaching or has reached a plateau such that successive iterations no longer produce better results.
- Combinations of the above.

## 7.3 Generalised LWGMDH (G-LWGMDH) algorithm based EA

The G-LWGMDH has a high level of flexibility as each node can have a different number of input variables as well as a different order of polynomial. Moreover, in the LWGMDH, the nodes in each layer are only connected to node in its adjacent layer. The G-LWGMDH removes this restriction. This means that the nodes in different layers including the input layer can be connected to others far away not only in the very adjacent layers.

The performance of the G-LWGMDH depends strongly on choosing the value of the weighting function's bandwidth, the number of input variables for each node, which input variables can be chosen among input variables for each node and what is the best type of the polynomials in each node. Therefore the designer must determine these parameters before the architecture of G-LWGMDH

is constructed but there is no guarantee that the obtained structure is the best one. In order to solve this problem, we use the EA which has been widely used as a parallel global search method for optimisation problems [167, 168].

### 7.3.1 Representation of chromosome for appropriate information of G-LWGMDH network

In the evolutionary design procedure of G-LWGMDH, the most important consideration is the representation strategy, that is how to encode the different parameters into the chromosome. In this work, each chromosome which represents the structure of the whole network consists of four sub-chromosomes.

We employ different coding methods for each sub-chromosome. The first sub-chromosome is a real coding for the bandwidth of the weighting function ( $h$ ). The second and third sub-chromosomes are a string of integer digits for the polynomial type and the number of inputs of each node, respectively. The length of the second and third sub-chromosomes is equal to the total number of nodes in the network which is initially determined by the user. The last sub-chromosome is a string of alphabetical digits of the input variables chosen to each node. The length of this sub-chromosome is equal to the sum of the numbers in the third sub-chromosome.

Fig. 7.2 illustrates an example of a chromosome which represents a network of G-LWGMDH consisting of three layers. Number of nodes of each layer are 3, 2 and 1 (output). This network is shown in Fig. 7.3. Different types of polynomial are used in this thesis. Table 7.1 shows these types for up to four inputs as an example [8, 156].

### 7.3.2 Fitness evaluation

The fitness function measures the performance of the model. It is quite important for evolving systems to find a good fitness measurement. The fitness ( $F$ ) of each entire string which represents a G-LWGMDH network is evaluated using mean square error defined as:

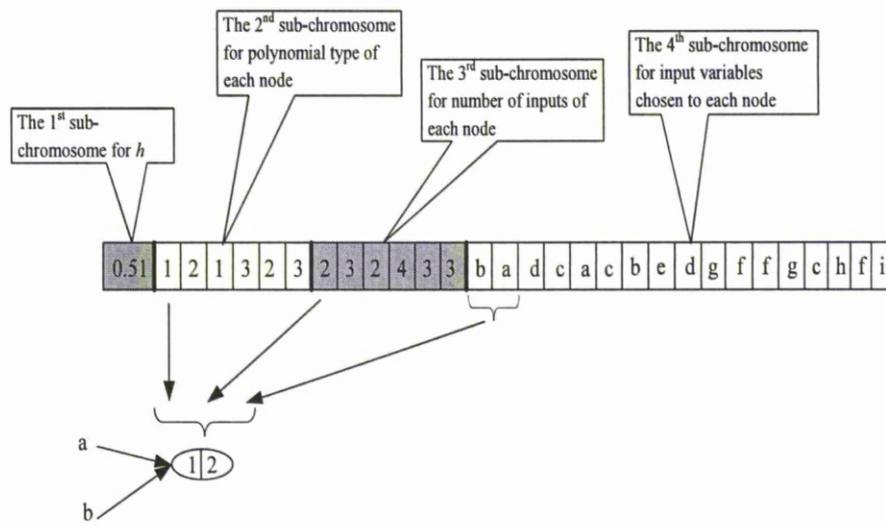


Figure 7.2: Structure of the chromosome which represents a G-LWGMDH network.

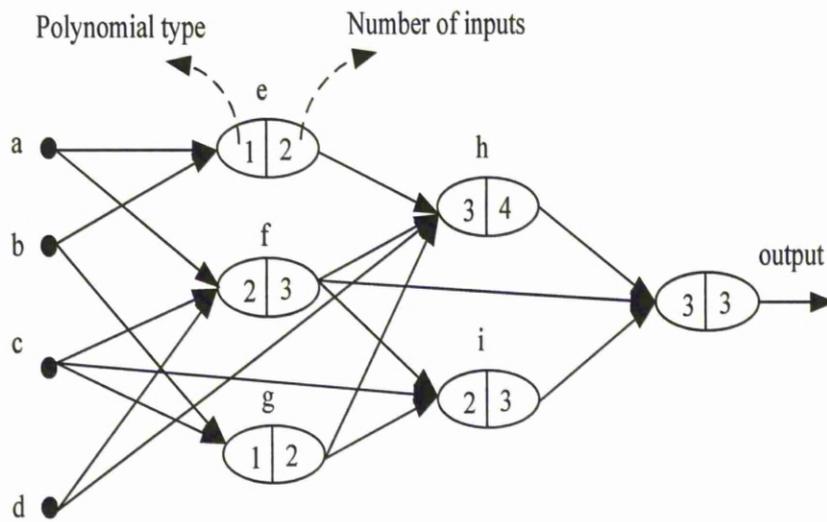


Figure 7.3: G-LWGMDH network.

Table 7.1: Different types of polynomials

Order of the polynomial	Number of inputs		
	2	3	4
1 (type 1)	Bilinear	Trilinear	Tetralinear
2 (type 2)	Biquadratic	Triquadratic	Tetraquadratic
2 (type 3)	Modified biquadratic	Modified triquadratic	Modified tetraquadratic

- Bilinear =  $c_0 + c_1x_1 + c_2x_2$ .
- Biquadratic =  $c_0 + c_1x_1 + c_2x_2 + c_3x_1x_2 + c_4x_1^2 + c_5x_2^2$ .
- Modified biquadratic =  $c_0 + c_1x_1 + c_2x_2 + c_3x_1x_2$ .
- Trilinear =  $c_0 + c_1x_1 + c_2x_2 + c_3x_3$ .
- Triquadratic =  $c_0 + c_1x_1 + c_2x_2 + c_3x_3 + c_4x_1x_2 + c_5x_1x_3 + c_6x_2x_3 + c_7x_1^2 + c_8x_2^2 + c_9x_3^2$ .
- Modified Triquadratic =  $c_0 + c_1x_1 + c_2x_2 + c_3x_3 + c_4x_1x_2 + c_5x_1x_3 + c_6x_2x_3$ .
- Tetralinear =  $c_0 + c_1x_1 + c_2x_2 + c_3x_3 + c_4x_4$ .
- Tetraquadratic =  $c_0 + c_1x_1 + c_2x_2 + c_3x_3 + c_4x_4 + c_5x_1x_2 + c_6x_1x_3 + c_7x_1x_4 + c_8x_2x_3 + c_9x_2x_4 + c_{10}x_3x_4 + c_{11}x_1^2 + c_{12}x_2^2 + c_{13}x_3^2 + c_{14}x_4^2$ .
- Modified Tetraquadratic =  $c_0 + c_1x_1 + c_2x_2 + c_3x_3 + c_4x_4 + c_5x_1x_2 + c_6x_1x_3 + c_7x_1x_4 + c_8x_2x_3 + c_9x_2x_4 + c_{10}x_3x_4$ .

$$F = \frac{1}{N_{va}} \sum_{i=1}^{N_{va}} (y_i - \hat{y}_i)^2 \quad (7.3.1)$$

where  $N_{va}$  is the number of points in the validation set,  $y_i$  and  $\hat{y}_i$  are the actual and output values, respectively.

### 7.3.3 EA operators for G-LWGMDH network reproduction

The operators of crossover and mutation can now be implemented to produce two offspring from two parents which are chosen using the roulette wheel selection method. For the first sub-chromosome which represents  $h$ , the line arithmetical crossover is used as follows [164]:

$$\begin{aligned}c_1 &= r \times h_1 + (1 - r) \times h_2 \\c_2 &= (1 - r) \times h_1 + r \times h_2\end{aligned}\tag{7.3.2}$$

where  $h_1$  and  $h_2$  are two parents,  $c_1$  and  $c_2$  are two offspring and  $r$  is a random generated uniform number ( $r \in [0, 1]$ ).

The crossover operator for the second and third sub-chromosomes is simply accomplished by exchanging the tail of each two sub-chromosomes from a randomly chosen point. While the change of the fourth sub-chromosome follows the change in the third one. Fig. 7.4 shows an example of crossover operator for two chromosomes in G-LWGMDH networks. These networks after the crossover are shown in Fig. 7.5.

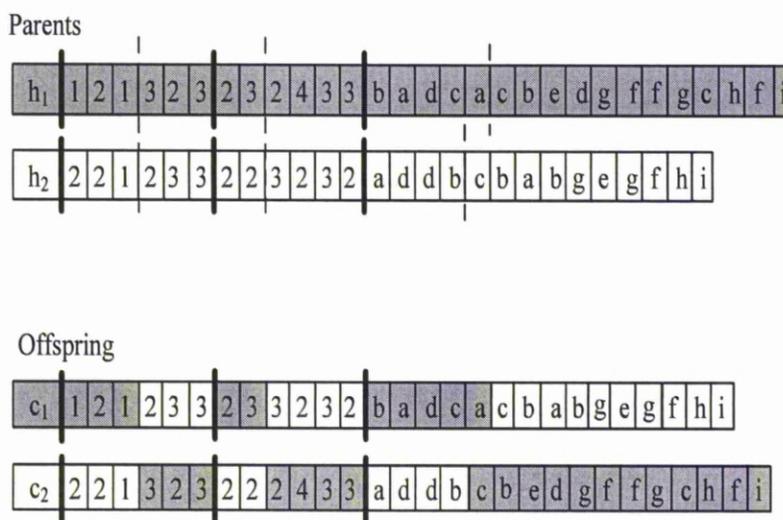


Figure 7.4: Crossover operation for two individuals in G-LWGMDH.

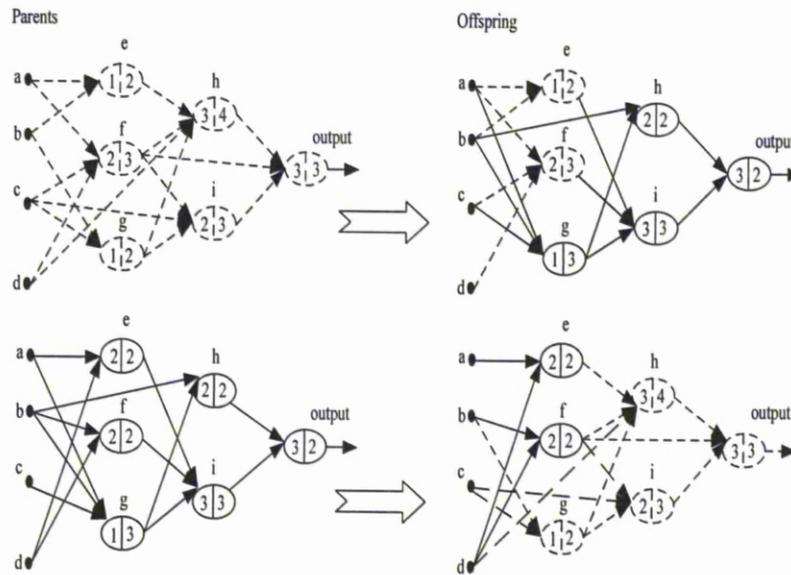


Figure 7.5: Crossover operation on two G-LWGMDH networks.

Similarly, the mutation operation can contribute effectively to the diversity of the population. In the first sub-chromosome, the Gaussian mutation (equation 7.2.2) is used. While, this operation is simply accomplished by changing one or more digits as genes in the second and third sub-chromosomes to another possible digit. In addition, the change of the fourth sub-chromosome follows the change in the third one. Fig. 7.6 shows an example of mutation operator for one chromosome in G-LWGMDH network. After that, each of the population generated goes through a series of evaluation, reproduction, crossover and mutation.

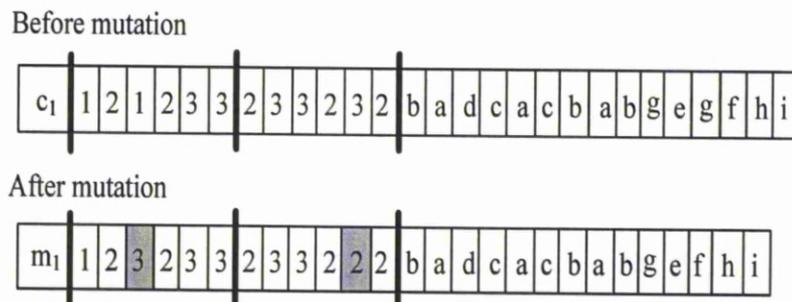


Figure 7.6: Mutation operation for an individual in G-LWGMDH.

In every generation, the chromosome that has the worst fitness value is replaced by the chromosome that has the best fitness value in the previous generation. These procedures are repeated until a termination condition is reached. In this thesis, we use a predetermined maximum number of generation as a termination condition. After the termination condition is satisfied, the chromosome which gives the best performance in the last generation is selected as the output G-LWGMDH network.

Overall, the framework of the design procedure of the G-LWGMDH based EA comes as a sequence of the following steps.

- Step 1: Reconstruct the time series: Load the multivariate time series dataset  $X = (x_1(t), x_2(t), \dots, x_M(t))$ , ( $t = 1, 2, \dots, N$ ). Using the KPCA method to calculate the number of principal components of each dataset (we set the time delay constant of all datasets equal to 1). Then, reconstruct the multivariate time series using these values.
- Step 2: Determine initial information for constructing the G-LWGMDH structure. The initial information for the G-LWGMDH structure can be determined in the following manner:
  - The value of the weighting function's bandwidth ( $h$ ).
  - The maximum number of layers and the number of nodes per layer.
  - The maximum number of input variables arriving at each node and the polynomial types used.
  - The EA parameters. These parameters are population size, crossover rate, mutation rate and the stopping criterion. In this thesis, the maximum number of generations is used as the stopping criterion.
- Step 3: Form a training and validation data: The input dataset after reconstruction  $\tilde{X}$  is divided into two parts, that is a training  $\tilde{X}_{tr}$  dataset and validation  $\tilde{X}_{va}$  dataset. The size of the training dataset is  $N_{tr}$  while the size of the validation dataset is  $N_{va}$ .
- Step 4: For each query point  $x_q$ , choosing the  $K$  nearest neighbours of this query point using the Euclidian distance between  $x_q$  and each point in  $\tilde{X}_{tr}$

( $1 < K \ll N_{tr}$ ). These points only will be used to train the G-LWGMDH network.

- Step 5: Generation of initial population: The weighting function's bandwidth, the polynomial type of each node, the number of inputs of each node and the input variables chosen to each node are encoded into a chromosome as described in Section 7.3.1. These chromosomes are all randomly initialised.
- Step 6: Estimate the coefficient parameters of each node: The vector of coefficient parameters is derived by minimising the locally weighted mean squared error (equation 6.4.1) using the nearest neighbours points only. The weighted least square solution of equation (6.4.1) is given by equation (6.4.3). This procedure is implemented repeatedly for all nodes of each chromosome.
- Step 7: Evaluation: each chromosome is evaluated and has its fitness value as described in Section 7.3.2. Then, to produce the next generation, the selection, crossover and mutation operations are carried out as described in Section 7.3.3.
- Step 8: Check the stopping criterion: The modelling can be terminated when the stopping criterion is reached. If the stopping criterion is not satisfied, the model has to be expanded. The steps 6 to 7 can be repeated until the stopping criterion is satisfied.
- Step 9: After the evolution process, the chromosome which has the best performance in the last generation is selected as the final structure of the network. The output of the selected network is the predicted value of the current query point.
- Step 10: Then, the steps 4 to 9 can be repeated until the future values of different query points are all acquired.

Figure 7.7 presents the computation procedure of the proposed method.

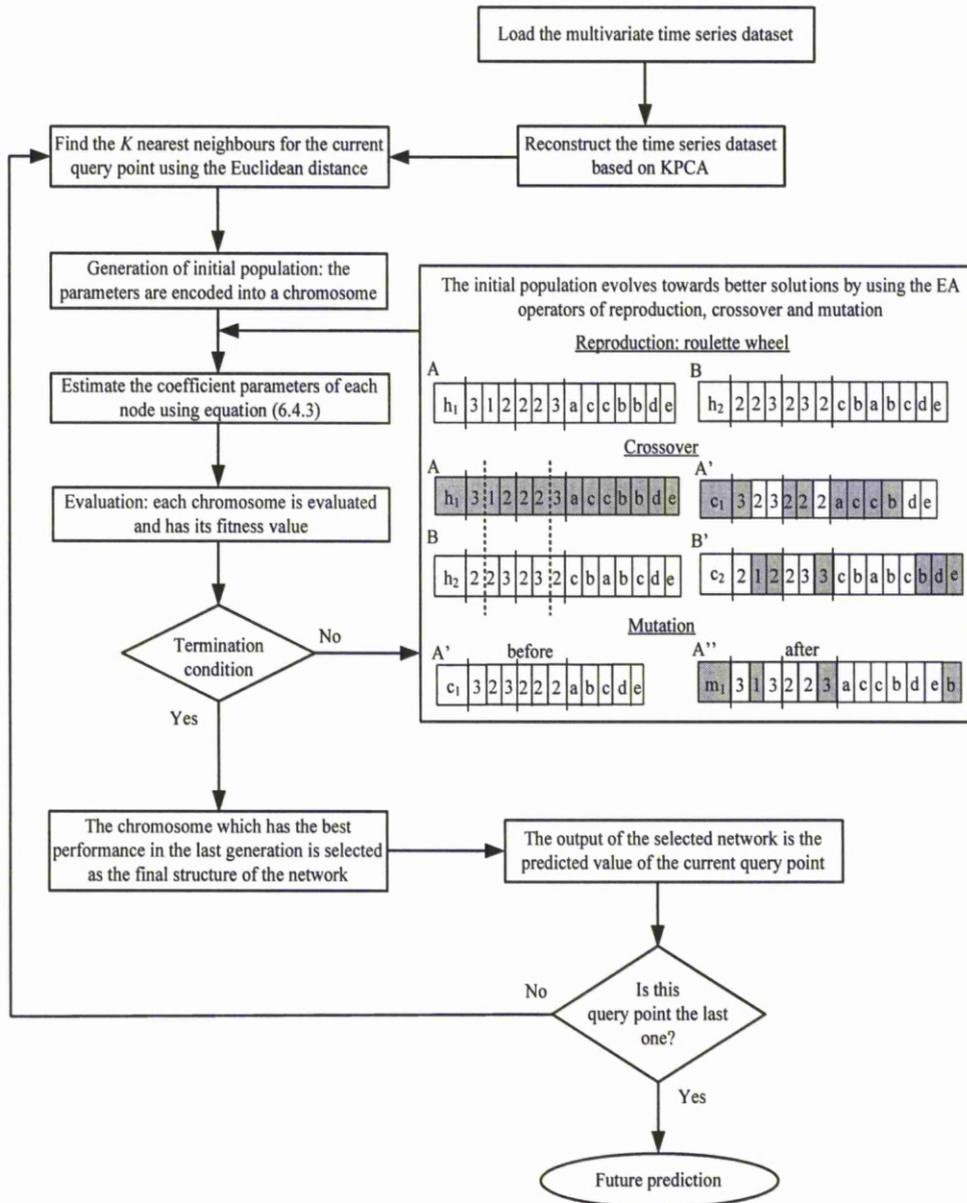


Figure 7.7: Flowchart of the G-LWGMDH based EA method.

## 7.4 Numerical results

Two different datasets are used to evaluate the performance of the proposed G-LWGMDH based EA method. The first dataset is the hourly electricity load in New York city and temperature data observed at Central Park [169] (for two years 2003 and 2004). While the second one is the historical load, temperature and price data (for two years 2002 and 2003) from the Victorian electricity market in Australia as presented in Section 6.5.

For the first dataset, Fig. 7.8 illustrates the electricity demand of New York City from 1<sup>st</sup> June 2003 to 31<sup>st</sup> December 2003.

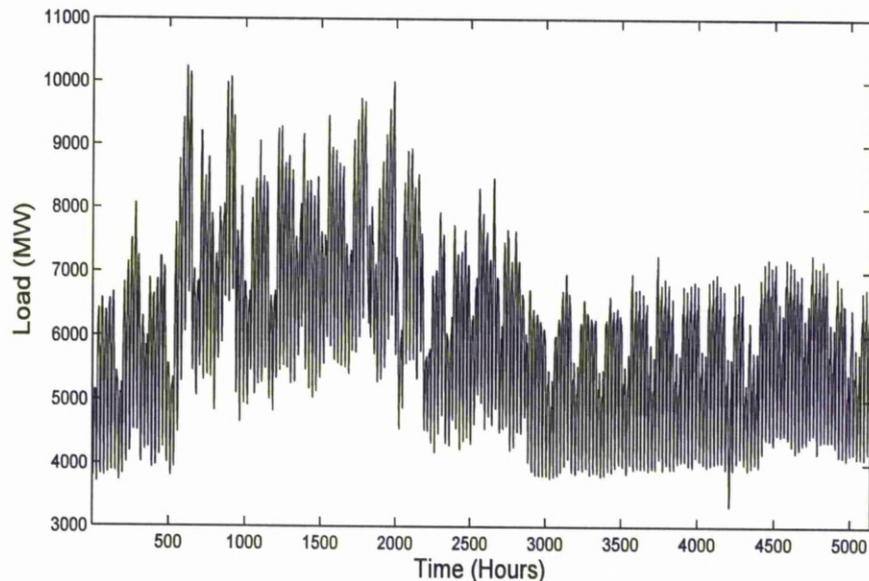


Figure 7.8: Hourly electricity demand in New York city from 1<sup>st</sup> June 2003 to 31<sup>st</sup> December 2003.

According to Fig. 7.8, it is clear that the load has multiple seasonal patterns. The electricity demand in summer period is higher than the electricity demand in winter period. This is because the gradual rise in temperature in the summer over the winter. In addition, it is also affected by calendar effect, *i.e.*, weekends and holidays.

To show the effectiveness of the proposed method, numerical simulations comparing with conventional GMDH (as a global method), local SVR, LWSVR

based KPCA and LWGMDH method are conducted. Table 7.2 shows the optimal values of the KPCA parameters. In addition, equation (4.3.1) is used to calculate  $K$  as described in Section 4.3 where the parameters  $k_{\max}$  and  $\alpha$  are fixed for all test cases in this Chapter at 50% of  $N$  and 90, respectively. Table 7.3 summarises the design parameters of EA and the initial design information for the LWGMDH network for all test cases.

Table 7.2: KPCA parameters of each dataset

Dataset	$\sigma^2$	$n_c$
Victoria dataset	0.90	19
New York	1.30	15

Table 7.3: Design parameters of the G-LWGMDH based EA method

	Parameters	
EA	Maximum generation	100
	Population size	40
	Crossover rate	0.8
	Mutation rate	0.09
LWGMDH network	Number of layers	5
	Number of nodes per layer	12,9,6,3,1
	Number of inputs to be selected	2 ~ 5
	polynomial type	1 ~ 3
	Weighting function's bandwidth	$0 < h \leq 1$

To choose the suitable values of SVR's key parameters, the same procedures which described in subsection 4.4.2 are used. The values of these parameters are  $C = 13$  and  $\sigma = 0.90$  for the New York dataset, while these values are  $C = 10$  and  $\sigma = 1.10$  for the Victoria dataset.

In addition, for the GMDH network, the maximum number of nodes in each

layer of the network is chosen as 18 and 23 for the New York dataset and the Victoria dataset respectively. The number of layers of the GMDH network is chosen automatically so that it may generate quite complex polynomial for relatively simple data. The proposed G-LWGMDH based EA method solves this problem by choosing the best number of layers and the best number of nodes in each layer using EA. The maximum number of layers and the maximum number of nodes per layer for both datasets are shown in Table 7.3.

Moreover, all nodes in the GMDH network has limited generic structure (quadratic two-variable polynomial), so that it tends to produce an exceedingly complex network when it comes to highly nonlinear systems. This problem can be overcome in the G-LWGMDH based EA method by using different polynomial order and different number of inputs for each node. The range of these parameters are shown in Table 7.3 for both dataset.

#### 7.4.1 New York dataset

In this case, the hourly electricity load in New York City and temperature data observed at Central Park have been considered [169]. The performance of the proposed G-LWGMDH based EA method is compared with GMDH (as a global method), local SVR, LWSVR based KPCA, LWGMDH, the prediction of New York Independent System Operator (NYISO) [169] and an adaptive two-stage hybrid network with self-organised map and support vector machine (SVM) [13].

To make results comparable with other methods, the training and testing periods are used as described in [13]. In this case, four typical months have been used for forecasting. The first one corresponds to January 2004 which is a winter month, the second one corresponds to April which is a spring month, the third one corresponds to July 2004 which is a summer month while the fourth one corresponds to October 2004 which is an autumn month.

The hourly load and temperature data from 1<sup>st</sup> January 2003 to 31<sup>st</sup> December 2003 are used to forecast the winter testing month. In addition, the hourly load and temperature data from 1<sup>st</sup> April 2003 to 31<sup>th</sup> March 2004 are used to forecast the spring testing month. Moreover, the hourly load and temperature

data from 1<sup>st</sup> July 2003 to 30<sup>th</sup> June 2004 are used to forecast the summer testing month. Whereas, the hourly load and temperature data from 1<sup>st</sup> October 2003 to 30<sup>th</sup> September 2004 are used to forecast the autumn testing month.

First, the MAPE and NMSE of each day during the four test months are calculated. Then the average MAPE and NMSE values of each method for the four test months are calculated. Also, we calculate the MAPE and NMSE of the forecasting load published by NYISO [169] in the same period. These results are shown in Table 7.4 and depicted in Figs. 7.9-7.12.

These results show that the proposed method outperforms other methods. Table 7.5 shows the MAPE improvements of the G-LWGMDH based EA over NYISO [169], hybrid network [13], GMDH, local SVR, LWSVR and LWGMDH for each testing month. Another observation from these results is that the MAPE of July is higher than January. This is due to the increase in power consumption because of a gradual rise in temperature.

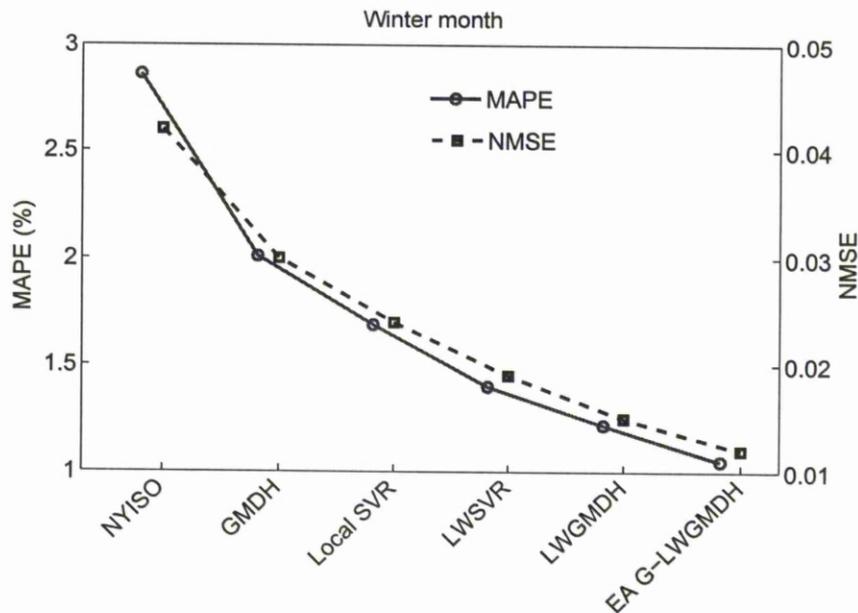


Figure 7.9: Comparison of the G-LWGMDH based EA method and other methods using MAPE and NMSE for winter month.

Table 7.4: Overall mean performance of each method for each testing month using the New York dataset

Prediction method	Error of each testing month											
	Winter month		Spring month		Summer month		Autumn month					
	MAPE	NMSE	MAPE	NMSE	MAPE	NMSE	MAPE	NMSE				
NYISO [169]	2.86	0.042	2.29	0.034	3.55	0.074	2.36	0.032				
Hybrid network [13]	1.82	–	–	–	2.29	–	–	–				
GMDH	2.01	0.030	1.88	0.026	2.54	0.057	1.87	0.025				
Local SVR	1.69	0.024	1.49	0.021	2.17	0.048	1.58	0.022				
LWSVR	1.40	0.019	1.26	0.017	1.78	0.039	1.29	0.018				
LWGMDDH	1.22	0.015	1.14	0.013	1.59	0.033	1.16	0.016				
G-LWGMDDH based EA	1.05	0.012	0.95	0.011	1.32	0.027	0.98	0.013				

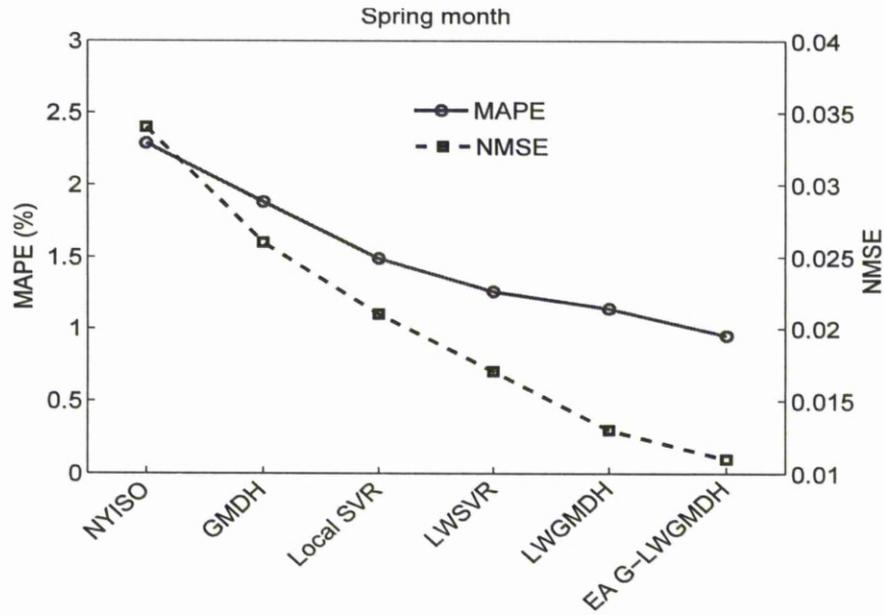


Figure 7.10: Comparison of the G-LWGMDH based EA method and other methods using MAPE and NMSE for spring month.

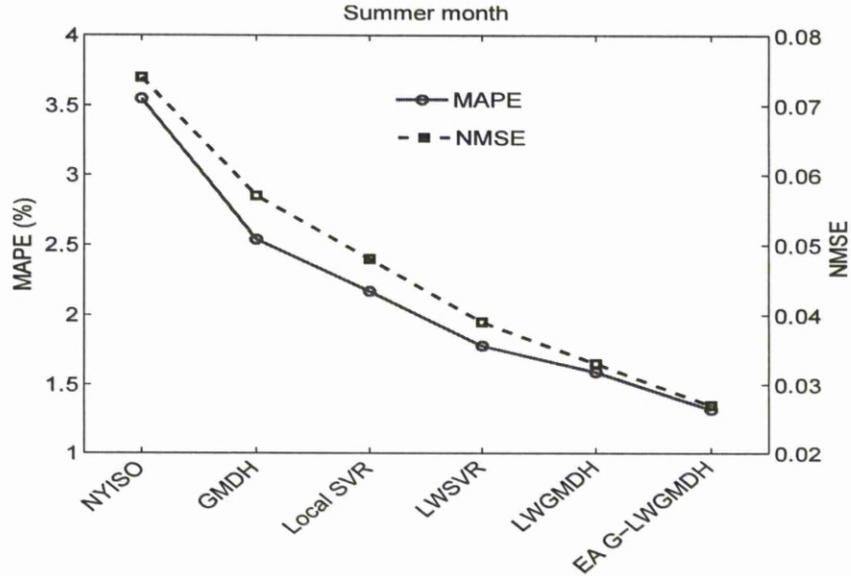


Figure 7.11: Comparison of the G-LWGMDH based EA method and other methods using MAPE and NMSE for summer month.

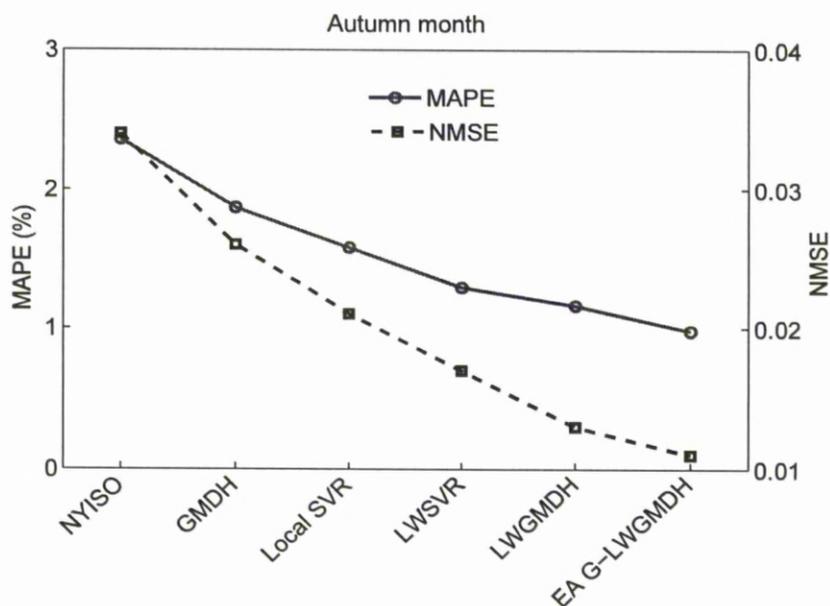


Figure 7.12: Comparison of the G-LWGMDH based EA method and other methods using MAPE and NMSE for autumn month.

Table 7.5: MAPE improvements of G-LWGMDH based EA over other methods using the New York dataset

Prediction method	Improvements			
	Winter	Spring	Summer	Autumn
G-LWGMDH based EA	–	–	–	–
NYISO [169]	63.29%	58.52%	62.82%	58.47%
Hybrid network [13]	42.31%	–	42.36%	–
GMDH	47.76%	49.47%	48.03%	47.59%
Local SVR	37.87%	36.24%	39.17%	37.97%
LWSVR	25.00%	24.60%	25.84%	24.03%
LWGMDH	13.93%	16.67%	16.98%	15.52%

In addition, the MAPE of the whole testing period (January 2004, April 2004, July 2004 and October 2004) for the 24 hours is summarised in Table 7.6. From these results, we can notice that the proposed method exhibits a better performance than other methods in all cases.

Figs. 7.13-7.16 show the actual load and one day ahead forecasted load values using G-LWGMDH based EA method of one week, as an example, of each testing month. The results of Figs. 7.13-7.16 show that the G-LWGMDH based EA method's prediction values are very close to the actual values.

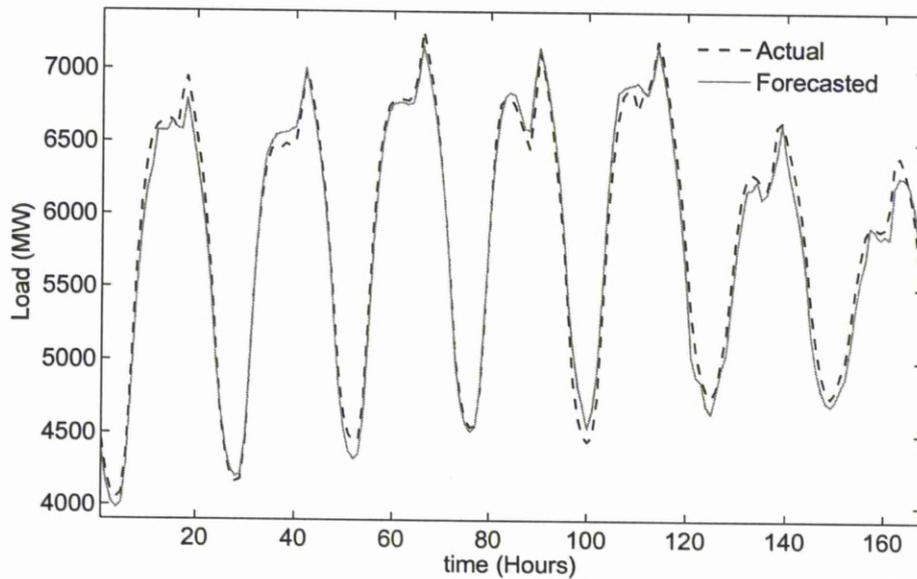


Figure 7.13: One day ahead forecasted and actual load from 5<sup>th</sup> January 2004 to 11<sup>th</sup> January 2004 using the New York dataset.

Table 7.6: Comparison of the G-LWGMDH based EA method and other methods using the New York dataset (MAPE of the 24 Hours)

Hour	MAPE (%)					
	NYISO [169]	GMDH	Local SVR	LWSVR	LWGMDH	G-LWGMDH based EA
1	3.07	2.06	1.61	1.36	1.20	1.12
2	3.03	2.65	1.91	1.77	1.60	1.32
3	3.00	2.39	1.84	1.62	1.35	1.17
4	2.92	2.08	1.69	1.40	1.33	1.14
5	2.87	1.68	1.58	1.41	1.37	1.18
6	2.78	1.66	1.19	1.08	0.88	0.73
7	2.71	1.57	1.29	1.23	1.22	1.03
8	2.67	2.20	1.35	1.16	1.14	1.01
9	2.70	1.67	1.24	1.15	0.92	0.84
10	2.58	1.97	1.42	1.17	0.99	0.79
11	2.62	1.91	1.88	1.28	1.26	1.09
12	2.54	2.07	1.82	1.34	1.03	0.92
13	2.61	1.81	1.71	1.24	1.15	1.11
14	2.69	1.63	1.45	1.32	1.13	1.02
15	2.73	1.72	1.57	1.44	1.40	1.26
16	2.85	1.96	1.68	1.49	1.36	1.20
17	2.88	2.05	1.67	1.39	1.06	0.74
18	2.79	2.42	1.83	1.68	1.61	1.29
19	2.79	2.34	1.94	1.45	1.25	1.08
20	2.54	2.33	2.16	1.63	1.29	0.96
21	2.65	2.52	2.24	1.58	1.47	1.23
22	2.77	2.60	2.11	1.70	1.58	1.25
23	2.77	2.23	2.04	1.66	1.53	1.16
24	2.96	2.37	2.18	1.67	1.48	1.22
Average	2.77	2.08	1.73	1.43	1.28	1.08

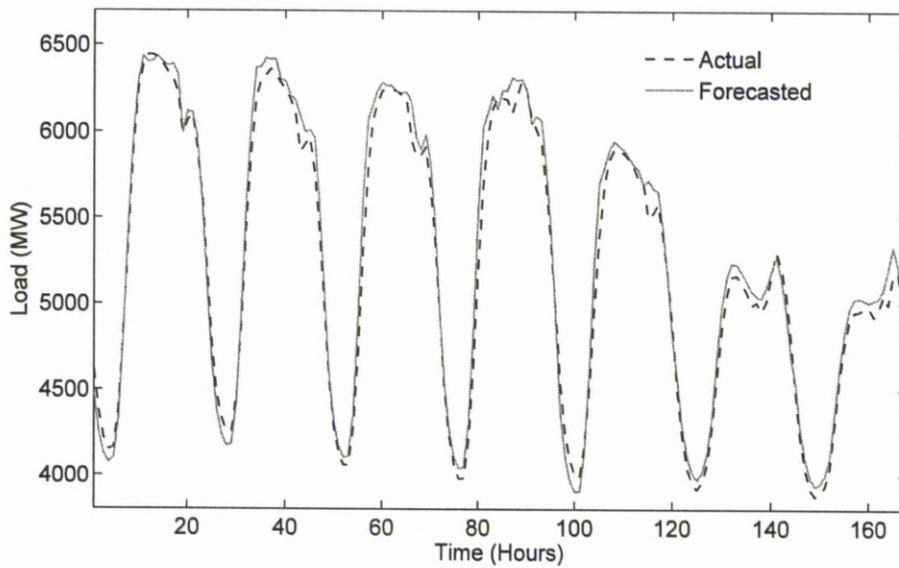


Figure 7.14: One day ahead forecasted and actual load from 5<sup>th</sup> April 2004 to 11<sup>th</sup> April 2004 using the New York dataset.

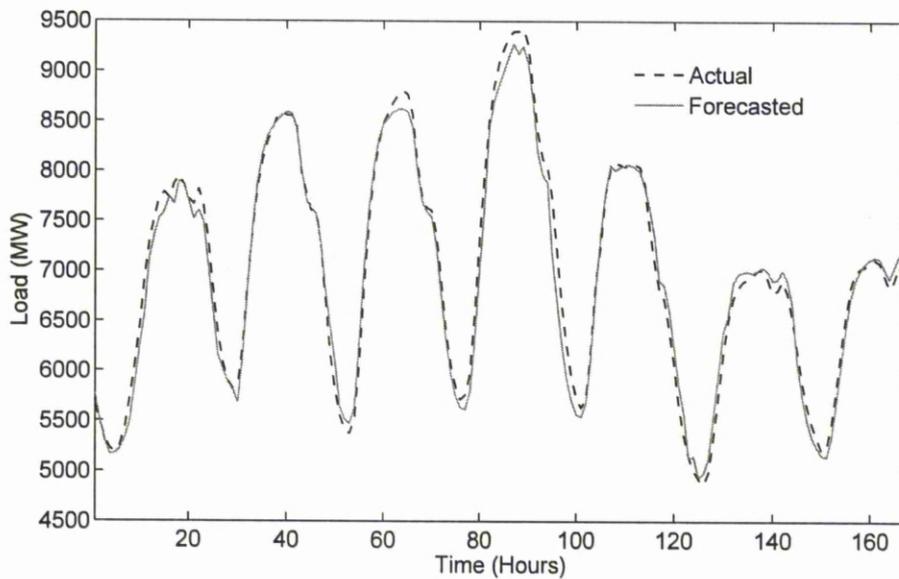


Figure 7.15: One day ahead forecasted and actual load from 5<sup>th</sup> July 2004 to 11<sup>th</sup> July 2004 using the New York dataset.

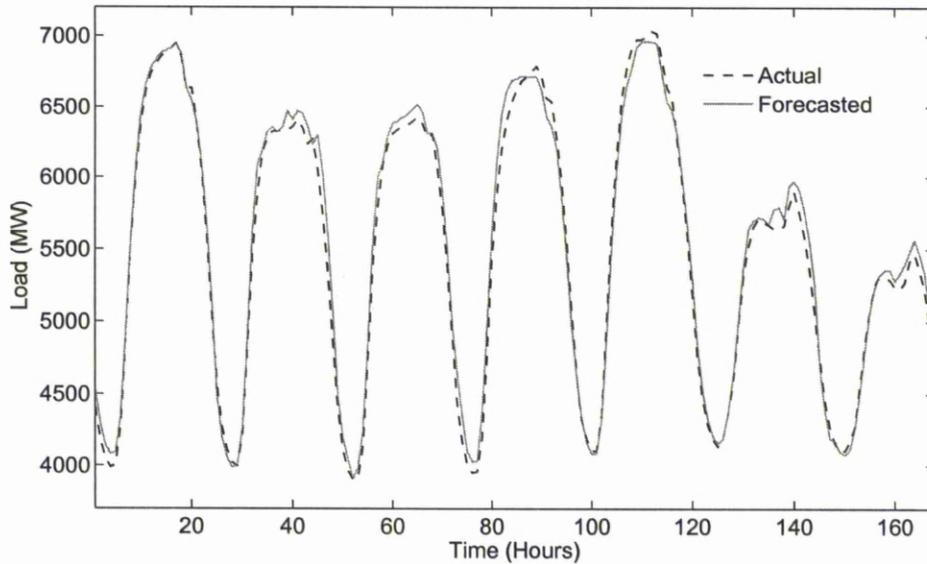


Figure 7.16: One day ahead forecasted and actual load from 4<sup>th</sup> October 2004 to 10<sup>th</sup> October 2004 using the New York dataset.

### 7.4.2 Victoria dataset

In this case, we use the same experimental setup as used in [15] and [144] in order to make results comparable with the literature. That is the hourly load and temperature data from the month to be forecasted and from two month earlier, along with the data corresponding to the same window in the previous year are used as a training set. The performance of the proposed G-LWGMDH based EA is compared with GMDH, Local SVR, LWSVR based KPCA, LWGMDH and two published methods (Method E [144] and Method F [15]).

As in Chapter 6, we calculate the MAPE at each lead time (from 1 up to 6 hours ahead) for each method during the period from Monday, 1<sup>st</sup> September 2003 to Sunday 7<sup>th</sup> September 2003. These results are shown in Fig 7.17.

Figure 7.17 shows that the G-LWGMDH based EA method gives better performance than other methods. In addition, the hourly load for one day ahead is predicted during four test months. These months are April 2003 (Autumn season in Australia), July 2003 (Winter season in Australia), October 2003 (Spring season in Australia) and December 2003 (Summer season in Australia).

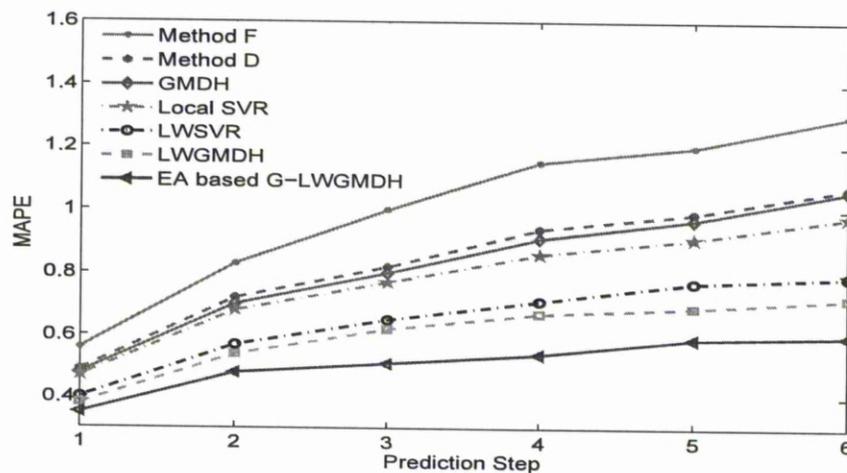


Figure 7.17: MAPE plotted against lead time for each method during the period from Monday, 1<sup>st</sup> September 2003 to Sunday, 7<sup>th</sup> September 2003 for the Victoria dataset.

The performance of the G-LWGMDH based EA method is compared with GMDH, local SVR, LWSVR based KPCA and LWGMDH methods. These results are shown in Table 7.7 which summarises the overall mean performance of each method for each testing month. In addition, the MAPE of each day during each testing period is calculated. Then the average MAPE value of each day of the week during the whole testing period (April 2003, July 2003, October 2003 and December 2003) can be calculated. These results are shown in Fig. 7.18.

From these results, it can be seen that the proposed method outperforms other methods. The proposed method has achieved the best results. Moreover, the MAPE of December is higher than other months. This happens due to the increase in power consumption because of a gradual rise in temperature and the celebration activities (Christmas and New Year). The MAPE improvements of the G-LWGMDH based EA over other methods are shown in Table 7.8.

Figures. 7.19- 7.22 show the actual load and one day ahead forecasted load values using G-LWGMDH based EA method of one week, as an example, of each testing month. The results of Figs. 7.19-7.22 show that the G-LWGMDH based EA method's prediction values are very close to the actual values.

Table 7.7: Overall mean performance of each method for each testing month using the Victoria dataset

Prediction method	Error of each testing month											
	April 2003			July 2003			October 2003			December 2003		
	MAPE	NMSE		MAPE	NMSE		MAPE	NMSE		MAPE	NMSE	
GMDH	1.97	0.059		2.04	0.060		2.02	0.060		2.25	0.070	
Local SVR	1.71	0.049		1.84	0.051		1.79	0.055		2.15	0.064	
LWSVR	1.22	0.028		1.37	0.036		1.33	0.032		1.70	0.049	
LWGMDDH	1.09	0.020		1.21	0.027		1.20	0.025		1.52	0.041	
G-LWGMDDH based EA	0.89	0.014		1.01	0.018		0.97	0.017		1.23	0.028	

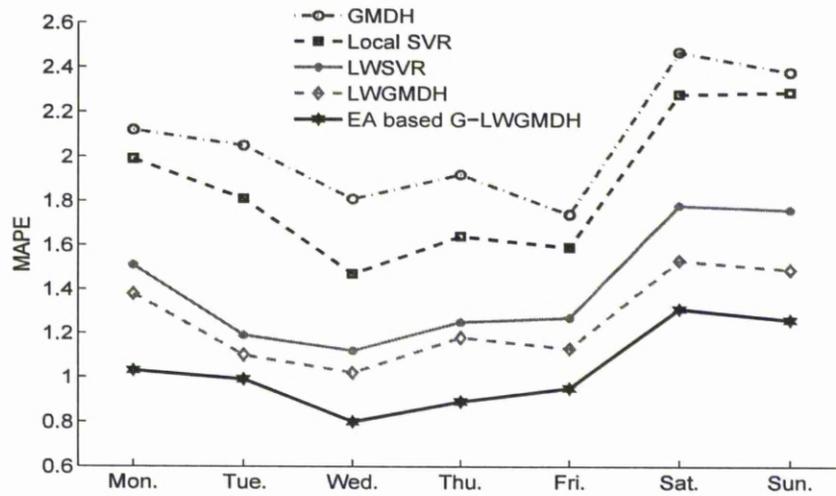


Figure 7.18: Average prediction MAPE of every day of the week during the whole testing period for the Victoria dataset.

Table 7.8: MAPE improvements of G-LWGMDH based EA over other methods using the Victoria dataset

Prediction method	Improvements			
	Apr. 2003	Jul. 2003	Oct. 2003	Dec. 2003
G-LWGMDH based EA	–	–	–	–
GMDH	54.82%	50.49%	51.98%	45.33%
Local SVR	47.95%	45.11%	45.81%	42.79%
LWSVR	27.05%	26.28%	27.07%	27.65%
LWGMDH	18.35%	16.53%	19.17%	19.08%

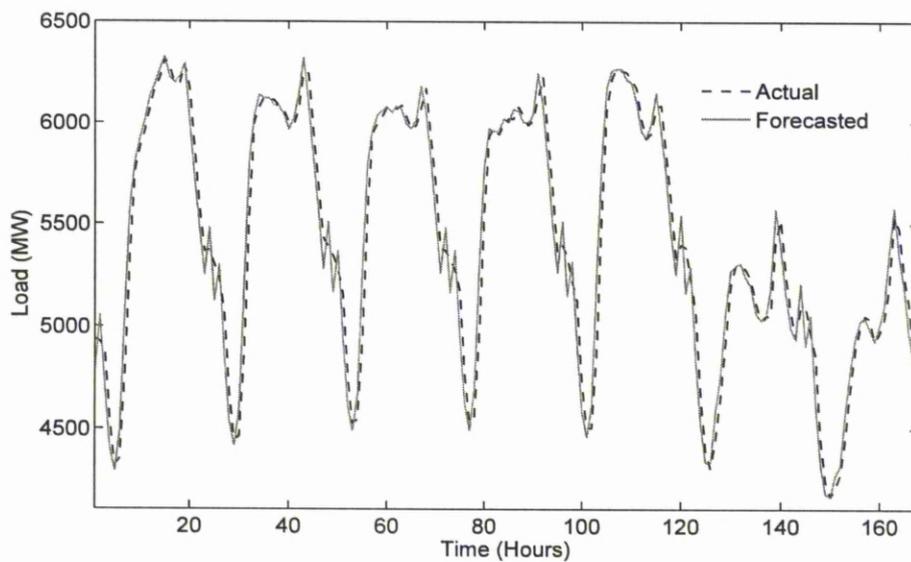


Figure 7.19: One day ahead forecasted and actual load from 7<sup>th</sup> April 2003 to 13<sup>th</sup> April 2003 using the Victoria dataset.

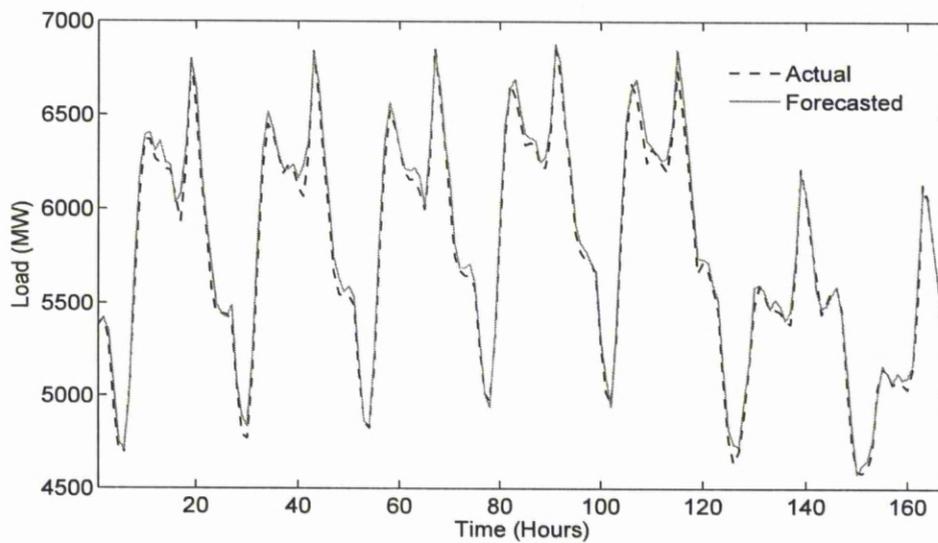


Figure 7.20: One day ahead forecasted and actual load from 7<sup>th</sup> July 2003 to 13<sup>th</sup> July 2003 using the Victoria dataset.

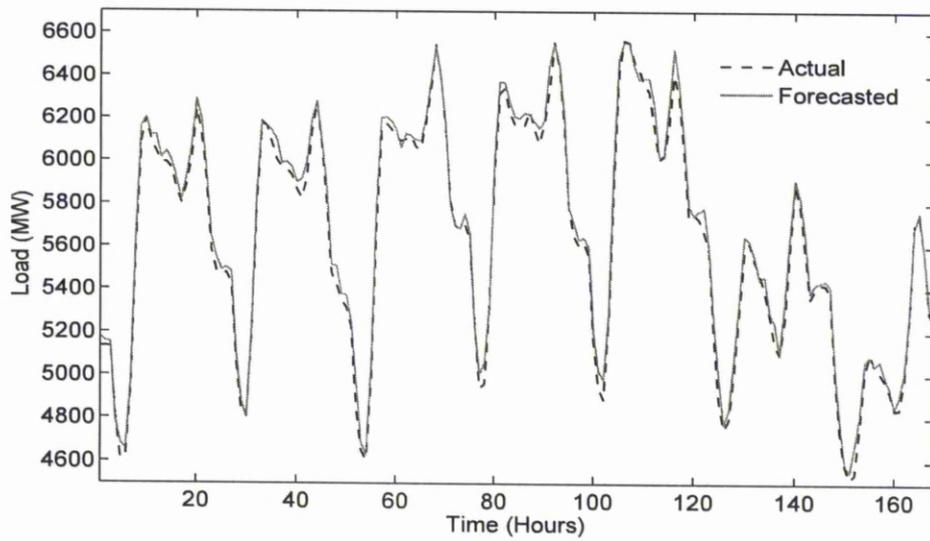


Figure 7.21: One day ahead forecasted and actual load from 6<sup>th</sup> October 2003 to 13<sup>th</sup> October 2003 using the Victoria dataset.

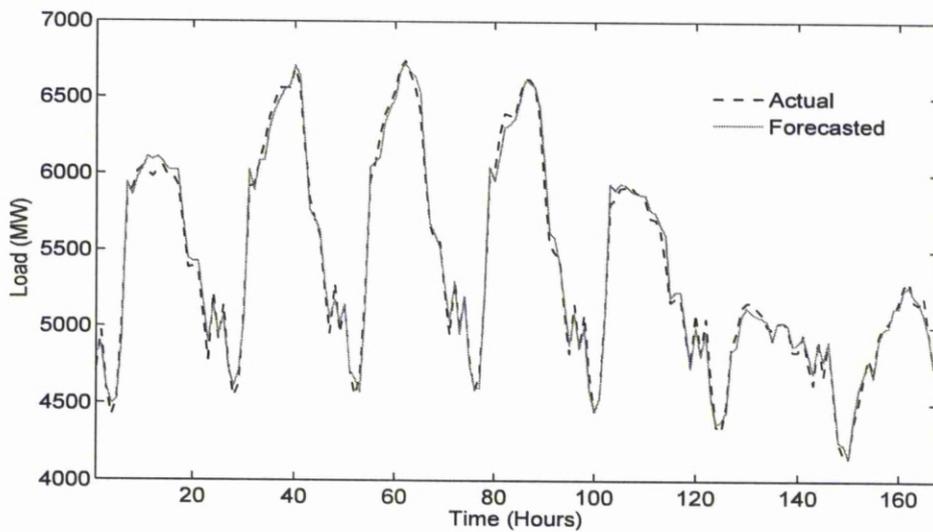


Figure 7.22: One day ahead forecasted and actual load from 1<sup>st</sup> December 2003 to 7<sup>th</sup> December 2003 using the Victoria dataset.

## 7.5 Conclusions

In this chapter, a new approach is proposed to solve the STL<sub>F</sub> problem, which is called as the G-LWGMDH based EA. The architecture of the proposed G-LWGMDH based EA method is not predetermined, but can be self-organised automatically during the design process.

The nodes of the proposed G-LWGMDH based EA method can have a different number of input variables as well as a different order of the polynomial. In addition, the connectivity configuration in the G-LWGMDH is not limited to adjacent layers unlike the conventional GMDH. Consequently, the proposed G-LWGMDH based EA method can overcome the disadvantages of the conventional GMDH due to its high level of flexibility.

The performance of the proposed G-LWGMDH based EA method is demonstrated using two different real world datasets. The results show that the proposed G-LWGMDH based EA method provides a much better forecasting performance in comparison with GMDH, local SVR, LWSVR based KPCA, LWGMDH and other published methods employing the same data.

# Chapter 8

## Conclusions

This chapter concludes the work presented in this thesis and summaries the major achievements of the presented research in the field of power system short term load forecasting and presents some suggestions for future work.

### 8.1 Summary

In order to overcome the drawbacks of global predictors, a new predictor approach is presented. The new approach can be derived by combining the three regression algorithms, such as RBF, SVR or GP with a local prediction model. For data preprocessing, the embedding dimension and the time delay constant for the power load data are computed firstly, and then the continuous power load data are used for the phase space reconstruction. In addition, after choosing the best number of the nearest neighbours using the proposed systematic method, the neighbouring points are presented by Euclidian distance. According to these neighbouring points, the local model is set up. The local predictors can overcome the drawbacks of the global predictors by involving more than one model to utilise the local information. Therefore, the accuracy of the local predictor is better than the global predictor in which only one model is engaged for all available data that contains irrelevant patterns to the current prediction point.

Three different real world datasets with two typical load types have been used to evaluate and compare the performance of local RBF, SVR and GP predictors with their global versions, as well as two benchmark methods which are seasonal

ARIMA and Holt-Winters models. The experimental cases with prediction up to day ahead have employed to consistently confirm the superiority of the local predictors over the global ones, seasonal ARIMA and Holt-Winters models. The results show that the local GP gives better performance than other methods for two datasets whereas local SVR gives the best performance for the third dataset. The local GP and local SVR methods can be recommended to the utility engineers because the obtained accuracy seems to be very good for the real-world applications.

The LWSVR method has been proposed to solve the load forecasting problem. The proposed approach combines the support vector regression and locally weighted regression and employs the weighted distance algorithm to optimise the weighting function's bandwidth. In the proposed method, each training data point is weighted according to its distance from the current point under prediction.

In the LWSVR method, the phase space is reconstructed based on multivariate time series using the embedding dimension and time delay constant for each scalar time series. In addition, the neighbouring points are selected using Euclidian distance. Then the new regularisation constant of SVR is calculated using the weighting function whose bandwidth is optimised using the weighted distance algorithm. According to these neighbouring points and the new regularisation constant, the LWSVR model is set up.

The proposed LWSVR method is tested using two different real world datasets. Our experiments have shown that the proposed LWSVR method has a better prediction accuracy than LWR, local SVR, local GP and other published methods. The effectiveness of LWSVR comes from weighting the SVR's regularisation parameter using the LWR method where each point in the neighbourhood is weighted according to its distance from the current point under prediction. The points that are close to the current point under prediction have larger weights than others. Moreover, by using the weighted distance algorithm, the drawback of using the weighting function's bandwidth as a fixed value has been overcome. This has led to improve the accuracy of LWSVR method.

Since precise STLF remains a great challenge, the LWGMDH based KPCA

has been proposed. The proposed method is derived by integrating the KPCA method with the LWGMDH which can be derived by combining the GMDH with the local regression method and WLS regression. In the proposed method, KPCA is used to extract features of the time series and obtain kernel principal components for constructing the phase space of time series. Therefore, the drawbacks of using CD method can be overcome. The coefficient parameters of GMDH's nodes are calculated using the WLS regression where each point in the neighbourhood is weighted according to its distance from the current point under prediction. By using LWGMDH to solve the load forecasting problem, the limitations of SVR can be avoided.

Two different real world datasets have been used to evaluate the performance of the proposed LWGMDH method which has been compared with GMDH (as a global method), local SVR, LWSVR based CD, LWSVR based KPCA and some published methods. In addition, electricity price is considered as one of the main characteristics of the system load in the second dataset. The numerical results have shown that the proposed LWGMDH method has achieved a better performance than GMDH (as a global method), local SVR, LWSVR based CD, LWSVR based KPCA and some published methods.

Furthermore, the G-LWGMDH based EA has been proposed and applied to STLF. In the proposed method, the EA is used to automatically select the weighting function's bandwidth and the most appropriate architecture of the G-LWGMDH network. Besides, a new encoding scheme is also proposed in which each chromosome consists of four sub-chromosomes.

The proposed G-LWGMDH based EA method results in a structurally optimised structure and comes with a higher level of flexibility in comparison to the conventional GMDH. In addition, the weighting function's bandwidth is optimised using EA. Through the consecutive process of such structural and parametric optimisation, the proposed method becomes generated in a highly dynamic fashion. This leads to avoid the drawbacks of conventional GMDH. By applying the proposed G-LWGMDH based EA method on two different real world datasets, the results show the superiority of the proposed method over the GMDH, local SVR, LWSVR, LWGMDH and other published methods.

## 8.2 Suggestions for future work

This study has presented some novel techniques for solving the STLF problem. Although this study has achieved excellent results, there are still many aspects which can be developed in order to increase the forecasting accuracy. Therefore, this section addresses several related points that deserve further investigation.

In the application of support vector regression, there is a lacking of the structural methods for confirming the selection of SVR's parameters efficiently (specially the trade-doff parameter (C)) and choosing the best kernel function. So, further research is still needed to locate the most appropriate parameters. In addition, if an inappropriate kernel has been used, the generalisation performance will suffer from overfitting. Although some work has been done on limiting kernels using prior knowledge [170], the best choice of the kernel function for a given problem still remains as a research issue.

Genetic programming is viewed by many researchers as a specialisation of genetic algorithms. The population of genetic programming individuals is constructed as tree-structured expressions. The tree structure of individuals allows genetic programming to vary its size and shape, thus, achieving a high efficiency in searching of a solution space with respect to what genetic algorithms are able to do. Therefore, a new design approach of genetic programming based G-LWGMDDH in order to improve the forecasting accuracy might be employed in the future.

The novel algorithms proposed in this thesis are methods to find the input-output relationship. So, they should not be limited to solve STLF problem. Future work might employ these algorithms to medium term load forecasting, long term load forecasting, electricity price forecasting and wind power forecasting.

Recent research on demand side management enhancements have been applied to electrical energy consumers where the load curve of these users may have some new characteristics. Future work can focus on the load forecasting of the demand side management users.

STLF is the prediction of the system load over a defined time interval that

---

ranges from one hour to few days for power system operation purposes. Due to the random nature of the load and its dependency on numerous factors, forecasting is never 100% accurate. Inaccuracies in load forecasting will cause an increase in the cost of operating the power system and on the risk measured in terms of expectation of energy not served due to unit outages. The economic impact analysis of STLF is not included in this study. Therefore, the economical impact of the STLF errors on the daily power system operation can be considered in the future.

# Bibliography

- [1] E. Kyriakides and M. Polycarpou. Short term electric load forecasting: A tutorial. *Studies in Computational Intelligence*, 35:391–418, 2007.
- [2] G. Gross and F. D. Galiana. Short-term load forecasting. *Proceedings of the IEEE*, 75(12):1558–1573, 1987.
- [3] A. G. Bakirtzis, V. Petridis, S. J. Kiartzis, M. C. Alexiadis, and A. H. Maissis. A neural network short-term load forecasting model for the Greek power system. *IEEE Transactions on Neural Networks*, 11(2):858–863, May 1996.
- [4] D. K. Ranaweera, G. G. Karady, and R. G. Farmer. Economic impact analysis of load forecasting. *IEEE Transactions on Power Systems*, 12(3):1388–1392, August 1997.
- [5] H. S. Hippert, C. E. Pedreira, and R. C. Souza. Neural networks for short-term load forecasting: A review and evaluation. *IEEE Transactions on Power Systems*, 16(1):44–55, February 2001.
- [6] M. Mohandes. Support vector machines for short-term electrical load forecasting. *International Journal of Energy Research*, 26:335–345, 2002.
- [7] C. M. Huang, C. J. Huang, and M. L. Wang. A particle swarm optimization to identifying the ARMAX model for short-term load forecasting. *IEEE Transactions on Power Systems*, 20(2):1126–1133, May 2005.
- [8] S. K. Oh. and W. Pedrycz. The design of self-organizing polynomial neural networks. *Information Sciences*, 141:237–258, 2002.

- 
- [9] A. G. Ivakhnenko. Polynomial theory of complex systems. *IEEE Transactions of System, Man and Cybernetics*, SMC-1:364–378, 1971.
- [10] D. Srinivasan. Energy demand prediction using GMDH networks. *Neurocomputing*, 72:625–629, 2008.
- [11] L. Caoa, K. Chuab, W. Chongc, H. Leea, and Q. Gud. A comparison of PCA, KPCA and ICA for dimensionality reduction in support vector machine. *Neurocomputing*, 55:321–336, 2003.
- [12] T. W. S. Chow and C. T. Leung. Neural networks based short-term load forecasting using weather compensation. *IEEE Transactions on Power Systems*, 11(4):1736–1742, May 1996.
- [13] S. Fan and L. Chen. Short-term load forecasting based on an adaptive hybrid method. *IEEE Transactions On Power Systems*, 21(1):392–401, February 2006.
- [14] H. Chen, C. A. Canizares, and A. Singh. ANN-based short-term load forecasting in electricity markets. In *Proceedings of the IEEE Power Engineering Society Transmission and Distribution Conf.*, pages 496–499, Columbus, OH, USA, 2001.
- [15] P. Mandal, T. Senjyu, and T. Funabashi. Neural networks approach to forecast several hour ahead electricity prices and loads in deregulated market. *Energy Conversion and Management*, 47:2128–2142, 2006.
- [16] K. H. Kim, J. K. Park, K. J. Hwang, and S. H. Kim. Implementation of hybrid short-term load forecasting system using artificial neural networks and fuzzy expert systems. *IEEE Transactions on Power Systems*, 10(3):1534–1539, August 1995.
- [17] S. Rahman and O. Hazim. Load forecasting for multiple sites: Development of an expert system-based technique. *Electric Power Systems Research*, 39:161–169, 1996.

- 
- [18] A. D. Papalekopoulos and T. C. Hesterberg. A regression-based approach to short-term system load forecasting. *IEEE Transactions on Power Systems*, 5(4):1535–1547, November 1990.
- [19] N. Amjady. Short-term hourly load forecasting using time-series modeling with peak load estimation capability. *IEEE Transactions on Power Systems*, 16(3):498–505, November 2001.
- [20] S. A. Villalba and C. A. Bel. Hybrid demand model for load estimation and short-term load forecasting in distribution electrical systems. *IEEE Transactions on Power Delivery*, 15(2):764–769, April 2000.
- [21] I. Moghram and S. Rahman. Analysis and evaluation of five short-term load forecasting techniques. *IEEE Transactions on Power Systems*, 4(4):1484–1491, November 1989.
- [22] T. Senjyu, H. Takara, K. Uezato, and T. Funabashi. One-hour-ahead load forecasting using neural network. *IEEE Transactions on Power Systems*, 17(1):113–118, February 2002.
- [23] K. L. Ho, Y. Y. Hsu, F. F. Chen, T. E. Lee, C. C. Liang, T. S. Lai, and K. K. Chen. Short term load forecasting of taiwan power system using a knowledge-based expert system. *IEEE Transactions on Power Systems*, 5(4):1214–1221, November 1990.
- [24] H. Mori and T. Itagaki. A fuzzy inference neural network based method for short-term load forecasting. In *The IEEE Int. Joint Conf. on Neural Network*, pages 2403–2406, July 25–29, 2004.
- [25] D. K. Ranaweera, N. F. Hubele, and A. D. Papalexopoulos. Application of radial basis function neural network model for short-term load forecasting. *IEEE Proc. Generation, Transmission and Distribution*, 142:45–50, January 1995.
- [26] B. J. Chen, M. W. Chang, and C. J. Lin. Load forecasting using support vector machines: A study on EUNITE competition 2001. *IEEE Transactions on Power Systems*, 19(4):1821–1830, November 2004.
-

- [27] H. Mori and M. Ohmi. Probabilistic short-term load forecasting with Gaussian process. In *The 13<sup>th</sup> Int. Conf. on Intelligent Systems, Application to Power Systems*, pages 452–457, 2005.
- [28] E. A. Feinberg and D. Genethliou. Load forecasting. *Applied Mathematics for Restructured Electric Power Systems (Springer-Verlag, New York, 2005)*, chapter 12:269–285, 2005.
- [29] T. Haida and S. Muto. Regression based peak load forecasting using a transformation technique. *IEEE Transactions on Power Systems*, 9(4):1788–1794, November 1994.
- [30] R. Ramanathan, R. Engle, C. W. J. Granger, F. V. Araghi, , and C. Brace. Short-run forecasts of electricity loads and peaks. *Int. Journal of Forecasting*, 13:161–174, 1997.
- [31] M. E. El-Hawary and G. A. N. Mbamalu. Short-term power system load forecasting using the iteratively reweighted least squares algorithm. *Electric Power Systems Research*, 19:11–22, 1990.
- [32] R. P. Broadwater, A. Sargent, A. Yarali, H. E. Shaalan, and J. Nazarko. Estimating substation peaks from load research data. *IEEE Transactions on Power Delivery*, 12(1):451–456, January 1997.
- [33] A. D. Papalexopoulos and T. C. Hesterberg. A regression based approach to short term load forecasting. *IEEE Transactions on Power Systems*, 5(4):1535–1550, November 1990.
- [34] S. Ruzic, A. Vuckovic, and N. Nikolic. Weather sensitive method for short term load forecasting in electric power utility of serbia. *IEEE Transactions on Power Systems*, 18(4):1581–1586, November 2003.
- [35] G. E. P. Box and G. M. Jenkins. *Time series analysis, forecasting and control*. Holden Day, San Fransisco, 1970.
- [36] H. T. Yang, C. M. Huang, and C. L. Huang. Identification of ARMAX model for short-term load forecasting: An evolutionary programming ap-

- proach. *IEEE Transactions on Power Systems*, 11(1):403–408, February 1996.
- [37] M. Espinoza, C. Joye, R. Belmans, and B. De Moor. Short-term load forecasting, profile identification, and customer segmentation: A methodology based on periodic time series. *IEEE Transactions on Power Systems*, 20(3):1622–1630, August 2005.
- [38] J. W. Taylor and P. E. McSharry. Short-term load forecasting methods: An evaluation based on European data. *IEEE Transactions on Power Systems*, 22(4):2213–2219, November 2007.
- [39] S. C. Bhattacharyyan and L. T. Thanh. Short-term electric load forecasting using an artificial neural network: case of northern vietnam. *International Journal of Energy Research*, 28:463–472, 2004.
- [40] F. Cavallaro. Electric load analysis using an artificial neural network. *International Journal of Energy Research*, 29:377–392, 2005.
- [41] P. J. Santos, A. Gomes Martins, and A.J. Pires. On the use of reactive power as an endogenous variable in short-term load forecasting. *International Journal of Energy Research*, 27:513–529, 2003.
- [42] M. Ramezani, H. Falaghi, M. Haghifam, and G. Shahryari. Short-term electric load forecasting using neural networks. In *EUROCON 2005*, pages 1525–1528, Serbia Montenegro, Belgrade, November 22–24, 2005.
- [43] S. Haykin. *Neural networks, A comprehensive foundation*. Printic-Hall, Inc., 1999.
- [44] A. Khotanzad, R. A. Rohani, T. L. Lu, A. Abaye, M. Davis, and D. J. Maratukulam. ANNSTLF- a neural-network-based electric load forecasting system. *IEEE Transactions on Neural Networks*, 8(4):835–846, July 1997.
- [45] A. Khotanzad, R. A. Rohani, and D. Maratukulam. ANNSTLF- artificial neural network short-term load forecaster- generation three. *IEEE Transactions on Neural Networks*, 13(4):1413–1422, November 1998.

- [46] R. Afkhami and F. Yazdi. Application of neural networks for short-term load forecasting. In *Power India Conference*, pages 349–353, India, April 10–12, 2006.
- [47] S. Ling, F. Leung, H. Lam, Y. Lee, and P. Tam. A novel genetic-algorithm-based neural network for short-term load forecasting. *IEEE Transactions on Industrial Electronics*, 50(4):793–799, August 2003.
- [48] G. Liao and T. Tsao. Application of a fuzzy neural network combined with a chaos genetic algorithm and simulated annealing to short-term load forecasting. *IEEE Transactions on Evolutionary Computation*, 10(3):330–340, June 2006.
- [49] K. Methaprayoon. Neural network based short-term load forecasting for unit commitment scheduling. Master’s thesis, The university of Texas at Arlington, August 2003.
- [50] S. Rahman and O. Hazim. A generalized knowledge-based short term load forecasting technique. *IEEE Transactions on Power Systems*, 8(2):508–514, 1993.
- [51] K. J. Hwang and G. W. Kim. A short-term load forecasting expert system. In *The Fifth Russian-Korean Int. Symposium on Science and Technology (KORUS '01)*, pages 112–116, 2001.
- [52] N. J. Balu, R. Adapa, G. Cauley, M. Lauby, and D. J. Maratukulam. Review of expert systems in bulk power system planning and operation. *Proceedings of the IEEE*, 80(5):727–731, May 1992.
- [53] Y. Chen, P. B. Luh, and S. J. Rourke. Short-term load forecasting: Similar day-based wavelet neural networks. In *Proceedings of the 7<sup>th</sup> World Congress on Intelligent Control and Automation (WCICA)*, pages 3353–3358, Chongqing, China, June 25–27, 2008.
- [54] K. N. Pantazopoulos, L. H. Tsoukalas, N. G. Bourbakis, M. J. Brun, and E. N. Houstis. Financial prediction and trading strategies using neurofuzzy

- approaches. *IEEE Transactions on System, Man and Cybernetics. Part B, Cybernetics*, 28(4):520–531, August 1998.
- [55] B. K. Chauhan, A. Sharma, and M. Hanmandlu. Neuro-fuzzy approach based short term electric load forecasting. In *IEEE/PES Transmission and Distribution Conf. and Exhibition: Asia and Pacific*, pages 1–5, Dalian, China, 2005.
- [56] A. Khotanzad, E. Zhou, and H. Elragal. A neuron-fuzzy approach to short-term load forecasting in a price-sensitive environment. *IEEE Transactions on Power Systems*, 17(4):1273–1282, September 2002.
- [57] S. H. Ling, F. H. F. Leung, H. K. Lam, and P. K. S. Tam. Short-term electric load forecasting based on a neural fuzzy network. *IEEE Transactions on Industrial Electronics*, 50(6):1305–1316, December 2003.
- [58] H. Mori and N. Kosemura. Optimal regression tree based rule discovery for short-term load forecasting. In *Proceedings of the IEEE Power Engineering Society Transmission and Distribution Conf.*, pages 421–426, 2001.
- [59] J. Q. Li, C. L. Niu, J. Z. Liu, and J. J. Gu. The application of data mining in electric short-term load forecasting. In *5<sup>th</sup> Int. Conf. on Fuzzy Systems and Knowledge Discovery (FSKD 2008)*, pages 519–522, 2008.
- [60] X. Li, C. Sun, and D. Gong. Application of support vector machine and similar day method for load forecasting. *Lecture Notes in Computer Science*, 3611:602–609, 2005.
- [61] C. C. Cai and W. Min. Support vector machines with similar day’s training sample application in short-term load forecasting. In *3<sup>rd</sup> Int. Conf. on Deregulation and Restructuring and Power Technologies, (DRPT 2008)*, pages 1221–1225, Nanjing, China, April 6–9, 2008.
- [62] A. L. Edwards. *An introduction to linear regression and correlation*. Freeman, New York, 1984.
- [63] M. N. Gibbs. *Bayesian Gaussian processes for regression and classification*. PhD thesis, University of Cambridge, 1997.

- [64] A. N. Tikhonov and V. Y. Arsenin. *Solution of ill-posed problems*. Winston and Sons, Washington, DC, 1977.
- [65] N. Christiani and J. S. Taylor. *An introduction to support vector machines and other kernel-based learning methods*. Cambridge University Press, Cambridge, 2000.
- [66] J. Fox. *Nonparametric simple regression: Smoothing scatterplots*. Sage, Thousand Oaks, CA, 2000.
- [67] D. Asber, S. Lefebvre, M. Saad, and C. Desbiens. Non-parametric short-term load forecasting. *Int. Journal of Electrical Power and Energy Systems*, 29(8):630–635, 2007.
- [68] R. Zivanovic. Local regression-based short-term load forecasting. *Journal of Intelligent and Robotic Systems*, 31(1-3):115–127, 2001.
- [69] H. Reinsch. Smoothing by spline functions. *Numerische Mathematik*, 10:177–183, 1967.
- [70] C. Stone. Consistent nonparametric regression. *Annals of Statistics*, 5(4):595–645, 1977.
- [71] A. T. Lora, J. M. R. Santos, J. C. Riquelme, A. G. Exposito, and J. L. M. Ramos. Time-series prediction: Application to the short-term electric energy demand. *Lecture Notes in Artificial Intelligence*, 3040:577–586, 2004.
- [72] K. W. Lau. *Nonlinear prediction and classification*. PhD thesis, The University of Liverpool, Department of Electrical Engineering and Electronics, 2002.
- [73] J. H. Friedman. An overview of predictive learning and function approximation. In *Cherkassky V., Friedman J. and Wechsler H., editors, From Statistics to Neural Networks, Proc. NATO/ASI Workshop*, pages 1–55, 1994.

- [74] F. M. A. Acosta. Radial basis function and related models: An overview. *Signal Processing*, 45(1):37–58, July 1995.
- [75] I. T. Nabney. *Netlab algorithms for pattern recognition*. Springer, 2002.
- [76] M. Kraaijveld and R. Duin. Generalization capabilities of minimal kernel-based networks. In *Proceedings of the International Joint Conference on Neural Networks*, pages 843–848, Seattle, WA, USA, July 8–14, 1991.
- [77] C. Bishop. *Neural networks for pattern recognition*. Clarendon Press, Oxford, 1995.
- [78] J.E. Moody and C.J. Darken. Fast learning in networks of locally tuned processing units. *Neural Computation*, 1(2):281–294, 1989.
- [79] R. Zemouri, D. Racocceanu, and N. Zerhouni. Recurrent radial basis function network for time-series prediction. *Engineering Applications of Artificial Intelligence*, 16(5-6):453–463, 2003.
- [80] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistic Society, Series B (Methodological)*, 39(1):1–38, 1977.
- [81] C. Harphama and C.W. Dawsonb. The effect of different basis functions on a radial basis function network for time series prediction: A comparative study. *Neurocomputing*, 69:2161–2170, 2006.
- [82] V. N. Vapnik. *The natural of statistical learning theory*. Springer, New York, 1995.
- [83] V. N. Vapnik. *Statistical learning theory*. John Wiley and Sons, 1998.
- [84] B. Scholkopf, C. Burges, and V. Vapnik. Extracting support data for a given task. In *Proceedings of the First Int. Conf. on Knowledge Discovery and Data Mining, Menlo Park*, pages 252–257. AAAI Press, 1995.
- [85] B. Ribeiro. Support vector machines for quality monitoring in a plastic injection molding process. *IEEE Transactions on System, Man and Cybernetics. Part C, Application and Review*, 35(3):401–410, August 2005.

- 
- [86] J. I. Park, S. H. Baek, M. K. Jeong, and S. J. Bae. Dual features functional support vector machines for fault detection of rechargeable batteries. *IEEE Transactions on System, Man and Cybernetics. Part C, Application and Review*, 39(4):480–485, July 2009.
- [87] Y. Tang, Y. Q. Zhang, and N. V. Chawla. Svms modeling for highly imbalanced classification. *IEEE Transactions on System, Man and Cybernetics. Part B, Cybernetics*, 39(1):281–288, January 2009.
- [88] A. J. Smola and B. Scholkopf. A tutorial on support vector regression. *NeuroCOLT Technical Report NC-TR-98-030, Royal Holloway College, University of London*, 1998.
- [89] K. W. Lau and Q. H. Wu. Local prediction of non-linear time series using support vector regression. *Pattern Recognition*, 41:1539–1547, May 2008.
- [90] C. C. Chuang and J. T. Jeng. Selection of initial structures with support vector regression for fuzzy neural networks. *Int. Journal Fuzzy Systems*, 6(2):63–70, 2004.
- [91] A. Shilton, D. T. H. Lai, and M. Palaniswami. A division algebraic framework for multidimensional support vector regression. *IEEE Transactions on System, Man and Cybernetics. Part B, Cybernetics*, Article in press.
- [92] M. Espinoza, J. A. K. Suykens, R. Belmans, and B. De Moor. Electric load forecasting using kernel based modeling for nonlinear system identification. *IEEE Control Systems Magazine, Special Issue on Applications of Systems Identification*, 27(5):43–57, October 2007.
- [93] K. Y. Chen and C. H. Wang. Support vector regression with genetic algorithms in forecasting tourism demand. *Tourism Management*, 28:215–226, 2007.
- [94] W. Karush. Minima of functions of several variables with inequalities as side constraints. Master’s thesis, Department of Mathematics, University of Chicago, 1939.
-

- 
- [95] H. W. Kuhn and A. W. Tucker. Nonlinear programming. In *In Proc. of 2<sup>nd</sup> Berkeley Symposium on Mathematical Statistics and Probabilistics*, pages 481–492, University of California Press, 1951.
- [96] F. Girosi. An equivalence between sparse approximation and support vector machines. *Neural Comutation*, 10(6):1455–1480, August 1998.
- [97] S. R. Gunn. Support vector machines for classification and regression. *Technical Report Image Speech and Intelligent Systems Research Group, University of Southampton*, 1997.
- [98] N. Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68(3):337–404, May 1950.
- [99] B. Scholkopf and A. J. Smola. *Learning with kernels- support vector machines, regularization, optimization and beyond*. The MIT Press, Cambridge, 2002.
- [100] A. Karatzoglou, D. Meyer, and K. Hornik. Support vector machines in R. *Journal of Statistical Software*, 15(9):1–28, April 2006.
- [101] R. Fletcher. *Practical methods of optimization*. John Wiley and Sons, 1987.
- [102] T. Joachims. Text categorization with support vector machines: Learning with many relevant features. *Lecture Notes in Computer Science*, 1398:137–142, 1998.
- [103] J. C. Platt. Sequential minimal optimization: A fast algorithm for training support vector machines. *Microsoft Research Technical Report MSR-TR-98-14*, 1998.
- [104] E. Osuna, R. Freund, and F. Girosi. An improved training algorithm for support vector machines. In *Proceedings of the IEEE Workshop on Neural Networks for Signal Processing*, pages 276–285, 1997.
-

- [105] R. M. Neal. Monte carlo implementation of Gaussian process models for Bayesian regression and classification. *Technical Report No. 9702, Department of Statistics, University of Toronto*, 1997.
- [106] S. B. Belhouari and A. Bermak. Gaussian process for nonstationary time series prediction. *Computational Statistics and Data Analysis*, 47(4):705–712, 2004.
- [107] W. Chu and Z. Ghahramani. Gaussian processes for ordinal regression. *Machine Learning Research*, 6:1019–1041, July 2005.
- [108] C. K. I. Williams. Prediction with Gaussian processes: From linear regression to linear prediction and beyond. In *M. I. Jordan, editor, Learning in Graphical Models, MIT Press, Cambridge*, pages 599–621, 1999.
- [109] R. M. Neal. *Bayesian learning for neural networks*. PhD thesis, Department of Computer Science, University of Toronto, 1994.
- [110] S. Barnett. *Matrix methods for engineers and scientists*. McGraw-Hill, London, 1979.
- [111] C. E. Rasmussen. *Evaluation of Gaussian processes and other methods for non-linear regression*. PhD thesis, Department of Computer Science, University of Toronto, 1996.
- [112] P. Abrahamsen. A review of Gaussian random fields and correlation functions. *Technical Report No. 917, Norwegian Computing Center, Oslo, Norway*, 1997.
- [113] F. Takens. Detecting strange attractors in turbulence. *Lecture Notes in Mathematics (Springer Berlin)*, 898:366–381, 1981.
- [114] T. Sauer, J. A. Yorke, and M. Casdagli. Embedology. *Journal of Statistical Physics*, 65:579–616, November 1991.
- [115] P. Grassberger and I. Procaccia. Estimation of the kolmogorov entropy from a chaotic signal. *Physics Review A*, 28:2591–2593, 1983.

- 
- [116] D. S. Broomhead and G. P. King. Extracting qualitative dynamics from experimental data. *Physica D*, 20:217–236, 1986.
- [117] H. D. I. Abarbanel. *Analysis of observed chaotic data*. Springer-Verlag, New York, 1996.
- [118] J. P. Eckmann and D. Ruelle. Ergodic theory of chaos and strange attractors. *Reviews of Modern Physics*, 57(3):617–656, July 1985.
- [119] A. M. Albano, J. Muench, A. I. Mees, and P. E. Rapp. Singular-value decomposition and the Grassberger-Procaccia algorithm. *Physical Review A*, 38(6):3017–3026, September 1988.
- [120] A. M. Fraser and H. L. Swinny. Independent coordinates for strange attractors from mutual information. *Physical Review*, 33(2):1134–1140, February 1986.
- [121] W. Liebert and H. G. Schuster. Proper choice of the time delay for the analysis of chaotic time series. *Physics Letters A*, 142(2-3):107–111, December 1989.
- [122] J. D. Farmer and J. J. Sidorowich. Predicting chaotic time series. *Physical Review Letters*, 59:845–848, 1987.
- [123] M. Casdagli. Non-linear prediction of chaotic time series. *Physica D*, 35(3):335–356, 1989.
- [124] J. McNames, J. A. K. Suykens, and J. Vandewalle. Winning entry of the K.U. Leuven time series prediction competition. *Int. Journal of Bifurcation and chaos*, 9(8):1485–1500, August 1999.
- [125] A. T. Lora, J. C. Riquelme, and J. L. M. Ramos. Influence of kNN-based load forecasting errors on optimal energy production. *Lecture Notes in Computer Science. (Springer Berlin)*, 2902:189–203, 2003.
- [126] A. Sorjamaa, N. Reyhani, and A. Lendasse. Input and structure selection for k-NN approximator. *Lecture Notes in Computer Science. (Springer Berlin)*, 3512:985–992, 2005.
-

- [127] National grid (NatG). Available at: <http://www.nationalgrid.com>.
- [128] Transmission system operator (RTE). Available at: <http://www.rte-france.com>.
- [129] Hellenic transmission system operator (HTSO). Available at: <http://www.desmie.gr/home>.
- [130] W. C. Hong and P. F. Pai. Predicting engine reliability by support vector machines. *Int. Journal of Advanced Manufacturing Technology*, 28:154–161, February 2006.
- [131] User’s Guide and Reference Manual. *Minitab Statistical Software*, 1995.
- [132] F. E. H. Tay and L. J. Cao. Modified support vector machines in financial time series forecasting. *Neurocomputing*, 48:847–861, 2002.
- [133] D. E. Lee, J. H. Song, S. O. Song, and E. S. Yoon. Weighted support vector machine for quality estimation in the polymerization process. *Industrial and Engineering Chemistry Research*, 44(7):2101–2105, 2005.
- [134] G. S. Hu, Y. Z. Zhang, and F. F. Zhu. Short-term load forecasting based on fuzzy C-mean clustering and weighted support vector machines. In *Proceeding of the Third Int. Conf. on Natural Computation, (ICNC 2007)*, pages 654–659, 2007.
- [135] L. Cao, A. Mees, and K. Judd. Dynamics from multivariate time series. *Physica D*, 121:75–88, October 1998.
- [136] W. S. Cleveland and S. J. Devlin. Locally weighted regression: An approach to regression analysis by local fitting. *Journal of the American Statistical Association*, 83(403):596–610, September 1988.
- [137] C. C. Atkeson, A. W. Moore, and S. Schaal. Locally weighted learning. *Artificial Intelligence Review (Special Issue on Lazy Learning)*, 11:11–73, April 1997.

- [138] M. P. Wand and W. R. Schucany. Gaussian-based kernels for curve estimation and window width selection. *Canadian Journal of Statistics*, 18:197–204, 1990.
- [139] H. Wang, C. Cao, and H. Leung. An improved locally weighted regression for a converter re-vanadium prediction modeling. In *Proceedings of the 6<sup>th</sup> World Congress on Intelligent Control and Automation*, pages 1515–1519, Dalian, China, June 21–23, 2006.
- [140] World wide competition within the eunite network. Available at:<http://neuron.tuke.sk/competition>.
- [141] Available: [http : //www.ee.washington.edu/class/555/el - sharkawi/index\\_files/Page3404.htm](http://www.ee.washington.edu/class/555/el_sharkawi/index_files/Page3404.htm).
- [142] S. R. Abbas and M. Arif. Electric load forecasting using support vector machines optimized by genetic algorithm. In *IEEE Multitopic Conf. (INMIC '06)*, pages 395–399, December 23–24, 2006.
- [143] X. Tao, H. Renmu, W. Peng, and X. Dongjie. Input dimension reduction for load forecasting based on support vector machines. In *The IEEE Int. Conf. on Electric Utility Deregulation, Restructuring and Power Tech. (DRPT2004)*, pages 510–514, Hong Kong, 2004.
- [144] V. H. Ferreira and A. P. Alves da Silva. Toward estimation autonomous neural network-based electric load forecasters. *IEEE Transactions on Power Systems*, 22(4):1554–1562, November 2007.
- [145] D. Tao and X. Hongfei. Chaotic time series prediction based on radial basis function network. In *Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing*, pages 595–599, 2007.
- [146] F. Chen and C. Han. Time series forecasting based on wavelet KPCA and support vector machine. In *IEEE Int. Conf. on Automation and Logistics*, pages 1487–1491, 2007.

- [147] C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2:121–167, 1998.
- [148] A. Vicino, R. Tempo, R. Genesio, and M. Milanese. Optimal error and GMDH predictors: A comparison with some statistical techniques. *Int. Journal of Forecasting*, 3:313–328, 1987.
- [149] R. E. Abdel-Aal, M. A. Elhadidy, and S. M. Shaahid. Modeling and forecasting the mean hourly wind speed time series using GMDH-based abductive networks. *Renewable Energy*, 34(7):1686–1699, 2009.
- [150] S. J. Farlow. *Self-organizing method in modeling: GMDH type algorithm*. Marcel Dekker Inc., 1984.
- [151] J. Xi and M. Han. Reduction of the multivariate input dimension using principal component analysis. *Lecture Notes in Computer Science*, 4099:366–381, 2006.
- [152] G. Onwubolu. Hybrid computational intelligence and GMDH systems. *Studies in Computational Intelligence*, 211:1–26, 2009.
- [153] H. Iba N. Y. Nikolaev. Polynomial harmonic GMDH learning networks for time series modeling. *Neural Networks*, 16:1527–1540, 2003.
- [154] A. G. Ivakhnenko and G. A. Ivakhnenko. The review of problems solved by algorithms of the group method of data handling (GMDH). *Pattern Recognition and Image Analysis*, 5:527–535, 1995.
- [155] Australian energy market operator (AEMO) website. Available at: <http://www.aemo.com.au/>.
- [156] H. Park, B. Park, H. Kim, and S. Oh. Self-organizing polynomial neural networks based on genetically optimized multi-layer perceptron architecture. *Int. Journal of Control, Automation and Systems*, 2(4):423–434, 2004.
- [157] N. Amanifard, N. Zadeh, M. Borji, A. Khalkhali, and A. Habibdoust. Modelling and pareto optimization of heat transfer and flow coefficients in

- microchannels using GMDH type neural networks and genetic algorithms. *Energy Conversion And Management*, 49(2):311–325, 2008.
- [158] A. Sharma and G. Onwubolu. Hybrid particle swarm optimization and GMDH system. *Studies in Computational Intelligence*, 211:193–231, 2009.
- [159] I. Hitoshi. Hybrid genetic programming and GMDH system: STROGANOFF. *Studies in Computational Intelligence*, 211:27–98, 2009.
- [160] H. Iba, H. de Garis, and T. Sato. Genetic programming using a minimum description length principle. *Advances in Genetic Programming (MIT Press, Cambridge)*, pages 265–284, 1994.
- [161] N. Zadeh, A. Darvizeh, A. Jamali, and A. Moeini. Evolutionary design of generalized polynomial neural networks for modelling and prediction of explosive forming process. *Journal of Materials Processing Tech.*, 164-165:1561–1571, 2005.
- [162] A. P. Engelbrecht. *Computational intelligence: An introduction*. Jhon Willey and Sons, Ltd., 2007.
- [163] D. B. Fogel. An introduction to simulated evolutionary optimization. *IEEE Transactions on Neural Networks*, 5(1):820–834, January 1994.
- [164] Z. Michalewicz. *Genetic algorithms + data structures = Evolving programs*. Springer-Verlag, Berlin Heidelberg, 1996.
- [165] J. E. Baker. Reducing bias and inefficiency in the selection algorithm. In *Proceedings of Int. Conf. Genetic Algorithms*, pages 14–21, 1987.
- [166] J. H. Holland. *Adaptation in natural and artificial systems*. MIT Press, Cambridge, MA, 1992.
- [167] D. E. Goldberg. *Genetic algorithm in search, optimization and machine learning*. Addison Wesley, 1989.
- [168] Y. Shi, R. Eberhart, and Y. Chen. Implementation of evolutionary fuzzy system. *IEEE Transactions on Fuzzy Systems*, 7(2):109–119, April 1999.

- 
- [169] New york independent system operator (NYISO). Available at: <http://www.nyiso.com>.
- [170] B. Scholkopf, P. Y. Simard, A. J. Smola, and V. N. Vapnik. Prior knowledge in support vector kernels. In *Proceedings of the 1997 Conf. on Advances in Neural Information Processing Systems*, volume 10, pages 640–646, Cambridge, MA, USA, 1998. MIT Press.