OPTIMAL AND ADAPTIVE RADIAL BASIS FUNCTION NEURAL NETWORKS

AKBAR SHAHSAVAND

A THESIS SUBMITTED TO THE UNIVERSITY OF SURREY FOR THE AWARD OF THE DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF CHEMICAL AND PROCESS ENGINEERING SCHOOL OF ENGINEERING IN THE ENVIRONMENT UNIVERSITY OF SURREY SURREY, GUILDFORD GU2 7XH, UK

August 2000
ACKNOWLEDGMENT

I would like to express my best gratitude to my first supervisor Dr. F. A. Farhadpour. I am deeply indebted to him, for his guidance during my visits to the University of Surrey.

I also wish to thank my second supervisor Dr. M. Jami-al-ahmadi for his valuable cooperation and Dr Andy Tate for his technical support.

Thanks are also extended to Professor Hans Muller Steinhagen, ex-head of school of Engineering in the Environment for his support during the course of this study and the Ministry of Culture and Higher Education (MCHE) of the Islamic Republic of Iran for supporting this project.

And last but not least, I would like to thank my wife Mrs Marzieh Daneshamouz for her patience and priceless encouragement during this period.
NOMENCLATURE

Latin’s:

\( a \) General Model parameters

\( A \) Reactant (Section 4.6)

\( A(F) \) Standard error term functional

\( AIC \) Akaike’s information criterion

\( B(F) \) Regularisation (constraint) term functional

\( C \) Norm weighting matrix (Chapters 3)

\( C \) Diagonal matrix of singular values for GSVD (Chapter 2)

\( C \) Centres of the data clusters (Chapter 4)

\( CV(\lambda) \) Cross Validation criterion

\( C_{A_i} \) Reactant inlet concentration (Section 4.6)

\( C_{A_o} \) Reactant outlet concentration (Section 4.6)

\( C_{R_i} \) Product inlet concentration (Section 4.6)

\( C_{R_o} \) Product outlet concentration (Section 4.6)

\( D \) Diagonal matrix

\( D_k \) \( N \times N \) diagonal matrix with all diagonal elements equal to one except the \( k^{th} \) element which is zero

\( df(\lambda) \) Effective degrees of freedom

\( d \) Given vector quantity (Chapter 2)

\( d \) Maximum distance between centres (Chapter 4)

\( e_k \) The \( k^{th} \) unit vector

\( e \) Energy (Equation 2.116)
\( F(e) \)  Energy distribution in adsorption example (chapter 2)

\( f(\cdot) \)  Functional form of the model

\( F_\lambda \)  Solution of the regularisation network

\( \bar{F}_\lambda \)  Approximate solution of the generalisation network

\( g_k \)  \( k^{th} \) Column of the Green's matrix

\( G(x_i, x_j) \)  \( ij^{th} \) element of the Green's matrix

\( \hat{G} \)  Complete Green's matrix with global features

\( G_0(l_i, l_j) \)  \( ij^{th} \) element of regularisation matrix

\( \hat{G}_0 \)  Regularisation matrix with global features

\( h \)  Arbitrary function for Frechet derivative

\( H(\lambda) \)  Smoother matrix

\( I \)  Identity matrix

\( k_0 \)  Arhenius constant (Equation 2.116)

\( k_f \)  Forward path reaction rate constant

\( k_r \)  Reverse path reaction rate constant

\( L \)  Order of algebraic polynomial bases

\( M \)  Number of neurones

\( m_j \)  Number of active sites with energy \( e_j \) (Chapter 2)

\( n \)  Order of partial differential operator

\( n \)  Iteration counter of the gradient decent optimisation method

\( N \)  Number of training exemplars (data points)

\( p \)  Dimension of input variable
\( p_i \)  Probability density of the \( i^{th} \) data point

\( p_i \)  Bulk pressure (Equation 2.116)

\( P \)  Probability density of the entire data set

\( q \)  \( \text{Min}[M,N] \)

\( Q \)  Rotation matrix of eigenvectors after similarity transformation

\( Q_i \)  Reactor inlet flow rate (Section 4.6)

\( Q_o \)  Reactor outlet flow rate (Section 4.6)

\( r_j \)  \( j^{th} \) element of the model residual vector

\( r_i \)  Kernel of the inverse problem at point \( i \)

\( R \)  Product (Section 4.6)

\( R \)  Design matrix of kernels for the inverse problem

\( R \)  Universal gas constant (equation 2.116)

\( \text{sgn} \)  sign function

\( S \)  Diagonal matrix of singular values for SVD and GSVD

\( s \)  singular values

\( t_j \)  \( j^{th} \) centre of the RBF network (Chapters 4-6)

\( t \)  An instance or an experiment (Chapter 2)

\( T \)  Temperature (Equation 2.116)

\( T_i \)  Reactor inlet temperature (Section 4.6)

\( T_o \)  Reactor outlet temperature (Section 4.6)

\( u \)  Underlying function of the inverse problem

\( U \)  \( N \times M \) orthogonal matrix of SVD

\( V \)  \( M \times M \) orthonormal matrix of SVD
$V$  Volume of the reactor (Section 4.6)

$w_j$  Synaptic weight of the $j^{th}$ neurone

$\tilde{w}_j$  Linear coefficient of the $j^{th}$ polynomial bases representing global features

$\tilde{w}$  Linear weights of complete RBF network with global terms

$x$  Input (predictor) vector

$X$  Invertible matrix for GSVD

$y_i$  Measured output

$\tilde{y}(x)$  True response

$\hat{y}(x)$  Predicted response

$z$  Projected inputs

**Greek's:**

$\alpha$  Linear parameters of the model

$\beta$  Transformed linear coefficients

$\beta$  Non-linear parameters of the model

$\beta$  Scalar of the Frechet differentiation

$\chi^2$  Chi-squared error criterion

$\chi$  Critical value of the chi-square distribution

$\epsilon_i$  Measurement error for $i^{th}$ observation.

$\nu$  Variance of the model residuals

$\gamma$  Constraint constant (Chapter 2)

$\gamma$  Regularisation parameters of the univariate functions (Chapter 5)
$\mathcal{K}$ Batch k-means clustering algorithm objective function

$\varphi(.)$ Arbitrary non-linear basis (activation) function

$\Phi$ Design matrix

$\Phi_0$ Regularisation matrix

$\lambda$ Lagrangian multiplier (regularisation parameter)

$\nu$ Norm of the solution vector for linear parameters

$\rho(\varepsilon_i)$ Distribution of measurement error at point $i$

$\delta$ Weighted difference of measurement responses and the model predictions

$\delta$ Dirac delta function

$\eta(\delta_i)$ Negative logarithm of the probability density distribution at point $i$

$\psi$ Derivative of $\eta(\delta_i)$ with respect to its argument (chapter 2)

$\psi$ Polynomial bases for the global features (chapters 3-6)

$\psi$ Forcing function (chapter 3)

$\Psi$ Cholesky decomposition of the Constraint matrix $\Omega$

$\Psi$ Global part of the Green's matrix $G$

$\eta$ Learning rate constant for gradient descent

$\rho$ Tolerance for the OLS algorithm

$\Im$ Total objective function

$\Re$ Domain or range indicator

$\sigma_i$ Variance of distribution $\rho(\varepsilon_i)$

$\sigma_j$ Width of the $j^{th}$ radial basis function (Chapter 2)

$\Sigma$ Covariance matrix $= C^T C$

$\Lambda$ Diagonal matrix of eigenvalues after similarity transformation
$\Omega$ Constraint matrix
$\theta$ Threshold
$\theta^l$ Fractional adsorption
$r$ Reactor time constant (Section 4.6)

**Mathematical operators:**

- $D[.]$ Linear differential operator
- $D^* [.]$ Adjoint of operator $D[.]$
- $\mathfrak{S}$ Total operator (Chapter 3)
- $d$ Ordinary differential operator
- $\partial$ Partial differential operator
- $L[.]$ Linear differential operator
- $\| . \|$ Euclidean norm

**Indices:**

- $\tilde{x}$ Vector notation
- $\alpha^*$ Optimal value
- $D^*$ Adjoint operator (Chapters 3-6)
- $\hat{y}$ Estimated value
- $\bar{y}$ True value
- $G^T$ Transpose of matrix $G$
- $G^{-1}$ Inverse of matrix $G$
- $\bar{F}$ Approximated solution
Contents

Title
Acknowledgement
Nomenclature
Contents
Abstract

Chapter 1: Introduction
1.1 Historical Review of Neural Networks
1.2 Classification of Neural Networks
1.2.1 Structural Classification
1.2.2 Learning algorithms (Unsupervised and Supervised learning)
1.3 Scope of this study

Chapter 2: Function Approximation and Neural Networks
2.1 Introduction
2.2 Modelling of Data
2.3 Fitting the Model: Best-fit Parameters
2.3.1 Robust Estimation
2.4 Generalisation and over-fitting
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.4.1</td>
<td>Singular Value Decomposition: ill-conditioned least square problems</td>
<td>43</td>
</tr>
<tr>
<td>2.4.2</td>
<td>Ridge Regression</td>
<td>50</td>
</tr>
<tr>
<td>2.4.3</td>
<td>Constrained Least Squares: Generalised singular value decomposition</td>
<td>53</td>
</tr>
<tr>
<td>2.4.4</td>
<td>Penalised Least squares: Smoothness constraint</td>
<td>58</td>
</tr>
<tr>
<td>2.5</td>
<td>Univariate Linear Regularisation Technique</td>
<td>61</td>
</tr>
<tr>
<td>2.5.1</td>
<td>An Illustrative Example: Adsorption on a Heterogeneous Surface</td>
<td>70</td>
</tr>
<tr>
<td>2.6</td>
<td>Cross Validation: Automatic selection of the regularisation parameter</td>
<td>73</td>
</tr>
<tr>
<td>2.7</td>
<td>Conclusion</td>
<td>82</td>
</tr>
</tbody>
</table>

**Chapter 3: Regularisation of Multivariate RBF Networks**

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Multivariate Linear Regularisation Theory</td>
<td>85</td>
</tr>
<tr>
<td>3.1.1</td>
<td>Ill-Posed Problems and the Need for Regularisation</td>
<td>89</td>
</tr>
<tr>
<td>3.1.2</td>
<td>Linear Differential Operators and Green's Function</td>
<td>91</td>
</tr>
<tr>
<td>3.1.3</td>
<td>Solution of the Multivariate Linear Regularisation Problem</td>
<td>94</td>
</tr>
<tr>
<td>3.2</td>
<td>The Regularisation Network</td>
<td>99</td>
</tr>
<tr>
<td>3.2.1</td>
<td>Regularisation Network with Isotropic Spreads</td>
<td>101</td>
</tr>
<tr>
<td>3.2.2</td>
<td>Regularisation network with non-isotropic spreads</td>
<td>102</td>
</tr>
<tr>
<td>3.2.3</td>
<td>Regularisation network with linear superposition of different basis functions</td>
<td>105</td>
</tr>
<tr>
<td>3.3</td>
<td>Automatic selection of the regularisation parameter (Revisited)</td>
<td>108</td>
</tr>
<tr>
<td>3.4</td>
<td>An illustrative example</td>
<td>112</td>
</tr>
<tr>
<td>3.5</td>
<td>Conclusions</td>
<td>123</td>
</tr>
</tbody>
</table>
Chapter 4: Optimisation of Generalised RBF Networks  

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>Literature Review</td>
<td>127</td>
</tr>
<tr>
<td>4.2</td>
<td>Generalised Radial Basis Function Networks</td>
<td>148</td>
</tr>
<tr>
<td>4.3</td>
<td>Design and Training of Generalised RBF Network</td>
<td>153</td>
</tr>
<tr>
<td>4.4</td>
<td>Independent selection of the Generalised RBF network</td>
<td>154</td>
</tr>
<tr>
<td></td>
<td>spreads and centres</td>
<td></td>
</tr>
<tr>
<td>4.4.1</td>
<td><em>A priori</em> selection of spreads for given centres</td>
<td>155</td>
</tr>
<tr>
<td>4.4.2</td>
<td><em>A priori</em> and supervised selection of centres for given spreads</td>
<td>157</td>
</tr>
<tr>
<td>4.4.3</td>
<td>Results for an illustrative example</td>
<td>158</td>
</tr>
<tr>
<td>4.4.3.1</td>
<td>Pre-specified Selection of Centres</td>
<td>159</td>
</tr>
<tr>
<td>4.4.3.2</td>
<td>Self Organised Selection of Centres</td>
<td>165</td>
</tr>
<tr>
<td>4.4.4</td>
<td>Supervised Selection of Centres</td>
<td>171</td>
</tr>
<tr>
<td>4.5</td>
<td>Training of Generalised RBF networks viewed as an optimisation problem</td>
<td>175</td>
</tr>
<tr>
<td>4.5.1</td>
<td>Hierarchical simultaneous optimisation of Generalised RBF networks</td>
<td>175</td>
</tr>
<tr>
<td>4.5.2</td>
<td>Sequential optimisation of Generalised RBF networks</td>
<td>181</td>
</tr>
<tr>
<td>4.5.2.1</td>
<td>Sequential back-fit algorithm for constructing the entire RBF network</td>
<td>189</td>
</tr>
<tr>
<td>4.6</td>
<td>A Steady State Isothermal Continuous Stirred Tank Reactor</td>
<td>195</td>
</tr>
<tr>
<td>4.7</td>
<td>Conclusions</td>
<td>199</td>
</tr>
</tbody>
</table>
Chapter 5: Separation of Local and Global Features: Adaptive Generalised RBF networks

5.1 Introduction 202
5.2 Training of a Generalised RBF network with null space terms 206
5.3 An illustrative example 210
5.3.1 Generalised RBF network with precisely known null space term 213
5.3.2 Generalised RBF network with assumed bases for the null space term 216
5.4 An Adaptive Generalised RBF network 221
5.4.1 B-splines 223
5.4.2 Estimation of the coefficients 227
5.5 Conclusions 232

Chapter 6: Conclusions and Suggestions for Future Work 235

6.1 Conclusions 235
6.2 Suggestions for Future Work 241

Appendix A 243

Applications of Neural Networks in Chemical Engineering

References 250
ABSTRACT

The optimisation and adaptation of single hidden layer feed-forward neural networks employing radial basis activation functions (RBFNs) was investigated. Previous work on RBFNs has mainly focused on problems with large data sets. The training algorithms developed with large data sets prove unreliable for problems with a small number of observations, a situation frequently encountered in process engineering. The primary objective of this study was the development of efficient and reliable learning algorithms for the training of RBFNs with small and noisy data sets.

It was demonstrated that regularisation is essential in order to filter out the noise and prevent over-fitting. The selection of the appropriate level of regularisation, \( \lambda^* \), with small data sets presents a major challenge. The leave-one-out cross validation technique was considered as a potential means for automatic selection of \( \lambda^* \). The computational burden of selecting \( \lambda^* \) was significantly reduced by a novel application of the generalised singular value decomposition.

The exact solution of the multivariate linear regularisation problem can be represented as a single hidden layer neural network, the Regularisation Network, with one neurone for each distinct exemplar. A new formula was developed for automatic selection of the regularisation level for a Regularisation Network with given non-linearities. It was shown that the performance of a Regularisation Network is critically dependent on the non-linear parameters of the activation function employed; a point which has received surprisingly little attention. It was demonstrated that a measure of the effective degrees of freedom \( \text{df}(\lambda^*, \sigma) \) of a Regularisation Network can be used to select the appropriate width of the local radial basis functions, \( \sigma \), based on the data alone.

The one-to-one correspondence between the number of exemplars and the number of hidden neurones of a Regularisation Network may prove computationally prohibitive. The remedy is to use a network with a smaller number of neurones, the Generalised Radial Basis Function Network (GRBFN). The training of a GRBFN ultimately settles down to a large-scale non-linear optimisation problem. A novel sequential back-fit algorithm was developed for training the GRBFNs, which enabled the optimisation to proceed one neurone at a time. The new algorithm was tested with very promising results and its application to a simple chemical engineering process was demonstrated.

In some applications the overall response is composed of sharp localised features superimposed on a gently varying global background. Existing multivariate regression techniques as well as conventional neural networks are aimed at filtering the noise and recovering the overall response. An initial attempt was made at developing an Adaptive GRBFN to separate the local and global features. An efficient algorithm was developed simply by insisting that all the activation functions which are responsible for capturing the global trend should lie in the null space of the differential operator generating the activation function of the kernel based neurones. It was demonstrated that the proposed algorithm performs extremely well in the absence of strong global input interactions.
Chapter 1

Introduction

*The sad thing about artificial intelligence is that it lacks artifice and therefore intelligence.*

Jean Baudrillard

Conventional engineering systems behave remarkably rigidly when compared to biological ones. They lack the ability to learn from their environment or adapt gracefully to new situations compared to even the simplest biological organisms [Chester, 1993; Baughman and Liu, 1996]. This low level of achievement is due to the fundamental differences between the design of biological nervous systems and present day computers [Lerner, 1984; Omohundro, 1987]. To bridge this gap researchers in the fields of neurophysiology, psychology and computer science are seeking to understand and implement the mechanisms of intelligence.

---

Artificial neural networks may be regarded as initial attempts at developing artificial intelligence. Similar to their biological counterparts, these networks are composed of many simple interconnecting elements, or neurones, working in parallel to solve a problem. The most versatile feature of them is the fact that once the network has been set up, it can learn in a self-organised way that seems to mimic simple biological nervous systems.

Numerous companies and research laboratories are investing heavily in projects related in some way to neural networks [Omohundro, 1987]. Despite the tremendous increase in applications of neural networks in many fields such as electrical, electronics, civil, and control engineering, they were practically unknown to many chemical engineers until the 1990’s. Serious efforts to apply neural networks for the simulation and optimisation of chemical, biochemical and mineral processes have only begun since the late 1980’s. A concise review of the reported applications of neural networks to chemical and bio-chemical engineering problems is presented in Appendix A. In this chapter we first present a brief historical review and a classification of neural networks and then outline the scope of the work undertaken in this study.
1.1- Historical Review of Neural Networks

In the early forties (1943) W.S. McCulloch, a neurophysiologist, and W.Pitts, a mathematician, presented a simple mathematical model for a nerve cell [Chester, 1993; Haykin, 1999]. The *McCulloch-Pitts* neurone, which is shown in Figure 1.1 and is regarded as the ancestor of all artificial neural networks, performed logical operations on two or more inputs to produce an output. The variable inputs were multiplied by fixed synaptic weights and the products were summed. If the sum exceeded the neurone's threshold ($\theta$), the neurone turned (or stayed) on, otherwise it turned (or stayed) off.

Figure 1.1 - Schematic diagram of a *McCulloch-Pitt's* neurone.

In the late forties Donald Hebb a neurophysiologist, used the *McCulloch-Pitt* neurone with a variable synaptic weight for a single input [Chester, 1993; Haykin, 1999]. In 1958, Frank Rosenblatt combined the *McCulloch-Pitt* and *Hebb* neurones to construct the *Perceptron* on an analogue computer, which became the essential building block of neural networks in the 1980's and 1990's. Figure 1.2 gives a schematic diagram of the Rosenblatt's neurone known as Perceptron.
Figure 1.2 - Schematic diagram of Rosenblatt’s neurone (Perceptron),
(a) Forward path (calculation of the neurone’s response),
(b) Backward path (Updating the synaptic weights)
The training loop suggested by Rosenblatt involved the following steps:

a) Initialise the synaptic weights of the neurone (Perceptron).

b) Introduce an input: \( x(t) = [x_1(t), x_2(t), \ldots, x_p(t)]^T \)

c) Calculate the output of the neurone: \( \hat{y}(t) = \text{sgn} \sum_{i=1}^{p} [w_i(t)x_i(t) - \theta] \)

d) Adjust the synaptic weights: \( w_i(t+1) = w_i(t) + \eta[y(t) - \hat{y}(t)]x_i(t) \)

e) Repeat from (b) until the synaptic weights stop changing.

In the above procedure, \( t \) represents an instance or iteration, \( y(t) \) is the target value associated with the input \( x(t) \), and the \( \text{sgn} \) function is \(+1\) if its argument is positive and \(-1\) if its argument is negative or zero. The symbol \( \eta \) represents a “small” positive learning rate constant chosen to ensure a smooth synaptic change.

In the early sixties (1960, 1962), Bernard Widrow and Marcian Hoff [Omohundro, 1987; Hunt et al., 1992; Haykin, 1999] introduced the Adaptive Linear Neurone “Adaline” by omitting the \( \text{sgn} \) function and calculating the response of the neurone as

\[
\hat{y}(t) = \sum_{i=1}^{p} [x_i(t)w_i(t) - \theta]
\]

The major difference with the Perceptron is that removing the \( \text{sgn} \) enables a continuous rather than binary output. Adaline employed the same learning rule as the Perceptron for correction of the synaptic weights,

\[
w_i(t+1) = w_i(t) + \eta[y(t) - \hat{y}(t)]x_i(t)
\]
Because of the change in the definition of $\hat{y}(t)$, the correction of the weights of the "Adaline" are performed continuously and the convergence of the synaptic weights to their ultimate values is much faster than that for the Perceptron. This continuous application of the Rosenblatt learning rule $[\Delta w_i = \eta[y(t) - \hat{y}(t)]x_i(t)]$ has become known as the Widrow-Hoff law or the *delta rule*.


In 1982 John Hopfield of the California Institute of Technology designed a fully connected single layer neural network which revived the subject. The aim of his network was to solve an optimisation problem [Hopfield, 1982]. Subsequently, neural networks have been studied more intensively with special interests in engineering applications. Techniques based on neural networks have been developed to solve different problems ranging from banking and speech recognition to process scheduling [Chovan et al, 1996].
Despite the increased application of neural networks in various scientific topics they remained largely unknown to chemical engineers until the leading work of Bhat and McAvoy in 1990 [Bhat and McAvoy, 1990]. Appendix A presents a concise survey of the articles published to date on application of neural networks in chemical and biochemical engineering. With a few exceptions, the emphasis has been on the application of the standard neural network programs as a tool for solving various engineering problems. A review of the literature on the fundamental aspects of neural networks, in particular radial basis function (RBF) networks, is deferred to chapter 4.

1.2- Classification of Neural Networks

Neural networks generally consist of several interconnected neurones in one or more hidden layers. They can be classified from different points of views such as type of input transformation, type of structural architecture and the type of learning algorithm.

It is well known that the complexity of any modelling task increases significantly with an increase in the dimensions of the input space, which is often referred to as “the curse of dimensionality” [Box, 1962; Kruskal, 1965; Priestley and Chao, 1972; Box and Hill, 1974; Box and Cox, 1982; Breiman and Friedman, 1985]. Neural networks may employ either (non-local) projection or (local) kernel based transformations to account for correlation among the inputs.
In a non-local transformation the inputs are projected on a single axis, the projection may be linear or non-linear. The McCulloch-Pitt neurone, Perceptron and Adaline (section 1.2) are examples of linear projections. Figure 1.3 illustrates the general schematic diagram of a linear projection based neurone. In local transformations, the norm (usually Euclidean) of the input vector with respect to a fixed point (centre) is used to transform the inputs. Radial basis function networks are the most popular examples of kernel based input transformations. Figure 1.4 illustrates the schematic diagram of a general kernel based neurone.

![Figure 1.3](image1.png)

**Figure 1.3**—Schematic diagram of a general linear projection based neurone.

![Figure 1.4](image2.png)

**Figure 1.4**—Schematic diagram of a general kernel based neurone.
1.2.1 - Structural Classification

Neural networks are commonly classified based on the direction of the signal flow into feed-forward and recurrent networks. In a feed-forward network signals flow from the input layer, to hidden layer(s) and then to the output layer via unidirectional connections. The neurones are connected from one layer to the next but not within the same layer. These networks can most naturally perform static mappings between input and output spaces. In other words, the output of a feed-forward network at a given instant is only a function of its input at the same instant. In general, two different classes of single and multiple layer feed-forward network architectures can be identified.

In its simplest form, a feed-forward network is constructed from an input layer of source nodes that are projected onto an output layer of computation nodes via synaptic weights. Figure 1.5 illustrates a single layer feed-forward network with four nodes in the input layer and two nodes in the output layer. No computation is performed within the input layer and it is not counted as an independent layer.

![Synaptic Weights](image)

*Figure 1.5* - Single layer Feed-forward neural network.
The “single layer” designation refers to the output layer containing feed-forward computation nodes (neurons). A linear associative memory is an example of a single-layer neural network, the network associates an output pattern (vector) with an input pattern (vector), and the information is stored in the network by virtue of the modifications made to the synaptic weights of the network.

Multi-layer feed-forward neural networks contain one or more hidden layer(s), whose corresponding computational nodes are called hidden neurones or hidden units. The function of the hidden neurones is to intervene between the external input and the network output. By adding one or more hidden layers, the network is enabled to extract higher order statistics (more information) by virtue of the extra set of synaptic connections and increased neural interactions. The ability of the hidden neurones to extract higher order statistics is crucial for large input dimensions. As illustrated in Figure 1.6, the neurones of each layer may be either partially or fully connected to the neighbouring layers.

In a recurrent network, the outputs of some neurones are fed-back to the neurones of the same layer or to the nodes of the preceding layers. The signals can therefore flow in both forward and backward directions. Recurrent networks have dynamic memories, that is their outputs at a given instant reflect both the current input as well as past inputs and outputs. A typical recurrent network is shown in Figure 1.7. Due to their dynamic memory, recurrent neural networks are particularly suited for control applications and dynamic simulations.
Figure 1.6 - Feed-forward neural networks with one hidden layer and output layer. (a) Fully connected (b) Partially connected

Figure 1.7 - Fully connected recurrent neural networks with one hidden layer.
1.2.2- Learning algorithms (Unsupervised and Supervised learning)

The learning algorithm deals with the adjustment of the network parameters and usually settles to solving an unconstrained or constrained optimisation problem. The model representing the neural network may be linear, non-linear or a combination of both with respect to the network parameters. The merit function characterising the network performance may depend on the inputs alone or both the inputs and the outputs. The former objective function leads to learning without a teacher (unsupervised learning) while the latter objective requires a teacher to direct the learning (supervised learning). The learning process of an unsupervised network does not require target (measured) output(s). Only input patterns are presented to the network during training. For a fixed architecture, the network automatically adapts the network parameters to cluster the input pattern into groups of similar features.

In many engineering applications we are concerned with the estimation of an underlying trend (or function) from a limited number of input-output data points with little or no knowledge of the form of the true function (truth). This problem is sometimes referred to as non-parametric regression, function approximation, system identification or inductive learning. In neural network parlance, it is usually called supervised learning. The underlying function is learned from the exemplars which a teacher supplies. The set of examples (the training set) contains elements that consist of paired values of the independent (input) and the dependent (output) variable(s). A supervised learning algorithm adjusts the network parameters according to the differences between the measured response \( y(x_i) \) and the network outputs \( \hat{y}(x_i) \) corresponding to a given input
Supervised learning requires a *supervisor*, to provide the target signals. In practical applications the measurements are by definition subject to error. The learning algorithm should therefore be equipped with proper provisions to effectively filter out the noise.

1.3 – Scope of this study

The specific problem addressed in this work is to reconstruct a model of a process based on a limited number of the measured values of its inputs and outputs. This is a task well suited for feed-forward neural networks. The above problem can also be viewed as a task of multivariate function approximation for reconstructing a hyper-surface which maps the process inputs into the output space. Two main approaches are available for multivariate function approximation or hyper-surface reconstruction problems. The first one is the feed-forward neural network approach which will be considered in Chapters 3 to 5. The second approach, favoured by statisticians, includes the adaptive linear additive models [Hastie and Tibshirani, 1990] and projection pursuit regression [Friedman and Stuetzle, 1981]. Bakshi and Utojo [1998] have attempted to present a unified view of these two approaches. Poggio *et al* [1993] have shown that the two approaches can be grouped together and only differ in the details of the stabiliser operators employed.

Feed-forward neural networks have a close relationship with function approximation techniques. Chapter 2 introduces various aspects of function approximation such as modelling of a given set of observations (*modelling of data*), generalisation and overfitting, and various stabilisation techniques in a simple univariate setting. Major aspects
of modelling of data for finding the optimal model parameters by classical methods such as ordinary least squares regression and robust estimation are considered with the aid of simple examples.

The fitted model is expected not only to recall the observed data with the required accuracy, but also to produce acceptable predictions for unseen (test) data drawn from the same population as the observed data. For a set of noisy data, naïve minimisation of the ordinary least squares merit function may fit the noise and lead to an unrealistic and highly oscillatory prediction. Stabilisation techniques are crucial for filtering out the noise and capturing the true underlying function from a set of limited, sparse and noisy data. A variety of indirect and direct stabilisation methods such as singular value decomposition (SVD), ridge regression, penalised least squares and linear regularisation techniques are considered in Chapter 2. In particular it will be shown that linear regularisation provides the most effective procedure to filter out the noise and recover the true trend underlying the data. It will also be demonstrated that any stabilisation technique is only effective if the selected model is appropriate for the given observations.

A difficult problem in the application of any stabilisation technique is deciding on the appropriate level of stabilisation. Too small a stabilisation level leaves the noise unaffected and too high a level may lead to severe over-smoothing. The appropriate level of stabilisation is highly problem dependent and its simplistic selection may prove subjective. An objective method is therefore desired for the selection of the stabilisation
level based on the data alone. A common procedure is to divide the measured data into distinct training and test sets followed by the *a posteriori* selection of the stabilisation level using the test set alone. This approach can not be afforded when the number of data points available is limited, a situation frequently encountered in chemical engineering.

A chief advantage of the linear regularisation technique is the possibility of developing efficient procedures for the automatic selection of the appropriate regularisation level based on the data alone. A promising approach, the *leave one out cross validation* (CV) technique which makes full use of all the available data is described in Chapter 2. An efficient procedure based on the generalised singular value decomposition (GSVD) will be developed which can be used for a wide variety of linear stabilisation operators. The performance of the proposed procedure is assessed for the recovery of the unknown energy distribution of a heterogeneous solid adsorbent. To our knowledge the combination of the leave one out cross validation technique and generalised singular value decomposition has not been previously reported and is used to good effect throughout this study.

Another property, not enjoyed by other stabilisation methods, is the theoretical possibility of extending the linear regularisation technique to problems with multiple inputs. Chapter 3 provides an introduction to the multivariate regularisation theory which serves as a solid theoretical foundation for an important class of feed-forward neural networks employing radial basis functions. The concepts of a linear differential operator and its corresponding Green’s function are touched upon and lead to the development of
the so-called *Regularisation network* as a means of representing the solution of the multivariate regularisation problem. The leave one out cross validation technique of Chapter 2 is also extended and a new working formula is developed for the automatic selection of the regularisation level of a Regularisation network.

The number of neurones in the hidden layer of a Regularisation network is by definition equal to the number of data points. The activation function of each neurone is a radial basis function (RBF) which is centred at a data point and also contains other parameters which establish the receptive field of the neurone. The number of such parameters depends on the linear stabilisation operator employed. The linear operator can also be chosen to produce an RBF with either a local or a global receptive field. In this study we are particularly interested in separating the global and local features embedded in a data set and therefore concentrate on RBF networks with a local receptive field, in particular Gaussian functions. The receptive field of a Gaussian is established by the choice of spreads.

Regularisation networks employing Gaussian activation functions are usually applied using the same identical spread for all neurones. An illustrative bivariate example is used to highlight the significant influence of the choice of the spread on the performance of Regularisation networks. It will be demonstrated that the performance of the Regularisation network in a given application is critically dependent on the choice of the spread(s), a point which has surprisingly received little attention to date. A simple and readily calculable measure of the approximate degrees of freedom of a Regularisation
network is introduced. Combination of the leave one out cross validation technique and the approximate degrees of freedom provides a convenient criterion for the selection of a suitable value of the spread. The application of this procedure results in significant improvement in the performance of the Regularisation network with an isotropic spread and to our knowledge has not been previously reported.

A major drawback of the Regularisation network is the one-to-one correspondence between the number of input exemplars \( N \) and the number of hidden neurones. The computational cost of a Regularisation network grows as \( N^3 \) and may prove prohibitive for large data sets. To ease the computational burden it is necessary to break the one-to-one correspondence by using a smaller number of hidden neurones \( M << N \), such networks are referred to as Generalised RBF networks. Decreasing the number of neurones reduces the flexibility of the network and it becomes essential to distinguish the centres from the data points and choose the centres and spreads of the activation functions carefully. The training of the Generalised RBF network ultimately settles down to a large scale non-linear optimisation problem. This problem has received considerable attention over the past decade. We start Chapter 4 with a thorough review of the literature in order to place this study in the context of previous work.

The synaptic weights of a Generalised RBF network appear linearly and their calculation for given non-linear parameters is fairly straightforward. Chapter 4 focuses on testing and developing suitable procedures for the optimisation of the centres and spreads which appear non-linearly. A number of heuristic procedures have been proposed to
avoid the non-linear optimisation process and produce readily calculated values of the centres and/or spreads. An illustrative example is used to highlight the limited reliability of heuristic procedures and the need to tackle the non-linear optimisation directly.

The major difficulty is the number of non-linear parameters involved. For a Generalised RBF network with 20 neurones and a 5-dimensional input vector, the number of non-linear parameters may be as much as 620. Several procedures are considered to break down the large-scale non-linear optimisation problem into a series of smaller and therefore more manageable sub-problems. In particular, a novel sequential back-fit algorithm is developed which enables the optimisation to proceed one neurone at a time. This new algorithm is tested with very promising results and its application to a simple chemical engineering process is also presented.

In some applications the overall response is composed of sharp localised features superimposed on a slowly varying global background. Existing multivariate regression techniques as well as conventional neural networks are aimed at filtering the noise and recovering the overall response. Chapter 5 is an initial attempt at developing an adaptive Generalised RBF network capable of separating the local and global features embedded in a noisy data set. The ultimate aim is to produce a neural network containing two types of neurone: i) a set of projection based neurones, which are better suited for capturing the gently varying global background, and ii) a set of localised kernel based neurones which are more appropriate for the recognition of sharp local features.
An obvious procedure suggested by a few authors [Zhang and Morris, 1998; Holcomb and Morari 1992; Gemperline et al, 1991] is to select different classes of neurones with different activation functions, for example Gaussian, Sigmoidal and Linear functions. The major difficulty with this approach is that an initial decision is required about the number of hidden neurones of each class. The training of such networks proves difficult and may involve tedious trial and error to select both the type and the number of each type of neurone in the hidden layer. In addition, the interaction between the various classes of neurones is difficult to quantify and complicates the training process further. In Chapter 5 we present an approach which de-couples the contribution of the projection based neurones from the kernel based neurones. This is simply achieved by insisting that all the activation functions of the projection based neurones lie in the null space of the differential operator generating the activation function of the kernel based neurones. To our knowledge, the inclusion of null space terms which simplifies the training process significantly has not been previously reported.

An illustrative example is used to highlight the important influence of the null space terms on the performance of Generalised RBF networks. This is followed by a general consideration of various procedures for the inclusion of null space terms in the training procedure under a variety of assumptions. Chapter 5 closes with the development of an adaptive Generalised RBF network and the associated back-fit training algorithm which enables the null space terms to evolve automatically.
All the results presented in this thesis were obtained using self-contained computer codes developed by the author in the course of this study. The only external routines used were for the singular value and generalised singular value decomposition which were taken from the freely available LAPACK collection of routines [www.netlib.org] and various optimisation procedures used in section 4.4.3.2 which are available from Numerical Recipe [Press et al, 1992]. All the codes developed in ANSI Standard Fortran 77 and are available on request from the author.
Chapter 2

Function Approximation and Neural Networks

2.1 – Introduction

The general aspects of neural networks were briefly highlighted in the previous chapter. The specific problem addressed in this study is to reconstruct a model of a process based on a limited set of measured values of its input and outputs. This is a task well suited for feed-forward neural networks. There is a close relation between feed-forward neural networks and function approximation techniques. This chapter reviews the basic ideas of function approximation such as ordinary least square, robust estimation and some stabilisation techniques such as ridge regression, penalised least square and linear regularisation methods. These techniques provide the theoretical foundations for a special class of feed-forward networks known as radial basis function (RBF) networks which are considered in the next chapter.

The primary aim of this chapter is to highlight the basic concepts involved in developing a function to approximate or model the true underlying relationship between the inputs and the outputs of a process. For illustration purposes, we shall focus on a process with a single output variable, the generalisation to multiple outputs is straightforward. Consider the process shown below, the objective is to establish the dependence of the output $y(t)$ on the inputs $x(t)$, where $t$ represents an event, instance or an experiment.
The relationship between the true response \( \tilde{y}(t) \) and the independent variables \( x(t) \) can be expressed as,

\[
\tilde{y}(t) = \tilde{f}(x_1(t), x_2(t), \ldots, x_p(t))
\]  

(1)

The form of the function \( \tilde{f}(\cdot) \) is in general unknown but we can conduct a series of experiments to sample the truth. Experiments are by definition subject to error and the relationship between the true \( \tilde{y}(t) \) and the measured \( y(t) \) values may be expressed as,

\[
y(t) = \tilde{y}(t) + \epsilon(t)
\]  

(2)

where \( \epsilon(t) \) is the measurement error which is by definition uncorrelated with \( \tilde{y}(t) \). There is a random nature associated with measurement, if we repeat the same experiment twice we do not get the same measured value. The error \( \epsilon(t) \) is a random sample drawn from a particular distribution which is an inherent feature of the measurement device. The basic task is to estimate the truth given a set of measured values \( x(t_i), y(t_i); i = 1, 2, \ldots, N \) and sufficient information about the distribution of the measurement errors (noise). Evidently, to achieve this task the data set must be sufficiently dense with an adequate signal to noise ratio. Since the truth \( \tilde{y}(t) \) is in general unknown, we seek a model capable of providing a reliable estimate of the true underlying relationship based on the available observations.
2.2 – Modelling of Data

The model can be expressed as,

\[ \hat{y}(x, a) = \hat{f}(x_1, x_2, \ldots, x_p; a_1, a_2, \ldots, a_m), \]  

(3)

where the \( a_j \)'s are a set of adjustable parameters. Selecting a model as the estimator of the truth requires the selection of a functional form for \( \hat{f}(\cdot) \), which must be chosen to reflect the true underlying process as closely as possible, followed by the determination of the optimal model parameters \( a^* \) by minimising a suitably defined merit function measuring the agreement with the measured data.

Modelling of a process covers a broad spectrum. At one extreme lies theoretical (or parametric) models based on a fundamental knowledge of the process, such models have physically meaningful and measurable parameters. At the other end lies empirical (or non-parametric) models which do not rely on the fundamental principles governing the process. Such models involve a set of parameters with no physical meaning related to the actual process. In practice, a compromise between these two extremes (a semi-theoretical model) is employed to describe a real process. Semi-theoretical models provide a compromise between the model complexity and the effort needed for the measurement of the parameters. In this approach, parts of the model are approximated by empirical correlations based on curve fit to available experimental data. Evidently, the parameters based on theoretical concepts may be well defined but the empirical parameters may have little or no physical meaning. Error analysis of semi-theoretical models proves difficult because the final error in the model predictions may be due to inappropriate theoretical assumptions or inaccurate empirical correlations or both.
In an alternative purely empirical approach, the model can be considered as a linear combination of a set of basis functions. The familiar multiple linear regression model is the simplest example employing a set of linear basis functions [Hastie et al., 1990],

\[ \hat{y}(x,a) = a_1 x_1 + a_2 x_2 + \ldots + a_p x_p \]  

(4)

Given a set of measured data, the optimal parameters are readily obtained using standard least square minimisation techniques. If our a priori belief is that there is no interaction between the inputs and the true underlying response varies linearly with each individual input, this model provides a very convenient and powerful data analytic tool. In particular the multiple linear regression model provides a very simple and compact description of the data, summarises the contribution of each input to the response variable with a single coefficient, and enables a simple method for predicting the response at new observations.

The multiple linear regression model is inappropriate if the true response is non-linear with respect to one or more inputs or if there is significant interaction between the inputs. A useful generalisation of the linear regression model is obtained by using,

\[ \hat{y}(x,\alpha) = \sum_{j=1}^{P} \alpha_j \phi_j(x_j) \]  

(5)

where \( \phi_j(.) \) can be chosen as any arbitrary non-linear function of the single predictor \( x_j \). This model, referred to as the Linear Additive Model, is linear in the parameters \( \alpha_j \)'s but may be highly non-linear with respect to the independent variables \( x_j \)'s. It retains many of the useful features of the multiple linear regression model. In particular, the contribution of each predictor is clearly well defined and can be explored
independently. The optimal parameters can again be easily determined through standard least squares methods.

Neither the multiple linear regression model nor the linear additive model is capable of explicitly accounting for input interactions and will perform badly when such interactions are important. To allow for interactions explicitly it becomes necessary to introduce a new set of non-linear parameters in the basis functions leading to a model in the form of [Bakshi and Utojo, 1998],

\[ \hat{y}(x, \alpha, \beta) = \sum_{j=1}^{M} \alpha_j \phi_j(x, \beta_j) \]  

(6)

The model is always linear with respect to the \( \alpha_j \)'s but may be non-linear with respect to the \( \beta_j \)'s. The linear parameters \( \alpha_j \)'s are usually referred to as weights. The close connection between feed-forward neural networks and function approximation is captured in Figure 2.1 which represents (6) as a neural network with a single hidden layer.

The individual basis functions, \( \phi_j(.) \)'s may be chosen to have either a local or a global response. The response of a local basis function is restricted to a limited range about some central point. Consequently, only some of the local basis functions can affect the overall response at a given point. Gaussian functions shown in Figure 2.2 are a popular example of local basis functions.
**Figure 2.1** – Schematic representation of a feed-forward neural network.

**Figure 2.2** – Linear superposition of Gaussian basis functions for a univariate example.
The response of a global basis function is distributed over the whole input-space and affects the overall response at every point. Polynomials are the best known example of global basis functions. It is of course possible to choose some of the basis functions as local while others may be chosen as global. There is no limitation on the form of individual basis functions but the weighted sum of the basis functions must reconstruct the true underlying response.

In the case of multiple independent variables, the non-linear parameters $\beta_j$'s are used to transform the input vector into a scalar argument for the basis function $\phi_j(\cdot)$. This is usually achieved in one of two ways:

a) *Projection based (non-local) input transformation*: The parameters $\beta_j$'s project the inputs on a hyper-surface, this projection may be either linear or non-linear as illustrated in fig. 2.3a.

b) *Kernel based (local) input transformation*: The parameters $\beta_j$'s are used to define a scalar norm measure (usually Euclidean) of the input vector with respect to a fixed point (centre) as depicted in Figure 2.3b.
Once the input transformation is specified, it remains to choose the functional form of each basis function with respect to its scalar argument. This can again be achieved in two ways:

a) **Fixed form basis function**: a particular functional form is chosen to represent the variation of the basis function with its scalar argument. The optimal (linear and non-linear) parameters are obtained for this particular fixed functional form.

b) **Flexible form basis function**: no particular functional form is chosen *a priori*. Both the optimal parameters and the appropriate functional form of the basis function emerge in the course of the optimisation process.

We shall have more to say about Flexible or Adaptive form basis functions in Chapter 5 and for the moment concentrate on Fixed form basis functions. Table 2.1 and Figure 2.4 present a number of fixed form basis functions commonly employed in function approximation and neural network applications.
Table 2.1 – Examples of fixed form basis functions.

<table>
<thead>
<tr>
<th>Description</th>
<th>Input Transformation</th>
<th>Equation</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gaussian</td>
<td>( z^2 = \frac{p}{i=1} \frac{(x_i - t_i)^2}{\sigma_i^2} )</td>
<td>( \phi(z) = \exp(-z^2) )</td>
<td>Local, Kernel</td>
</tr>
<tr>
<td>Multiquadrics</td>
<td>( z^2 = \sum_{i=1}^p (x_i - t_i)^2 + \sigma^2 )</td>
<td>( \phi(z) = z^2 )</td>
<td>Global, Kernel</td>
</tr>
<tr>
<td>Inverse Multiquadrics</td>
<td>( z^2 = \sum_{i=1}^p (x_i - t_i)^2 + \sigma^2 )</td>
<td>( \phi(z) = \frac{1}{z^2} )</td>
<td>Local, Kernel</td>
</tr>
<tr>
<td>Thin plate splines</td>
<td>( z^2 = \frac{p}{i=1} \frac{(x_i - t_i)^2}{\sigma_i^2} )</td>
<td>( \phi(z) = z^{2n+1} )</td>
<td>Global, Kernel</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( \phi(z) = z^{2n} \cdot \ln(z) )</td>
<td></td>
</tr>
<tr>
<td>Threshold</td>
<td>( z = \sum_{j=1}^p \beta_j x_j + \beta_0 )</td>
<td>( \phi(z) = \begin{cases} 1 &amp; \text{if } z \geq 0 \ 0 &amp; \text{if } z &lt; 0 \end{cases} )</td>
<td>Global, Projection</td>
</tr>
<tr>
<td>Piecewise-Linear</td>
<td>( z = \sum_{j=1}^p \beta_j x_j + \beta_0 )</td>
<td>( \phi(z) = \begin{cases} 1 &amp; \text{if } z \geq 1 \ z &amp; \text{if } -1 &lt; z &lt; 1 \ -1 &amp; \text{if } z \leq -1 \end{cases} )</td>
<td>Global, Projection</td>
</tr>
<tr>
<td>Sigmoid</td>
<td>( z = \sum_{j=1}^p \beta_j x_j + \beta_0 )</td>
<td>( \phi(z) = \frac{1}{1 + \exp(-z)} )</td>
<td>Global, Projection</td>
</tr>
<tr>
<td>Hyperbolic Tangent</td>
<td>( z = \sum_{j=1}^p \beta_j x_j + \beta_0 )</td>
<td>( \phi(z) = \frac{1 - \exp(-z)}{1 + \exp(-z)} )</td>
<td>Global, Projection</td>
</tr>
</tbody>
</table>

29
Figure 2.4 – Typical examples of fixed form basis functions (see Table 2.1)
2.3 – Fitting the Model: Best-fit Parameters

Given a model in the form of:

$$\hat{y}(x, \alpha, \beta) = \sum_{j=1}^{M} \alpha_j \phi_j(z_j(x, \beta_{j}))$$  \hspace{1cm} (7)

with a specified input transformation of $z_j(x, \beta)$ and a specified functional form for $\phi_j(.)$, the objective is to find a set of optimal (best-fit) linear $\alpha^*$ and non-linear $\beta^*$ parameters which minimise a suitably defined merit function for a given set of observations. In practice, the observations are corrupted by measurement noise and the procedure for obtaining the best-fit model parameters should ideally provide: (a) the optimal parameter values, (b) an error estimate (confidence interval) for the optimal parameters, and (c) a statistical measure of the goodness of fit of the model [Press, 1992].

For a given model, some parameter values will be highly unlikely, those for which the model predictions are at odds with the observed data. Others will be more likely and produce model predictions which resemble the data. We wish to select among the likely parameters those which are most likely and reproduce the closest agreement with the true underlying trend. This goal can be achieved by posing the question: “Given a model and a particular set of parameters, what is the probability that the observed data set could have occurred?” In other words we seek to compute the probability of a data point occurring within an interval $\hat{y}(x_i, \alpha, \beta) \pm \Delta y$ for a given model with a particular choice of parameters. Suppose that each observation $y_i$ has an independently random measurement error $\epsilon_i$ drawn from a certain distribution $\rho(\epsilon_i)$ centred around the true value $\hat{y}(x_i)$. 


The probability of an individual data point $y_i$ occurring within the interval $\hat{y}(x_i, \alpha, \beta) \pm \Delta y$ may be taken as,

$$ p_i \propto \rho(y_i, \hat{y}(x_i; \alpha, \beta)) \Delta y, \quad (8) $$

where $\rho(\cdot)$ is chosen to correspond to the distribution of the error $\epsilon_i$. In general, the probability density function $\rho(y_i, \hat{y}(x_i; \alpha, \beta))$ does not depend on its individual arguments independently and is only a function of their weighted difference defined as,

$$ \delta_i(\alpha, \beta) = \left( \frac{y_i - \hat{y}(x_i; \alpha, \beta)}{\sigma_i} \right) \quad (9) $$

The probability of the entire data set is the product of the probabilities of the individual data points,

$$ P(\alpha, \beta) = \prod_{i=1}^{N} \rho(\delta_i(\alpha, \beta)) \Delta y \quad (10) $$

The best-fit parameters are obtained by seeking to maximise the probability $P(\alpha, \beta)$ of the data given the model. Maximising $P(\alpha, \beta)$ is equivalent to minimising the negative of its logarithm,

$$ \min_{\alpha, \beta} A(\alpha, \beta) = -\ln(P(\alpha, \beta)) = -\sum_{i=1}^{N} \ln[\rho(\delta_i(\alpha, \beta))] - N \ln(\Delta y) \quad (11) $$

Defining $\eta(\delta_i) = -\ln[\rho(\delta_i(\alpha, \beta))]$ and dropping the constant term $N \ln(\Delta y)$ leads to,

$$ \min_{\alpha, \beta} A(\alpha, \beta) = \sum_{i=1}^{N} \eta(\delta_i(\alpha, \beta)) \quad (12) $$

Defining $\psi(\delta_i)$ as the derivative of $\eta(\delta_i)$ with respect to its argument,

$$ \psi(\delta_i) = \frac{d\eta(\delta_i)}{d\delta_i} \quad (13) $$

32
and using the optimality conditions $\frac{\partial A}{\partial \alpha} = 0$ and $\frac{\partial A}{\partial \beta_j} = 0$, leads to

$$
\sum_{i=1}^{N} \psi(\delta_i(\alpha, \beta)) \left( \frac{1}{\sigma_i} \right) \phi_j(z_j(x_i, \beta)) = 0, \quad j = 1, 2, \ldots, M
$$

(14)

and,

$$
\sum_{i=1}^{N} \psi(\delta_i(\alpha, \beta)) \left( \frac{\alpha_j}{\sigma_i} \right) \frac{d\phi_j(z_j)}{dz_j} \left( \frac{\partial z_j(x_i, \beta)}{\partial \beta_j} \right) = 0, \quad j = 1, 2, \ldots, M
$$

(15)

The optimal parameters $(\alpha^*, \beta^*)$ may be computed either by directly minimizing the multivariate objective function (12) or by solving the set of non-linear equations (14) and (15). However, except for the simple case of a normal distribution, either approach involves substantial numerical difficulties and presents a major challenge.

If the distribution of measurement error $\rho(\varepsilon_i)$ is known a priori and the probability distribution $\rho(\delta(\alpha, \beta))$ is chosen accordingly, then we are able to quantify the goodness of fit of the model directly. For a correct model, the calculated distribution of the fitted residual $r_i = y_i - \hat{y}(x_i; \alpha, \beta)$ should resemble the known distribution of the measurement error $\rho(\varepsilon_i)$. In the absence of reliable information about $\rho(\varepsilon_i)$, we are free to choose any particular form for the probability distribution $\rho(\delta_i(\alpha, \beta))$ but the goodness of fit of the model can no longer be checked independently. In the simplest case of normal measurement errors, the probability $\rho(\delta_i)$ can be taken as a Gaussian distribution with a zero mean and standard deviation $\sigma_i$,

$$
\rho(\delta_i(\alpha, \beta)) \propto \exp \left[ -\frac{1}{2} \left( \frac{y_i - \hat{y}(x_i; \alpha, \beta)}{\sigma_i} \right)^2 \right] = \exp \left( -\frac{1}{2} \delta_i^2 \right)
$$

(16)
which implies that:

$$
\eta(\delta_i(\alpha, \beta)) = \frac{1}{2} \delta_i^2
$$

(17)

and

$$
\psi(\delta_i) = \frac{d\eta(\delta_i)}{d\delta_i} = \delta_i
$$

(18)

For this case, the merit function (12) reduces to the "chi square" or the "weighted sum of squared residuals",

$$
\min_{\alpha, \beta} A(\alpha, \beta) = \frac{1}{2} \sum_{i=1}^{N} \left( \frac{y_i - \hat{y}(x_i; \alpha, \beta)}{\sigma_i} \right)^2 = \frac{1}{2} \sum_{i=1}^{N} \delta_i^2(\alpha, \beta)
$$

(19)

If the variances are assumed identical \( \sigma_i = \sigma \), the objective function reduces further to the familiar least square criterion,

$$
\min_{\alpha, \beta} A(\alpha, \beta) = \frac{1}{2} \sum_{i=1}^{N} (y_i - \hat{y}(x_i; \alpha, \beta))^2.
$$

(20)

The normal (Gaussian) distribution of errors is often rather poorly realised. This distribution implies that the probability of a measurement falling within \( \pm \sigma \) of its true value is 68\%, the probabilities for \( \pm 2\sigma \) and \( \pm 3\sigma \) are 95\% and 97\% respectively. According to the normal distribution, one would expect a data point to be off by \( \pm 10\sigma \) only one time out of \( 5.1 \times 10^{21} \) measurements. In practice, however, "way off" data are much more likely than that. The normal distribution has unrealistically narrow tails and any data point which happens to be way off (an outlier) is deemed very improbable. Consequently, the fitted model is often distorted drastically to bring the outlier point into line. To avoid such problems we can resort to robust estimation employing a more realistic distribution with a wider tail. A useful introduction to robust estimation and problems caused by outliers is given by Press et al [1992].
2.3.1 – Robust Estimation

The subject of robust statistics deals with cases where the normal distribution is a bad representation of measurement error or cases where outliers are important. A more realistic distribution with a wider tail is the double or two-sided exponential,

\[
\rho(\delta_i(\alpha, \beta)) = \exp \left[ - \frac{|y_i - \hat{y}(x_i; \alpha, \beta)|}{\sigma_i} \right] = \exp \left[ - | \delta_i(\alpha, \beta) | \right]
\] (21)

For this distribution, the function \(\eta(\delta(\alpha, \beta))\) and its derivative are given by,

\[
\eta(\delta_i(\alpha, \beta)) = | \delta_i |
\] (22)
\[
\psi(\delta_i) = \frac{d\eta(\delta_i)}{d\delta_i} = \text{sgn}(\delta_i)
\] (23)

and the merit function reduces to the sum of absolute deviations,

\[
\min_{\alpha, \beta} A(\alpha, \beta) = \sum_{i=1}^{N} \left| \frac{y_i - \hat{y}(x_i; \alpha, \beta)}{\sigma_i} \right| = \sum_{i=1}^{N} | \delta_i(\alpha, \beta) |.
\] (24)

A distribution with an even more extensive tail is the Cauchy or Lorentzian distribution,

\[
\rho(\delta_i(\alpha, \beta)) = \frac{1}{1 + \frac{1}{2} \left( \frac{y_i - \hat{y}(x_i; \alpha, \beta)}{\sigma_i} \right)^2} = \frac{1}{1 + \frac{1}{2} \delta_i^2(\alpha, \beta)}
\] (25)

which implies:

\[
\eta(\delta_i(\alpha, \beta)) = \ln(1 + \frac{1}{2} \delta_i^2)
\] (26)
\[
\psi(\delta_i) = \frac{d\eta(\delta_i)}{d\delta_i} = \frac{\delta_i}{(1 + \frac{1}{2} \delta_i^2)}
\] (27)

and the merit function takes the form,

\[
\min_{\alpha, \beta} A(\alpha, \beta) = \sum_{i=1}^{N} [\ln(1 + \frac{1}{2} \left( \frac{y_i - \hat{y}(x_i; \alpha, \beta)}{\sigma_i} \right)^2)] = \sum_{i=1}^{N} [\ln(1 + \frac{1}{2} \delta_i^2(\alpha, \beta))]
\] (28)
The various distributions are contrasted in Figure 2.5. The double sided exponential and Lorentzian distributions exhibit a much broader tail compared to the normal distribution. From equations (14) and (15) we note that \( \psi(\delta) \) acts as a weighting factor for the response of basis functions at each data point. This weighting factor grows linearly and without bound for the normal distribution. The same identical weight is assigned to each point for the double sided exponential distribution with only its sign being important. For the Lorentzian distribution, the assigned weight grows initially with increasing deviation \( \delta \) but is reduced back to zero for outliers with very large \( \delta \).

**Figure 2.5** - The probability \( \rho(\delta(\alpha, \beta)) \) and the associated weighting factor \( \psi(\delta(\alpha, \beta)) \).
The influence of various distributions on the fitted model parameters can be illustrated by the simple example of fitting a line,

\[ \hat{y}(x, \alpha_1, \alpha_2) = \alpha_1 + \alpha_2 x \]  

(29)

to the set of noisy data in Figure 2.6. The data was generated by adding random deviates to the line \( \bar{y} = 5x + 2 \). Figure 2.6 illustrate the straight lines fitted using the merit function corresponding to each distribution. It is clear that the line fitted using the least square criterion (Gaussian distribution) is severely distorted by the outliers. The more realistic double sided exponential and Lorentzian distributions perform much better with the latter producing the best result.

Figure 2.6 – Influence of outliers on a line fitted using various distributions.

(\( \alpha_1 = \text{slope}, \ \alpha_2 = \text{intercept} \)
It should be noted, however, that the application of the double sided exponential, Lorentzian and other realistic distributions is accompanied with severe numerical difficulties. This is because the objective

\[ \min_{\alpha, \beta} \quad A(\alpha, \beta) = \sum_{i=1}^{N} \left| \delta_i(\alpha, \beta) \right| \quad \text{(Double sided exponential)} \]  

has discontinuous derivatives with respect to \( \alpha \) and \( \beta \). This rules out the use of general minimisation and non-linear equation solving techniques which demand continuous derivatives. One is forced to use methods which do not rely on derivatives which are far less developed and do not have favourable convergence properties. The objective,

\[ \min_{\alpha, \beta} \quad A(\alpha, \beta) = \sum_{i=1}^{N} \ln(1 + \frac{1}{2} \delta_i^2(\alpha, \beta)) \quad \text{(Lorentzian)} \]  

has continuous derivatives but is almost as difficult. Small changes in \( \alpha \) and \( \beta \) can drive \( \psi(\delta_i) \) off its peaks and into its small asymptotic regions. Consequently, different terms in equations (14) and (15) spring into and out of action making the numerical solution very unstable. In contrast, the objective

\[ \min_{\alpha, \beta} \quad A(\alpha, \beta) = \frac{1}{2} \sum_{i=1}^{N} \delta_i^2(\alpha, \beta) \quad \text{(Gaussian)} \]  

is well behaved with continuous derivatives allowing the use of powerful multidimensional minimisation techniques. It is therefore customary to employ the Gaussian distribution as a first attempt. In the absence of outliers, the results obtained are often indistinguishable from those based on the double sided exponential or Lorentzian distributions. When outliers are identified, the results obtained with the less robust normal distribution provide a good starting point for the more difficult task of robust estimation. In this thesis, we shall mainly concentrate on the normal distribution.
2.4 – Generalisation and over-fitting

The fitted model is expected not only to recall the observed data with the required accuracy but also to produce acceptable predictions for unseen (test) data drawn from the same population as the observed data. Such a model is said to generalise (interpolate) well within the data range. The estimation of the best fit parameters by the naïve minimisation of a merit function based on the fidelity to the data alone can lead to a fitted model with very poor predictive capability. Given a sufficient number of parameters (degrees of freedom) the model can be fitted to pass through each of the observed data points exactly. This strict interpolation is undesirable and must be avoided in practice since the observed data are inevitably corrupted with measurement noise. The severe consequence of this over-fitting phenomenon and the various procedures available for its prevention are easily demonstrated using a simple univariate example. The more complex task of improving generalisation for problems with multivariable inputs will be considered in the next chapter.

In selecting an empirical model to recover the true trend underlying a set of noisy data we must negotiate two problems. First, the basis functions chosen must be compatible with the underlying trend. Second, the data set must be capable of distinguishing between the basis functions chosen. There is no magical cure for using an inappropriate set of basis functions but there are a number of techniques available to remedy the numerical ill-conditioning caused by the data. To illustrate these techniques we consider the noisy data set shown in Figure 2.7. The data was generated by adding random deviates drawn from a normal distribution with a mean of zero and standard deviation of unity to the line
\( \hat{y}(x) = 5x + 2 \) at 100 different \( x_i \) values in the range \( 0 \leq x \leq 1 \). We shall attempt to recover the true linear trend underlying this data set using a collection of Gaussians centred at each data point,

\[
\hat{y}(x, \alpha) = \sum_{j=1}^{N} \alpha_j \exp\left(-\frac{(x-x_j)^2}{\sigma^2}\right)
\]  

(33)

The parameter \( \sigma \) will be pre-specified and controls the width (locality) of the basis functions. We do not expect such a model to perform well and it is only used to illustrate the consequences of over-fitting and ill-conditioning and the various remedies.

For a given \( \sigma \), the optimal linear parameters may be found by minimising the least squares criterion,

\[
A(\alpha) = \frac{1}{2} \sum_{i=1}^{N} \left[ y_i - \sum_{j=1}^{N} \alpha_j \exp\left(-\frac{(x_i-x_j)^2}{\sigma^2}\right) \right]^2
\]  

(34)

Defining the \( N \)-vector of observations \( y = [y_1, \ldots, y_N]^T \) and the \( N \times N \) design matrix \( \Phi \) with elements,

\[
\phi_{ij} = \exp\left(-\frac{(x_i-x_j)^2}{\sigma^2}\right) 
\]  

(35)

the objective can be expressed as:

\[
\min_{\alpha} \quad A(\alpha) = \frac{1}{2} \| y - \Phi \alpha \|^2
\]  

(36)

Applying the optimality condition, \( \partial A(\alpha) / \partial \alpha = 0 \), the optimal linear coefficients must satisfy the system,
\[ (\Phi^T \Phi) \alpha^* = \Phi^T y \]  \hspace{1cm} (37)

The existence and uniqueness of the solution for this system depends on the nature of the matrix \( \Phi \). Assuming for the moment that \( \Phi \) is non-singular, the system (37) reduces to \( \Phi \alpha^* = y \) and the optimal solution is then

\[ \alpha^* = \Phi^{-1} y \]  \hspace{1cm} (38)

and

\[ A(\alpha^*) = \frac{1}{2} \| y - \Phi \Phi^{-1} y \|^2 = 0 \]  \hspace{1cm} (39)

For a non-singular \( \Phi \), the model will therefore interpolate each data point exactly but its generalisation performance across the range \( 0 \leq x \leq 1 \) may be very poor.

It is easy to see that the matrix \( \Phi \) becomes progressively better conditioned as \( \sigma \) is reduced and in the limit of \( \sigma = 0 \) reduces to the identity matrix. The left panel in Figure 2.7 shows the generalisation performance of the fitted model for \( \sigma = 0.01 \), the data points are interpolated exactly but there are wild oscillations in between the data points. Such oscillations are inevitable because the basis functions with \( \sigma = 0.01 \) are highly localised and are incapable of capturing the linear trend underlying the data. To avoid such oscillations we must choose less localised basis functions compatible with the underlying trend that has a global nature.
Figure 2.7 – Poor generalisation performances of the over-fitted models.

\((M=N=100, \, x_j = x_i)\)

For this illustrative model we could for example employ a larger \(\sigma\), say \(\sigma = 0.1\), resulting in less localised Gaussians. The right panel in Figure 2.7 shows the result for \(\sigma = 0.1\) which does not interpolate the data and also shows wild oscillations. This time, however, the wild oscillations are due to the ill-conditioning of the design matrix \(\Phi\) and may be alleviated through a variety of procedures. Quite frequently the available data does not clearly distinguish between two or more of the suitably chosen basis functions. If two such functions or two different combinations of functions fit the data equally well (or equally badly) the columns of the design matrix \(\Phi\) become linearly dependent which causes the matrix \(\Phi^T \Phi\) to become very ill-conditioned and nearly singular. Least squares problems are over-determined in the sense that the number of data points is equal or greater than the number of parameters. They are, however, frequently under-determined in the sense that ambiguous combination of parameters exits. Such ambiguities are very hard to predict \textit{a priori} and the application of standard techniques, such as Gaussian elimination or LU decomposition, will yield a set of very large parameters. The
parameters are very finely balanced to give a good representation at the data points at the expense of wild oscillations elsewhere. A variety of techniques are available to stabilise the solution and improve the generalisation performance. These include singular value decomposition, ridge regression, constrained least squares and linear regularisation methods which will be considered in turn.

2.4.1 – Singular Value Decomposition: ill-conditioned least square problems

In this section we describe a very powerful technique, referred to as singular value decomposition, or SVD for short, which is capable of diagnosing the near linear dependency of the columns of a given matrix. SVD is central to the solution of least squares and affiliated problems and will be used heavily throughout this study. It provides a robust technique for obtaining meaningful solutions to ill-conditioned least square problems and serves as a valuable tool for stabilising solutions and improving generalisation capabilities of a fitted model. We shall also find it useful for improving the computational efficiency of a variety of calculations.

SVD is based on the following remarkable theorem of linear algebra [Golub et al, 1996],

"If $\Phi$ is a real $N \times M$ matrix, then there always exist an orthogonal matrix $U \in \mathbb{R}^{N \times N}$ ($UU^T = U^TU = I_N$) and an orthogonal matrix $V \in \mathbb{R}^{M \times M}$ ($VV^T = V^TV = I_M$) such that,

$$U^T\Phi V = D_\Phi = \text{diag}[s_1, s_2, \ldots, s_q]$$

where $s_1 \geq s_2 \geq \ldots \geq s_q \geq 0$ are the singular values of the matrix $\Phi$."
The proof of this theorem is beyond the scope of this thesis and can be found in [Golub and Van Loan, 1996] who also describe stable and efficient methods for computing the SVD. Computer codes for computing the SVD can be found in Press, et al [1992], The NAG Fortran library [manuel.brad.ac.uk/help/packlangtool.maths/nag.html] and the LAPACK collection of routines [www.netlib.org]. For our applications we have used the Fortran code freely available in LAPACK.

To begin with let us consider the solution of the system (38) for the case $M = N$. Substituting for SVD of $\Phi$ we have:

$$\alpha^* = \Phi^{-1} y = [UD\Phi V^T]^{-1} y \quad (41)$$

In this case all matrices are $N \times N$ square and the inverse is trivial to compute. $U$ and $V$ are orthogonal so their inverses are equal to their transposes and the inverse of the diagonal matrix $D_\Phi$ is the diagonal matrix $D_\Phi^{-1}$ with elements $1/s_j$. The unique solution is therefore,

$$\alpha^* = VD_\Phi^{-1}U^T y = V\text{diag}[1/s_1, 1/s_2, ..., 1/s_N]U^T y \quad (42)$$

with $s_j \neq 0, \ j = 1,2,...,N$. This solution of course breakdown if one or more of the singular values is exactly zero, the matrix $\Phi$ is then singular and its inverse does not exist. The a priori computation of the SVD will clearly diagnose this situation. More significantly, SVD suggests a simple procedure to obtain a useful solution for singular or
highly *ill-conditioned* linear systems. To reveal this procedure we need to consider the concepts of the *range* and *null* spaces and the *rank* and *nullity* of a matrix.

A square $N \times N$ matrix $\Phi$ defines a linear mapping from the $N$-vector space $\alpha$ to another $N$-vector space $\gamma$ through the relationship,

$$\Phi \alpha = \gamma$$ \hspace{1cm} (43)

If $\Phi$ is singular, there is some subspace of $\alpha$, called the nullspace, which is mapped to zero (i.e. $\Phi \alpha = 0$). The dimension of the nullspace, that is the number of linearly independent vectors that can be found in it, is called the *nullity* of $\Phi$. There is also a subspace of $\gamma$ that can be reached by $\Phi$, in the sense that there exists some $\alpha$ which is mapped there. This subspace of $\gamma$ is called the *range* of $\Phi$. The range of a nonsingular matrix $\Phi$ coincides with the whole of the $N$-vector space $\gamma$; the rank is therefore $N$ and the nullspace is empty and the nullity is zero. The range of a singular matrix $\Phi$ is some subspace rather than the whole space of $\gamma$; the rank is then less than $N$ and the nullity is such that "*rank plus nullity equals $N$". The significance of SVD is that it delivers directly the orthonormal bases for the range and nullspaces of the matrix $\Phi$. The columns of the matrix $U$ whose same numbered singular values are *nonzero* are an orthogonal set of basis vectors for the range of $\Phi$. The columns of the matrix $V$ whose same numbered singular values are *zero* are an orthogonal set of basis vectors which span the nullspace.

We can now reconsider the solution of the system (43) for a singular matrix $\Phi$. First we note that the homogeneous system $\Phi \alpha = 0$ is solved immediately by SVD: any
column of the matrix $V$ with a corresponding zero singular value is a solution of the homogeneous system. For a non-zero right hand side $y \neq 0$, the important question whether $y$ lies in the range of the matrix $\Phi$ or not. If it does, the singular system has a solution but this solution is not unique. This is because we can add a multiple of any column of $V$ which has a corresponding zero singular value to construct a new solution. We can isolate amongst all of these non-unique solutions that has the smallest length $\| \tilde{\alpha} \|$ by simply setting $1/s_j = 0$ for all $s_j = 0$ and compute the modified inverse of the diagonal matrix by [Press et al, 1992],

$$
\tilde{D}_\Phi^{-1} = \text{diag}\{1/s_1,1/s_2,...,1/s_N\}, \quad \text{with } 1/s_j = 0 \text{ when } s_j = 0
$$

(44)

To prove that the minimum norm solution is given by,

$$
\tilde{\alpha} = V\tilde{D}_\Phi^{-1}U^T
$$

(45)

we need only to add a vector $\alpha'$ that lies in the null space ($\Phi \alpha' = 0$) to $\tilde{\alpha}$ and consider,

$$
\| \tilde{\alpha} + \alpha' \|^2 = \| V\tilde{D}_\Phi^{-1}U^T y + \alpha' \|^2
$$

$$
= \| V(D^{-1}_\Phi U^T y + V^T \alpha') \|^2
$$

$$
= \| \tilde{D}_\Phi^{-1} U^T y + V^T \alpha' \|^2
$$

(46)

Here the second and third equalities are a consequence of the orthogonality of the matrix $V$. Now the first vector in (46) has a nonzero $j^{th}$ component only when $s_j \neq 0$ whereas the second vector has a nonzero $j^{th}$ component only when $s_j = 0$. It is evident that the minimum norm solution is obtained when $\alpha' = 0$ and is given by (45).
If the right hand side vector $y$ is not in the range of the singular matrix $\Phi$, then the linear system (43) has no solution. We can, however, still use the vector $\alpha$ given by (45) as a very useful approximate solution. This vector does not satisfy the linear system exactly, $\Phi \alpha \neq y$, but provides the vector which gives the smallest possible residuals:

\[ \alpha = \min_{\alpha} \| \Phi \alpha - y \|^2 \]  
(47)

To prove (47) suppose we add an arbitrary vector $\alpha''$ to $\alpha$, then letting $y'' = \Phi \alpha''$ we have:

\[
\| \Phi \alpha - y + y'' \|^2 = \| U \tilde{D}_\Phi V^T (V \tilde{D}_\Phi^{-1} U^T y) - y + y'' \|^2 \\
= \| (U \tilde{D}_\Phi \tilde{D}_\Phi^{-1} U^T - I) y + y'' \|^2 \\
= \| U (\tilde{D}_\Phi \tilde{D}_\Phi^{-1} - I) U^T y + y'' \|^2 \\
= \| (\tilde{D}_\Phi \tilde{D}_\Phi^{-1} - I) U^T y + U^T y'' \|^2
\]  
(48)

Now the diagonal matrix $(\tilde{D}_\Phi \tilde{D}_\Phi^{-1} - I)$ has a non-zero $j^{th}$ component only when $s_j = 0$ whereas the vector $U^T y''$ has a nonzero $j^{th}$ component only when $s_j \neq 0$. It is evident therefore that the minimum norm is obtained for $y'' = 0$ and the $\alpha$ vector given by (45).

We have so far taken an analytic viewpoint and assumed that the matrix $\Phi$ is either singular, has one or more exactly zero singular value, or else it is non-singular. Numerically, however, problems can arise if one or more of the singular values is not exactly zero but is so small that it is dominated by the round off error and is therefore unknowable. The matrix is then analytically non-singular but such small singular values
will result in an $\alpha^*$ vector with very large components. These large components are very finely balanced to give accurate prediction at the data points but cause wild oscillations at other values of the independent variable. More generally, the condition number of a square matrix is given by the ratio of its largest and smallest singular values. Problem can be expected when the reciprocal of the condition number approaches the machine's floating point precision (for example $10^{-12}$ for double precision calculations). In such situations the vector $\bar{\alpha}$, obtained but setting the small singular values to zero and then using (45), often provides a much better generalisation performance than the $\alpha^*$ given directly by (42). This is because zeroing a small singular value is equivalent to throwing away a combination of the set of equations, which is very likely linearly dependent on others and therefore carries no useful information. Leaving in such a combination will only serve to compound the round off error and pulls the solution towards infinity.

Figure 2. $\mathcal{G}$ illustrates the generalisation performances obtained by SVD for various thresholds of $\theta = 10^{-10}, \theta = 10^{-6}$ and $\theta = 10^{-2}$. The superior performance of SVD and its stabilising power for curing the degeneracy of the design matrix $\Phi$ for the case of $\sigma = 0.1$ is self-evident. The major difficulty in using SVD is in deciding on the appropriate threshold which is highly problem dependent.
Figure 2.8 - Generalisation performances obtained by SVD using various thresholds ($\theta$).

(----- True trend, ______ Predictions)
2.4.2 - Ridge Regression

In a more direct approach for improving the generalisation capability of the fitted model we may attempt to solve the constrained minimisation of [Golub et al, 1996]

\[
\min_{\alpha} \quad A(\alpha) = \frac{1}{2} \| y - \Phi \alpha \|^2 \quad \text{subject to} \quad \| \alpha \|^2 = \nu^2
\]

(49)

where \( \nu \) is a constant chosen to limit the norm of the solution \( \alpha \) and damp out the excessive oscillations in the fitted function. To solve this problem we use the undetermined Lagrange multiplier (ridge parameter) \( \lambda \) to append the zero quantity \( (\| \alpha \|^2 - \nu^2) \) to the objective and consider instead the minimisation of:

\[
\min_{\alpha} \quad \mathcal{J}(\alpha) = \frac{1}{2} \| y - \Phi \alpha \|^2 + \frac{1}{2} \lambda \| \alpha \|^2 - \frac{1}{2} \nu^2
\]

(50)

Applying the optimality condition \( \partial \mathcal{J}(\alpha) / \partial \alpha = 0 \) shows that for a given value of \( \lambda \) the unique minimiser \( \alpha_\lambda \) of (50) must satisfy the system,

\[
\left( \Phi^T \Phi + \lambda I \right) \alpha_\lambda = \Phi^T y
\]

(51)

and is given by

\[
\alpha_\lambda = \left( \Phi^T \Phi + \lambda I \right)^{-1} \Phi^T y
\]

(52)

The solution (52) could of course violate the ridge constraint \( \| \alpha \|^2 = \nu^2 \) and we seek that value \( \lambda^* \) which satisfies the constraint exactly. The determination of \( \lambda^* \) is greatly simplified by substituting the SVD of \( \Phi = U \Sigma V^T \) and using the orthogonality of \( U \) and \( V \) to reduce the system (51) to,

\[
\left( D_{\Phi}^T D_{\Phi} + \lambda I \right) (V^T \alpha_\lambda) = D_{\Phi}^T U^T y
\]

(53)
Defining the vector $\hat{y} = U^T y$, the solution to the above system for a given $\lambda$ is,

$$(V^T \alpha_{\lambda}) = (D_{\Phi}^T D_{\Phi} + \lambda I)^{-1} D_{\Phi}^T \hat{y} \tag{54}$$

Noting that the ridge constraint may be taken as $\| \alpha_{\lambda} \|_2^2 = \| V^T \alpha_{\lambda} \|_2^2 = \nu^2$ and substituting for $V^T \alpha_{\lambda}$ from (54) leads to the following algebraic equation in $\lambda$:

$$\mathfrak{A}(\lambda) = -\nu^2 + \hat{y}^T D_{\Phi} (D_{\Phi}^T D_{\Phi} + \lambda I)^{-T} (D_{\Phi}^T D_{\Phi} + \lambda I)^{-1} D_{\Phi}^T \hat{y}$$

$$= \sum_{i=1}^{M} \left( \frac{s_i \hat{y}_i}{s_i^2 + \lambda} \right)^2 - \nu^2 = 0 \tag{55}$$

where the $s_i$ are the singular values of the design matrix $\Phi$. We note immediately that $\mathfrak{A}(\lambda)$ is a monotone decreasing function of $\lambda$. Furthermore, at $\lambda = 0$,

$$\mathfrak{A}(\lambda = 0) = \sum_{i=1}^{M} \left( \frac{\hat{y}_i}{s_i} \right)^2 - \nu^2 \tag{56}$$

The ridge regression problem has a unique solution $\lambda^*$ if and only if $\mathfrak{A}(\lambda = 0) > 0$ which can be found through any univariate root finding technique. The optimal parameters for the ridge regression problem are therefore given by,

$$\alpha_{\lambda^*} = V(D_{\Phi}^T D_{\Phi} + \lambda^* I)^{-1} D_{\Phi}^T U^T y \tag{57}$$

with the $\lambda^*$ selected as the unique root of equation (55). The major computational burden is therefore reduced to the computation of the SVD of the design matrix $\Phi$. Evaluating the SVD will also indicate if a solution does not exist for the ridge constant $\nu$ specified.
Figure 2.9 illustrates the generalisation performance of the model fitted using ridge regression with various values of $\nu = 10^6$, $10^3$, 10 and 1. Increasing the parameter $\lambda$ decreases the norm of the model parameters which results in milder oscillations. Indeed, a sufficiently large value of $\lambda$ damps the solution completely and forces the model predictions to zero across the range $0 \leq x \leq 1$. As with the threshold value in the SVD approach, the major difficulty is on deciding what constitutes a suitable ridge parameter $\lambda$ which is highly problem dependent. A more general approach known as “constrained least squares” will be considered next.

\begin{figure}[h]
\centering
\begin{subfigure}{0.4\textwidth}
  \centering
  \includegraphics[width=\textwidth]{figure1}
  \caption*{$\sigma = 0.1$}
  \end{subfigure}
\begin{subfigure}{0.4\textwidth}
  \centering
  \includegraphics[width=\textwidth]{figure2}
  \caption*{$\sigma = 0.1$}
  \end{subfigure}
\begin{subfigure}{0.4\textwidth}
  \centering
  \includegraphics[width=\textwidth]{figure3}
  \caption*{$\sigma = 0.1$}
  \end{subfigure}
\begin{subfigure}{0.4\textwidth}
  \centering
  \includegraphics[width=\textwidth]{figure4}
  \caption*{$\sigma = 0.1$}
  \end{subfigure}
\caption{Generalisation performances obtained by ridge regression using various ridge constants $\nu^2$. (------- True trend, \quad Predictions)}
\end{figure}
2.4.3 - Constrained Least Squares: Generalised singular value decomposition

In a more general approach for stabilising the solution and improving the generalisation performance of the fitted model we may consider a constrained least square minimisation problem. Namely, we seek a solution to,

$$\min_{\alpha} A(\alpha) = \frac{1}{2} (y - \Phi \alpha)^T (y - \Phi \alpha)$$

subject to some quadratic inequality constraint,

$$\| \Psi \alpha - d \|^2 \leq \gamma^2$$

In this formulation $\Phi$ is as before the $N \times M$ design matrix, $\Psi$ is a given $P \times M$ constraint matrix and $d$ and $\gamma$ are a given $P$-vector and a given scalar quantity respectively. The inequality constraint (59) is quite general and reduces to specialised forms on choosing the matrix $\Psi$ and the vector $d$. For example, choosing $P = M$, $\Psi = I_M$ and $d = 0$ reduces the constrained minimisation to ridge regression. We shall highlight the solution for the general constraint (59) and consider special cases afterwards.

The solution of equations (58) and (59) is greatly facilitated by resorting to a generalisation of the singular value decomposition (or GSVD for short) contained in the following theorem of linear algebra [Golub and Van Loan, 1996]:

"For any given $\Phi \in \mathbb{R}^{N \times M}$ with $N \geq M$ and $\Psi \in \mathbb{R}^{P \times M}$ there exist an orthogonal matrix $U \in \mathbb{R}^{N \times N}$, an orthogonal matrix $V \in \mathbb{R}^{P \times P}$ and a non-singular matrix $X \in \mathbb{R}^{M \times M}$ such that,
\[ U^T \Phi X = C = \text{diag}[c_1, c_2, \ldots, c_M], \quad c_i \geq 0 \]  
(60)

\[ V^T \Omega X = S = \text{diag}[s_1, s_2, \ldots, s_q], \quad s_j \geq 0 \]  
(61)

and \( q = \min\{P, M\} \).

The proof of this theorem is beyond the scope of this thesis and appears in Golub and Van Loan [1996] who also discuss efficient and stable methods for constructing the generalised singular value decomposition. Computer code for performing the GSVD is freely available from the LAPACK collection of routines [www.netlib.org] and was used for our calculations.

Substituting the GSVD of \( \Phi = UCX^{-1} \) and \( \Psi = VSX^{-1} \) in (58) and (59) respectively and using the orthogonality of \( U \) and \( V \) we arrive at,

\[
\begin{align*}
\text{min.} & \quad A(\alpha) = \| \bar{y} - UCX^{-1}\alpha \|^2 \\
\alpha & \quad = \| U(U^T \bar{y} - CX^{-1}\alpha) \|^2 \\
& \quad = \| U^T \bar{y} - CX^{-1}\alpha \|^2 \\
\end{align*}
\]

subject to:

\[
\| \Psi \bar{a} - \bar{d} \|^2 = \| VSX^{-1}\alpha - \bar{d} \|^2 \\
= \| V(SX^{-1}\alpha - V^T \bar{d}) \|^2 \\
= \| SX^{-1}\alpha - V^T \bar{d} \|^2 \\
\]

Defining the vectors \( \bar{\alpha} = X^{-1}\alpha \), \( \bar{\alpha} = U^T \bar{y} \) and \( \bar{d} = V^T \bar{d} \) we can write (62) and (63) as a minimisation over \( \bar{\alpha} \),
\[
\begin{align*}
\min_{\bar{\alpha}} \quad & A(\bar{\alpha}) = \| \bar{y} - C\bar{\alpha} \|^2 \\
\text{subject to,} \quad & \| S\bar{\alpha} - \bar{d} \|^2 \leq \gamma^2
\end{align*}
\] (64)

Once the minimising \( \bar{\alpha} \) has been found we can recover the solution to the original problem from \( \bar{\alpha} = X\hat{\alpha} \). The simple form of (64) and (65) which involve the diagonal matrices \( C \) and \( S \) facilitates the solution greatly. The objective (64) has the simple form,

\[
\begin{align*}
\min_{\bar{\alpha}} \quad & A(\bar{\alpha}) = \sum_{i=1}^{M} (\bar{y}_i - c_i\bar{\alpha}_i)^2 + \sum_{j=M+1}^{N} \bar{y}_j^2 \\
\end{align*}
\] (66)

and the constraint equation can be written as,

\[
\| S\bar{\alpha} - \bar{d} \|^2 = \left[ \sum_{i=1}^{r} (s_i\bar{\alpha}_i - d_i)^2 + \sum_{j=r+1}^{p} \bar{d}_j^2 \right] \leq \gamma^2
\] (67)

Here \( r \) represents the rank of matrix \( \Psi \) and we assume that \( s_{r+1} = s_{r+2} = \ldots = s_q = 0 \).

We note immediately that the problem (66) and (67) has a solution if and only if,

\[
\sum_{j=r+1}^{p} d_j^2 \leq \gamma^2
\] (68)

If the equality holds in (68) we see immediately from (66) and (67) that the vector defined as,

\[
\bar{\alpha}_i = \begin{cases} 
\frac{d_i}{s_i} & i = 1, 2, \ldots, r \\
\frac{\bar{y}_i}{c_i} & i = r + 1, r + 2, \ldots, N; \text{ and } c_i \neq 0 \\
0 & i = r + 1, r + 2, \ldots, N; \text{ and } c_i = 0
\end{cases}
\] (69)

solves the constrained minimisation problem of (66) and (67). For the case where,
we may have a large number of solutions and would like to isolate that with the minimum norm. From (66) it is clear that the vector

\[
\vec{a}_i = \begin{cases} 
\vec{y}_i & c_i \neq 0 \\
\vec{c}_i & c_i = 0
\end{cases}
\]

(71)

is a minimiser of the objective (66). If the vector (71) also satisfies the constraint (67) it is an admissible solution but may not be the one with the minimum norm. To obtain the minimum norm solution we assume that the vector (71) is inadmissible and violates the constraint (67). In other words, we assume that

\[
\sum_{i=1}^{q} \left[ \frac{\vec{y}_i}{c_i} \right]^2 \geq \sum_{j=r+1}^{p} \frac{\vec{d}_j}{c_i}^2 = \gamma^2
\]

This in turn implies that the minimum norm solution of (66) and (67) occurs at the boundary of the feasible solution set. This can be found by considering the equality constrained least square problem,

\[
\min_{\vec{a}} A(\vec{a}) = \| \vec{y} - C\vec{a} \|^2
\]

subject to,

\[
\| S\vec{a} - \vec{d} \|^2 = \gamma^2
\]

(73)

To solve this problem we use the undetermined Lagrange multiplier \( \lambda \) to append the zero quantity \( \| S\vec{a} - \vec{d} \|^2 - \gamma^2 \) to the objective (64):
\[ \mathcal{I} (\lambda, \overline{\alpha}) = \| \overline{y} - C \overline{\alpha} \| ^2 + \lambda \left( \| S \overline{\alpha} - \overline{d} \| ^2 - \gamma^2 \right) \]  

(74)

For a fixed value of \( \lambda \) and applying the optimality condition \( \frac{\partial \mathcal{I} (\overline{\alpha})}{\partial \overline{\alpha}} = 0 \), the minimiser \( \overline{\alpha}(\lambda) \) of (74) is obtained as the solution of the linear system:

\[ \left[ C^T C + \lambda S^T S \right] \overline{\alpha} = C^T \overline{y} + \lambda S^T \overline{d} \]  

(75)

This admits the solution,

\[ \overline{\alpha} = \begin{cases} 
\frac{c_i \overline{y}_i + \lambda s_i \overline{d}_i}{c_i^2 + \lambda s_i^2} & i = 1, 2, ..., q \\
\frac{\overline{y}_i}{c_i} & i = q + 1, ..., N 
\end{cases} \]  

(76)

The final task is to evaluate the Lagrange multiplier \( \lambda \) such that the constraint (73) is fulfilled exactly. Substituting the solution (76) into the constraint (73) yields the algebraic equation.

\[ \sum_{i=1}^{q} \frac{c_i \overline{s}_i \overline{y}_i - c_i \overline{d}_i}{c_i^2 + \lambda s_i^2} \gamma^2 + \sum_{j=q+1}^{N} \overline{d}_j^2 - \gamma^2 = 0 \]  

(77)

We observe that \( \hat{\mathcal{I}} (\lambda) \) is a monotone decreasing function of \( \lambda \). Furthermore, from (72) we may conclude that \( \hat{\mathcal{I}} (\lambda = 0) > 0 \). It follows that (77) has a unique positive solution \( \lambda^* \) which ensures the equality constraint (73) is satisfied exactly. The solution \( \lambda^* \) can easily be found using any univariate root finding technique. The solution to the original problem is obtained by setting \( \overline{\alpha}^* = X \overline{\alpha}^* \). The next section presents a particular choice for the constraint matrix \( \Psi \) which imposes certain smoothness features on the generalisation performance of the fitted model.
2.4.4 – Penalised Least squares: Smoothness constraint

As a special case of an inequality constrained least square problem we may easily impose smoothness characteristics on the generalisation performance of the fitted model using the GSVD technique. For example, if we wish that the response is not highly oscillatory across the range of the independent variable we may require that,

\[ \int_a^b \left( \frac{d^2 \hat{y}}{dx^2} \right)^2 \, dx \leq \gamma^2 \tag{78} \]

Noting that the estimate of the true response is given by,

\[ \hat{y}(x) = \sum_{j=1}^{M} \alpha_j \phi_j(x) \tag{79} \]

we have:

\[ \left( \frac{d^2 \hat{y}}{dx^2} \right) = \sum_{j=1}^{M} \alpha_j \left( \frac{d^2 \phi_j}{dx^2} \right) \tag{80} \]

Substituting for \( \left( \frac{d^2 \hat{y}}{dx^2} \right) \) enables us to write the constraint (78) as,

\[ \alpha^T \Omega \alpha \leq \gamma^2 \tag{81} \]

where \( \Omega \) is an \( M \times M \) matrix with elements,

\[ \Omega_{ij} = \int_a^b \left( \frac{d^2 \phi_i}{dx^2} \right) \left( \frac{d^2 \phi_j}{dx^2} \right) \, dx \tag{82} \]

which can be precomputed by suitable quadrature. The constraint (81) can be cast into the form required by GSVD procedure using the similarity transformation of the matrix \( \Omega \),

\[ \Omega = Q \Lambda Q^T \]

\[ = (\sqrt{\Lambda} Q^T)^T (\sqrt{\Lambda} Q^T) \tag{83} \]
where \( Q \) is the orthogonal matrix of eigenvectors and \( \Lambda \) is the diagonal matrix of eigenvalues. We can immediately identify the constraint matrix \( \Psi \) as,

\[
\Psi = \Lambda^{\frac{1}{2}} Q^T
\]

and write the constraint (81) in the standard form,

\[
\| \Psi \alpha \|^2 \leq \gamma^2
\]

The original problem is then reduced to a form suitable for applying the GSVD procedure:

\[
\min_{\alpha} \quad \Lambda(\alpha) = \| y - \Phi \alpha \|^2
\]

subject to,

\[
\| \Psi \alpha \|^2 \leq \gamma^2
\]

with the constraint matrix \( \Psi \) given by (82) to (84).

Figure 2.10 shows the results obtained for various values of \( \gamma^2 \) and the stabilising influence of the smoothness constraint is clearly apparent. It is worth pointing out that the GSVD based procedure with a suitably small \( \gamma \) actually recovers the true underlying trend using a collection of Gaussians. This could not be achieved through the direct application of SVD or ridge regression which could alleviate the problem but did not recover a line (see Figures 2.8 and 2.9). Once again the major difficulty is in deciding on an appropriate value of \( \gamma^2 \) which is highly problem dependent.
Figure 2.10 – Generalisation performances obtained by penalised least square for various values of $\gamma^2$. (-------- True trend, ______ Predictions)
2.5 - Univariate Linear Regularisation Technique

The SVD and GSVD based procedures described above share a number of common features. First, we need to assume an *a priori* known set of basis functions and compute the response as a linear combination of these basis functions. Evidently, the choice of basis functions has an impact on the quality of the fitted response. In our illustrative example we deliberately chose to represent a true linear trend with a collection of localised Gaussian basis functions. The locality of the basis functions could be easily controlled by specifying a spread variable $\sigma$ for the Gaussians. The techniques described will only work if $\sigma$ is not too small and will fail badly if the basis functions are highly localised. The stabilising techniques described so far can alleviate the ill-conditioning problem but can not overcome an inherently inappropriate choice of basis functions. Second, for each technique we must decide on a highly problem dependent constant, the threshold in the direct application of SVD, the constant $\nu^2$ in ridge regression and the constant $\gamma^2$ in penalised least squares using GSVD. Third, the techniques described are also limited to univariate problems and can not be easily extended to problems with multiple independent variables. In this section we describe a more general approach, the linear regularisation technique, which overcomes such difficulties. In particular, we shall not need to assume a specific *a priori* model for the response function and can also develop efficient techniques for selecting an appropriate level of stabilisation based on the noisy data set alone. More significantly, the linear regularisation technique can be naturally extended to multivariate problems considered in the next chapter.

Consider the problem of finding an unknown and underlying function $u(x)$ from a set of noisy exemplars $(x_i, y_i; i = 1, 2, ..., N)$,

$$y_i = \int r_i(x) \, u(x) \, dx + \varepsilon_i \quad (88)$$

The relationship between $u(x)$ and each measured outputs $y_i$'s, is defined by its own linear response kernel $r_i(x)$ and $\varepsilon_i$ is the measurement error associated with the $i^{th}$ experiment. Equation (88) provides a general formulation within the assumption of linearity. The kernel $r_i(x)$ may represent a narrow instrumental response in which case $y_i$ is an approximation to $u(x_i)$. Alternatively, the measured responses ($y_i$'s) might “live” in an entirely different function space from $u(x)$, for example measuring different Fourier components of $u(x)$. Given the $y_i$'s, the kernels, $r_i(x)$'s and perhaps some information about the measurement errors $\varepsilon_i$'s (e.g. their covariance matrix $\Sigma_{ij} = Cov[\varepsilon_i, \varepsilon_j]$), the problem is to devise a procedure to find a good statistical estimator of $u(x)$ which will be denoted as $\hat{u}(x)$. This is an inherently ill-posed inverse problem. Depending on the smoothness of the kernel $r_i(x)$, sharp variations in the underlying function $u(x)$ are smoothed out by the integration. Conversely, small
variations in the data, $y_i$'s, may correspond to large variations in $\tilde{u}(x)$. The problem is further compounded by the presence of noise in the data.

In practice, we are not interested in every point of the continuous function $\hat{u}(x)$ and a large number $M$ of *evenly spaced* discrete points $x_j, j=1,2,...,M$ will suffice. Evidently, $M$ must be large enough so that neither $\hat{u}(x)$ nor $r_i(x)$ varies significantly between two successive $x_j$'s. For a "sufficiently" dense set of $x_j$'s, we may replace the integral in (89) with the following sum using simple quadrature:

$$y_i = \sum_{j=1}^{M} R_{ij} \hat{u}(x_j) + \varepsilon_i$$

(89)

where,

$$R_{ij} = r_i(x_j)(x_{j+1} - x_j)$$

(90)

As a first obvious, but not very sound attempt, we may require that the discrete estimator $\hat{u}(x)$ should minimise the merit function,

$$\chi^2 = \sum_{i=1}^{N} \sum_{k=1}^{N} \left[ y_i - \sum_{j=1}^{M} R_{ij} \hat{u}(x_j) \right] \sum_{ik}^{-1} \left[ y_k - \sum_{j=1}^{M} R_{kj} \hat{u}(x_j) \right]$$

$$\min. \quad \hat{u} = \sum_{i=1}^{N} \left[ \frac{y_i - \sum_{j=1}^{M} R_{ij} \hat{u}(x_j)}{\sigma_{ii}} \right]^2$$

(91)

Here, $\Sigma$ is the covariance matrix of the measurement errors and the approximate equality holds if the cross correlation between the measurement errors is negligible. If we further assume that the measurement errors all have the same standard deviation, (91) reduces to the simple least squares objective,
\[
\min \frac{\mathbf{A}(\hat{\mathbf{u}})}{\hat{\mathbf{u}}} = \sum_{i=1}^{N} \left[ y_i - \sum_{j=1}^{M} R_{ij} \hat{u}(x_j) \right]^2
\]

and can be expressed in the matrix form as,
\[
\min \frac{\mathbf{A}(\hat{\mathbf{u}})}{\hat{\mathbf{u}}} = (\mathbf{y} - \mathbf{R}\hat{\mathbf{u}})^T(\mathbf{y} - \mathbf{R}\hat{\mathbf{u}})
\]

where \( \mathbf{y} \) and \( \hat{\mathbf{u}} \) are vectors of size \( N \) and \( M \) respectively and the elements of the \( N \times M \) matrix \( \mathbf{R} \) are defined by (90). Using the optimality condition \( \frac{\partial \mathbf{A}}{\partial \hat{\mathbf{u}}} = 0 \), the minimiser of (93) is the solution of the following set of linear equations,
\[
(\mathbf{R}^T \mathbf{R}) \hat{\mathbf{u}} = \mathbf{R}^T \mathbf{y}
\]

The direct solution of (94) is hopeless and should be avoided. Since \( M \) is greater than \( N \) (in general \( M \gg N \)) the \( M \times M \) matrix \( \mathbf{R}^T \mathbf{R} \) will be singular and equation (94) will have a large number of highly degenerate solutions. We could of course use the SVD technique described in section 2.4.1 to select the solution with minimum norm from the infinity of possible solutions. It is evident, however, that the SVD process will find a large number of zero singular values. It is also clear that with the number of unknown \( (M) \) being much larger than the number of equations \( (N) \), there is sufficient flexibility in \( \hat{\mathbf{u}} \) to make the objective function (93) unrealistically small.

In a more direct approach, the ill-posed inverse problem can be stabilised by imposing some a priori information (or belief) about the unknown underlying function \( \tilde{u}(x) \) as a constraint on the least square merit function (93). Suppose that while minimising equation (93), we also demand that a positive functional \( B(\tilde{u}) \) (chosen to represent some a priori information about \( \tilde{u}(x) \)) has some particular value \( \kappa \). In this
In this case, the unknown vector \( \hat{u}(x) \) would be the solution of the following constrained minimisation problem,

\[
\begin{align*}
\min_{\hat{u}} \quad & A(\hat{u}) = (y - Ru)^T (y - Ru) \\
\text{Subject to:} \quad & B(\hat{u}) = \kappa
\end{align*}
\]

(Eq. 95)

Using a Lagrange multiplier \( \lambda \) to add the zero quantity \( (B(\hat{u}) - \kappa) \) to (95) and considering the variation of the augmented objective at the solution,

\[
\frac{\partial}{\partial \hat{u}} (A(\hat{u}) + \lambda(B(\hat{u}) - \kappa)) = \frac{\partial}{\partial \hat{u}} (A(\hat{u}) + \lambda B(\hat{u})) = 0
\]

(Eq. 97)

yields a one parameter family of solutions, say \( \hat{u}_\lambda \). As \( \lambda \) varies from zero to infinity the solution \( \hat{u}_\lambda \) varies smoothly from minimising \( A(\hat{u}) \) to minimising \( B(\hat{u}) \). At a given value of \( \lambda \), the solution \( \hat{u}_\lambda \) minimises the weighted sum of two positive functionals,

\[
\min_{\hat{u}} \quad \mathcal{R}(\hat{u}) = A(\hat{u}) + \lambda B(\hat{u})
\]

(Eq. 98)

Another essential requirement for adding \( B(\hat{u}) \) is to cure the inherent degeneracy of the matrix \( R^T R \). Let us assume for the moment that \( B(\hat{u}) \) can be represented as,

\[
B(\hat{u}) = \hat{u}^T \Omega \hat{u}
\]

(Eq. 99)

with \( \Omega \) an \( M \times M \) positive semi-definite matrix. The problem is thus reduced

\[
\min_{\hat{u}} \quad \mathcal{X}(\hat{u}) = \frac{1}{2} (y - Ru)^T (y - Ru) + \frac{1}{2} \lambda \hat{u}^T \Omega \hat{u}
\]

(Eq. 100)

For a fixed \( \lambda \), the optimality condition \( \frac{\partial \mathcal{X}}{\partial \hat{u}} = 0 \) leads to the system,

\[
(R^T R + \lambda \Omega) \hat{u} = R^T y
\]

(Eq. 101)
Assuming $\lambda$ and $\Omega$ are chosen such that $(R^T R + \lambda \Omega)$ is non-singular, the above system admits a unique solution $\tilde{u}_A$. We can now consider a number of frequently used options for selecting the matrix $\Omega$ and hence the regularising function $B(\tilde{u})$.

The degeneracy of the matrix $R^T R$ leads to a least square solution $\tilde{u}$ with very large elements with alternating signs. To prevent these large oscillatory components we may decide to limit the norm of the solution vector $\tilde{u}$ by adding the functional

$$B(\tilde{u}) = \int [\tilde{u}(x)]^2 dx$$

or its equivalent discrete form,

$$B(\tilde{u}) = \tilde{u}^T \tilde{u}$$

For this choice, which is referred to as zero order regularisation, the matrix $\Omega = I_M$ and is strictly positive definite. The system (101) therefore reduces to

$$(R^T R + \lambda I) \tilde{u}_A = R^T \gamma$$

For a sufficiently large $\lambda$, this system has a unique solution which is easily found either by SVD methods or by any other standard technique such as Gaussian elimination or LU decomposition. Increasing the regularisation parameter $\lambda$ pulls the solution away from minimising $A(\tilde{u})$ in favour of minimising the norm of the final solution $\tilde{u}^T \tilde{u}$. 
As an alternative first order regularisation approach, suppose our \textit{a priori} belief is that \( \tilde{u}(x) \) is not too different from a constant. In this case, the regularisation term \( B(\tilde{u}(x)) \) may be defined as,

\[
B(\tilde{u}(x)) = \left[ \frac{d\tilde{u}(x)}{dx} \right]^2 dx
\]  

Discretisationing the integral and employing a forward difference to approximate

\[
\left( \frac{d\tilde{u}(x)}{dx} \right)
\]

leads to the discrete form,

\[
B(\hat{u}) = \sum_{j=1}^{M} (\hat{u}_{j+1} - \hat{u}_j)^2
= \hat{u}^T (\Psi^T \Psi) \hat{u}
= \hat{u}^T \Omega \hat{u}
\]

where, \( \Psi \) is the \((M-1)\times M\) matrix given by,

\[
\Psi = \begin{bmatrix} -1 & 1 & 0 & 0 & 0 \\
0 & -1 & 1 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & -1 & 1 \end{bmatrix}
\]  

We note here that the \((M-1)\times M\) difference matrix \( \Psi \) has one fewer row than column. Consequently, the \(M\times M\) matrix \( \Omega = \Psi^T \Psi \) is degenerate and has one zero eigenvalue (for the unknown constant). We hope that data contains sufficient information to pick up the appropriate value of the constant. In this case, the solution is given by the system,

\[
(R^T R + \lambda \Psi^T \Psi) \hat{u}_\lambda = R^T \hat{y}
\]

with \( \Psi \) given by (107). Increasing \( \lambda \) pulls the solution \( \hat{u}_\lambda \) away from fitting the data towards a constant solution. For sufficiently large \( \lambda \), the solution can again can be found.
by any standard technique for solving systems of linear equations. We note, however, that the structure of (108) is ideal for solution by the GSVD technique. This will allow us to develop very efficient techniques for the automatic selection of the level of regularisation and will be discussed in section 2.6.

Next suppose that our \textit{a priori} belief is that the underlying function resembles a line and is not highly oscillatory. We may then take $B(\hat{u}(x))$ as,

$$B(\hat{u}(x)) = \int \left[ \frac{d^2 \hat{u}(x)}{dx^2} \right]^2 dx$$

(109)

The discrete version is found by replacing the integral with a sum and using forward differences to approximate $\left( \frac{d^2 \hat{u}(x)}{dx^2} \right)$ which leads to,

$$B(\hat{u}) = \hat{u}^T \Psi^T \Psi \hat{u}$$

$$= \hat{u}^T \Omega \hat{u}$$

(110)

with the $(M - 2) \times M$ matrix $\Psi$ is given by,

$$\Psi = \begin{bmatrix}
-1 & 2 & 1 & 0 & 0 & 0 \\
0 & -1 & 2 & 1 & 0 & 0 \\
: & : & : & : & : & : \\
0 & 0 & 0 & -1 & 2 & 1
\end{bmatrix}$$

(111)

In this case the matrix $\Psi$ has two rows fewer than columns. Consequently, the $M \times M$ matrix $\Omega$ has two zero eigenvalues and we hope that the data can provide the appropriate values for the unknown slope and intercept. The solution for the second order regularisation technique can be found exactly the same way as the first order case but with $\Psi$ given by (111).
The above approach can be easily extended to higher orders. For example, for an eighth order regularisation procedure the \((M-8) \times M\) matrix \(\Psi\) is given by,

\[
\Psi = \begin{bmatrix}
-1 & 8 & -28 & 56 & -70 & 56 & -28 & 8 & -1 & 0 & \cdots & 0 \\
0 & -1 & 8 & -28 & 56 & -70 & 56 & -28 & 8 & -1 & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & \cdots & -1 & 8 & -28 & 56 & -70 & 56 & -28 & 8 & -1 & 0 \\
0 & \cdots & 0 & -1 & 8 & -28 & 56 & -70 & 56 & -28 & 8 & -1 \\
\end{bmatrix}
\]

(112)

and \(\Omega = \Psi^T \Psi\) has eight zero eigenvalues. We would then hope that the data has sufficient information to pick up eight unknown constants. Furthermore, the linear regularisation method is not restricted to the specific examples cited and provides a very general approach. For example, if we know that the true underlying process satisfies the following linear differential equation.

\[
a_k(x) \frac{d^k \tilde{u}(x)}{dx^k} + a_{k-1}(x) \frac{d^{k-1} \tilde{u}(x)}{dx^{k-1}} + \cdots + a_1(x) \frac{\partial \tilde{u}(x)}{\partial x} + a_0(x) \tilde{u}(x) = h(x)
\]

(113)

we may consider a regularising functional specified as,

\[
B(\tilde{u}(x)) = \int \left[ g(\tilde{u}(x)) \right]^2 dx
\]

(114)

with

\[
g(\tilde{u}(x)) = a_k(x) \frac{d^k \tilde{u}(x)}{dx^k} + a_{k-1}(x) \frac{d^{k-1} \tilde{u}(x)}{dx^{k-1}} + \cdots + a_1(x) \frac{\partial \tilde{u}(x)}{\partial x} + a_0(x) \tilde{u}(x)
\]

(115)

The relevant \(\Psi\) matrix can then be found by discretising the integral and approximating each differential by an appropriate finite difference formula.

We are free to choose the regularisation functional in any form we wish but for a successful application it must be consistent with the true underlying trend. For example, imposing a high order regularisation constraint when the data does not carry sufficient
information to pick-out the unknown constants will give poor results. Conversely, imposing a low order regularisation when the underlying trend does not resemble a constant or a line will again produce poor results. These issues are clearly demonstrated by the following example drawn from the area of adsorption on heterogeneous solids.

2.5.1 - An Illustrative Example: Adsorption on a Heterogeneous Surface

Conventional treatment of energetic heterogeneity of a solid surface is based upon the concept of localised adsorption on independent sites with a spectrum of adsorption energies. To keep the analysis simple we shall assume that there is no lateral interaction between the molecules adsorbed on neighbouring sites. In the absence of lateral interactions the spatial distribution of sites on the surface is unimportant which simplifies the analysis greatly. The practically measurable quantity is the total adsorption which is simply a summation of the adsorption on the various sites. The local adsorption on particular sites of a given energy is not open to direct measurement and must be specified based on suitable theoretical assumptions. In the absence of lateral interactions, we may represent the fractional adsorption \( \theta_l(p_i, e_j) \) on sites with energy \( e_j \) by the well known Langmuir isotherm:

\[
\theta_l(p_i, e_j) = \frac{d_l(p_i, e_j)}{m_j} = \frac{k_0 \exp(e_j / RT) p_i}{1 + k_0 \exp(e_j / RT) p_i}
\]

(116)

where \( p_i \) is the bulk phase pressure, \( d_l(p_i, e_j) \) is the number of molecules adsorbed, \( m_j \) is the number of sites with energy \( e_j \), and \( k_0 \) is a constant.
The total adsorption on all sites is therefore, 

\[ a(p_i) = \sum_j a_j (p_i, e_j) = \sum_j m_j \frac{k_0 \exp(e_j / RT)p_i}{1 + k_0 \exp(e_j / RT)p_i} \]  

(117)

For a solid surface with a continuous spectrum of site energies we may replace the above summation with an integral, 

\[ a(p_i) = \int_{e_{\text{min}}}^{e_{\text{max}}} \left[ \frac{k_0 \exp(e / RT)p_i}{1 + k_0 \exp(e / RT)p_i} \right] F(e) \, de \]  

(118)

where \( F(e) \) represents the number of sites with energy between \( e \) and \( e + de \). The problem considered is to recover an estimate of the distribution \( F(e) \) from a given set of noisy data \( \{ p_i, a(p_i), i = 1, \ldots, N \} \). This is clearly an ill-posed inverse problem with the term in the square bracket (i.e. the local Langmuir isotherm) serving as the kernel.

The performance of the various linear regularisation method was tested for a simulated noisy data set for a true energy distribution given by: 

\[ F(e) = 1.125 \exp \left( \frac{-(e-2.5)^2}{0.5} \right) \]  

(119)

The total adsorption data was generated at 100 equispaced points in the range \( 1 < p_i < 1000 \text{ mbar} \). For each \( p_i \), the integral (118) was evaluated numerically using the true distribution (119) over the range \( 1 < e < 4 \text{ kcal/mole} \). The result was then contaminated with 10 percent random noise drawn from a uniform distribution to simulate experimental data. The constant \( k_0 \) was taken as \( 3.2 \times 10^{-9} (\text{mbar})^{-1} \) for the generation of the data and the temperature was fixed at \( T = 77.5K \).
Figures 2.11 illustrates the distributions recovered using of zero, first, second and eighth order linear regularisation at regularisation levels of $\lambda = 10^{-5}$, 1. and $10^5$.

![Figures 2.11 - Energy distribution recovered using various orders of regularisation.](image)

**Figure 2.11** – Energy distribution recovered using various orders of regularisation.

(----- True trend ----- $\lambda = 10^{-5}$ ----- $\lambda = 1$. ----- $\lambda = 10^5$)

The best results are obtained using zero order regularisation (see Figure 2.11a) which makes the least assumption about the form of the underlying function. As expected, the
first and second order regularisations tend to a constant and a line as the regularisation level is increased (see Figures 2.11b and c). At intermediate regularisation levels, the energy distributions recovered compromise between the true distribution and the (erroneous) *a priori* assumptions of similarity to a constant or a line. The eighth order regularisation is entirely inadequate and unrealistic. There is simply insufficient information in the simulated experimental data to allow the reliable estimation of eight free parameters. It is also clear that in each case there is an optimum level of regularisation. In the absence of knowledge of the true underlying function the determination of the optimum value of the regularisation parameter $\lambda$ by trial and error is wholly subjective. In the next section we present a fairly robust method for selecting the optimum level of regularisation based on the data alone.

2.6 - Cross Validation: Automatic selection of the regularisation parameter

The simple example cited above demonstrates that the solution of the inverse problem with appropriate choice of regularisation technique has an optimal value or range of values for the regularisation parameter which must in general be found by trial and error. A variety of procedures have been developed to enable the automatic selection of the appropriate level of regularisation based on the observed data alone. In this section we describe the *leave one out* cross validation technique which has a fairly solid theoretical foundation and can be applied to all linear regularisation techniques with good computational efficiency. The procedure described is a generalisation of the technique developed by Golub *et al* [1996] for ridge regression.
Consider the linear system corresponding to the general regularisation problem,

\[
(R^T R + \lambda \Psi^T \Psi) \hat{u}_\lambda = R^T \gamma 
\]

The solution of this system for any \( \lambda \) is given by,

\[
\hat{u}(\lambda) = (R^T R + \lambda \Psi^T \Psi)^{-1} R^T \gamma 
\]

Cross validation works by leaving out the data points one at a time and recomputing the best fit with the remaining \( N-1 \) data points. Let \( e_k \) be the \( k \)th unit vector and define \( D_k = I - e_k e_k^T \) as the \( N \times N \) diagonal matrix with all diagonal elements equal to 1 except the \( k \)th element which is equal to zero. The optimal parameters based on \( (N-1) \) data points excluding the \( k \)th observation is determined from,

\[
T A(k) = h (y - R \hat{u}_k) + x k w T^T u_k \]

and is given by,

\[
\hat{u}_k(X) = T R D_k D_k R + M T^T \psi^T \psi \hat{u}_k 
\]

Next consider reinstating the \( k \)th point and calculating the residual using \( \hat{u}_k(X) \):

\[
\| y - R \hat{u}_k(\lambda) \|^2 = \| D_k[y - R \hat{u}_k(\lambda)] \|^2 + (L_k^T \hat{u}_k(\lambda) - y_k)^2
\]

where \( y_k \) is the \( k \)th observation and \( L_k^T \) is the \( k \)th row of the matrix \( R \). The term \((L_k^T \hat{u}_k(\lambda) - y_k)^2\) represents the increase in the sum of squared errors resulting when the \( k \)th row of matrix \( R \) and the \( k \)th element of the vector \( y \) are "reinjected". The (leave one out) cross validation estimate of \( \lambda \) is obtained by forming the criterion,
Choosing $\lambda$ by minimising $CV(\lambda)$ is equivalent to selecting the regularisation level such that the fitted model is not overly dependent on any one experiment. Minimising $CV(\lambda)$ as it stands is not a practical proposition since for each value of $\lambda$ we must solve an $M \times M$ system $N$ times to obtain $\hat{u}_k(\lambda), k=1,2,...,N$. Fortunately, this can be avoided following the algebraic manipulation below which enables expressing $CV(\lambda)$ in terms of $\hat{u}(\lambda)$ rather than $\hat{u}_k(\lambda)$. Since $D_k^T D_k = D_k^T = D_k$ and $R^T D_k^T D_k R = R^T R - \ell_k \ell_k^T$, equation (122) can be written as,

$$\hat{u}_k(\lambda) = [(R^T R + \lambda \Psi^T \Psi)^{-1} - \ell_k]^{-1} R^T (I - e_k e_k^T) y$$

Using the Sherman-Morrison formula [Press et al, 1992],

$$Z^{-1} = Z^{-1} + \frac{Z^{-1} u u^T Z^{-1}}{1 - u^T Z^{-1} u}$$

and taking $(R^T R + \lambda \Psi^T \Psi)^{-1}$ as $Z$ we have,

$$[Z - \ell \ell^T]^{-1} = Z^{-1} + \frac{Z^{-1} \ell \ell^T Z^{-1}}{1 - \ell^T Z^{-1} \ell}$$

Equation (125) can therefore be written as

$$\hat{u}_k(\lambda) = \left[ Z^{-1} + \frac{Z^{-1} \ell \ell^T Z^{-1}}{1 - \ell^T Z^{-1} \ell} \right] [R^T y - \ell_k]$$

Substituting for $Z^{-1} R^T y = \hat{u}(\lambda)$ leads to,
\begin{align*}
\hat{u}_k(\lambda) &= \hat{u}(\lambda) + \left[ \frac{Z^{-1}r_k^T u(\lambda)}{1-r_k^T Z^{-1}r_k} \right] - \frac{Z^{-1}r_k^T Z^{-1}r_k y_k}{1-r_k^T Z^{-1}r_k} - Z^{-1}r_k y_k
\end{align*}
(129)

Noting that $r_k^T Z^{-1}r_k$, $r_k^T \hat{u}(\lambda)$ and $y_k$ are all scalars, equation (129) can be rearranged to,
\begin{align*}
\hat{u}_k(\lambda) &= \hat{u}(\lambda) + \left[ \frac{r_k^T \hat{u}(\lambda)-y_k}{1-r_k^T Z^{-1}r_k} \right] Z^{-1}r_k
\end{align*}
(130)

Multiplying (130) by $r_k^T$ and then subtracting $y_k$ leads to
\begin{align*}
\frac{r_k^T \hat{u}_k(\lambda)-y_k}{1-r_k^T Z^{-1}r_k}
\end{align*}
(131)

Substituting equation (131) back in (124) gives the desired results,
\begin{align*}
CV(\lambda) &= \frac{1}{N} \sum_{k=1}^{N} \left[ \frac{r_k^T \hat{u}(\lambda)-y_k}{1-r_k^T (R^T R + \lambda \Psi^T \Psi)^{-1} r_k} \right]^2
\end{align*}
(132)

which expresses $CV(\lambda)$ in terms of $\hat{u}(\lambda)$. The computational burden for calculating $CV(\lambda)$ for a given $\lambda$ is now reduced to solving a single $M \times M$ system for $\hat{u}(\lambda)$. This is still time-consuming when the minimum of $CV(\lambda)$ is sought. The computational efficiency can be significantly increased by resorting to the generalised singular value decomposition (GSVD) technique.

The cross validation criterion (132) can be expressed in matrix form as,
\begin{align*}
CV(\lambda) &= \frac{1}{N} \sum_{k=1}^{N} \left[ \frac{e_k^T \left[ I - R(R^T R + \lambda \Psi^T \Psi)^{-1} R^T \right] y}{e_k^T \left[ I - R(R^T R + \lambda \Psi^T \Psi)^{-1} R^T \right] e_k} \right]^2
\end{align*}
(133)

It is convenient at this point to introduce the $N \times N$ "smoother" matrix $H(\lambda)$ defined as,
\[ H(\lambda) = R(R^T R + \lambda \Psi^T \Psi)^{-1} R^T \]  

(134)

which enables us to express \( CV(\lambda) \) as,

\[
CV(\lambda) = \frac{1}{N} \sum_{k=1}^{N} \left[ \frac{\varepsilon_k^T [I_N - H(\lambda)] \varepsilon_k}{\varepsilon_k^T [I_N - H(\lambda)] \varepsilon_k} \right]^2
\]

(135)

It is important to note the physical significance of the smoother matrix \( H(\lambda) \). Recalling

that,

\[
\hat{u}(\lambda) = (R^T R + \lambda \Psi^T \Psi)^{-1} R^T \gamma
\]

C.f. (120)

and

\[
\hat{\gamma} = R \hat{u}(\lambda)
\]

(136)

it follows immediately that,

\[
\hat{\gamma} = R(R^T R + \lambda \Omega)^{-1} R^T \gamma
\]

\[ = H(\lambda) \gamma \]

(137)

The smoother matrix \( H(\lambda) \) operates on the noisy observed vector \( \gamma \) to produce the “smoothed” estimated vector \( \hat{\gamma} \). Changing the regularisation level alters the eigenvalues of the smoother matrix \( H(\lambda) \) and hence adjusts the level of smoothing or regularisation applied. The trace of the smoother matrix \( H(\lambda) \), which is the sum of its eigenvalues, serves as an overall measure of the level of smoothing applied and is a measure of the effective degrees of freedom (df) of the model [Hastie and Tibshirani, 1990].

The evaluation of \( H(\lambda) \), and hence \( CV(\lambda) \), at each trial value of \( \lambda \) requires the inversion of the \( M \times M \) matrix \( (R^T R + \lambda \Psi^T \Psi) \) and may prove too time consuming. This can be avoided by resorting to the GSVD technique. Let \( R = U_R D_R X^{-1} \) and
\[ \Psi = U_\Psi D_\Psi X^{-1} \] represent the GSVD of matrices \( R \) and \( \Psi \) respectively. Substituting these into \( H(\lambda) \) and using the orthogonality property of \( U_R \) and \( U_\Psi \) enables us to express \( H(\lambda) \) as:

\[
H(\lambda) = U_R D_R (D_R^T D_R + \lambda D_\Psi^T D_\Psi)^{-1} D_R^T U_R^T
\]

(138)
The inversion required is now that of the diagonal matrix \( (D_R^T D_R + \lambda D_\Psi^T D_\Psi) \) which is computationally trivial. Finally, substituting for \( H(\lambda) \) into (135) we arrive at:

\[
CV(\lambda) = \frac{1}{N} \sum_{k=1}^{N} \left[ y_k - \sum_{j=1}^{r} (u_R)_{kj} \left( \frac{d^2_{R_j}}{d^2_{R_j} + \lambda d^2_{\Psi_j}} \right) b_j \right]^2
\]

(139)

where, \( b = U^T y \) and \( r \) is the effective rank of \( (R^T | \Psi^T) \). Minimisation of equation (139) is now a computationally manageable task and can be performed automatically and efficiently using any appropriate univariate optimisation technique. To our knowledge this procedure has not been reported previously.

Returning to our adsorption example, Figure 2.12 shows the general behaviour of the cross validation criterion \( CV(\lambda) \) versus \( \log(\lambda) \) for zero, first, second and eighth order linear regularisation techniques. We note immediately that the complexity of the function \( CV(\lambda) \) increases with the increasing order of regularisation. The zero order regularisation method, which is most compatible with the underlying trend of the example considered, shows a very well defined and hence easily determined minimum.
The first and second order regularisation techniques exhibit a single minimum but the minimum is less well defined. In contrast, the eighth order regularisation has a \( CV(\lambda) \) with multiple minima which would wreak havoc with any minimisation procedure. This is not surprising as there is insufficient information in the data to warrant a high order regularisation operator.

Figure 2.12 – Variation of CV criterion with \( \lambda \) for different orders of regularisation.

Figures 2.13 and 2.14 show the energy distributions recovered and the corresponding total adsorption isotherms at the optimum level of regularisation. It is evident that the zero order regularisation does a remarkable job and recovers the underlying energy distribution closely. First and second order regularisation techniques perform less adequately and the eighth order regularisation is hopeless as expected. The damping effect of the integral is explicitly shown in Figure 2.14 where the predicted adsorbed amounts at the optimum level of regularisation are virtually the same for all regularisation schemes despite the large differences between their associated energy distributions.
Figure 2.13 – Energy distributions recovered by the automatic selection of the regularisation level.

(T = 77.5K, $\lambda^* = 3 \times 10^{-4}$)

(T = 77.5K, $\lambda^* = 5 \times 10^{-2}$)

(T = 77.5K, $\lambda^* = 110$)

(T = 77.5K, $\lambda^* = 1.8 \times 10^6$)
Figure 2.14 – Adsorption isotherms predicted by the automatic selection of the regularisation level.

( ———— True trend, ——— Predictions • Noisy data )
2.7 – Conclusion

The training algorithms of feed-forward neural networks share many aspects of the procedures developed in the vast topic of multivariate function approximation. The primary aim of this chapter was to introduce the basic concepts of function approximation within a simple univariate setting. Much of the material presented, for example least square regression and robust estimation, are well known to most practitioners. Working with a noisy data set demands some form of stabilisation to prevent over-fitting the noisy data. Various direct and indirect stabilisation methods, in particular the linear regularisation technique, were considered in some detail for two reasons. First, the subject is less widely known and second, it provides an easy introduction to the more complex multivariate linear regularisation theory considered in the next chapter.

Selecting an “optimal” level of stabilisation is a difficult problem and was therefore considered worthy of detailed attention. The appropriate level of stabilisation is highly problem dependent and its simplistic selection may prove subjective. A computationally efficient, and in parts novel, procedure for the automatic selection of the regularisation level based on the data alone was considered in section 2.6. In particular, the powerful generalised singular value decomposition was used to reduce the computational burden of the leave one out cross validation method significantly. To our knowledge this procedure has not been previously reported and provides a reliable and less subjective method for selecting the optimal regularisation level. The method was demonstrated for the recovery of the unknown energy distribution of a heterogeneous solid adsorbent. The cross validation technique developed in this chapter can be naturally extended to problems with multivariate inputs and lies at the heart of the computational procedures developed in succeeding chapters.
Chapter 3

Regularisation of Multivariate RBF Networks

The close relationship between the function approximation problem and feed-forward artificial neural networks was explored in the previous chapter in a univariate setting. Within this viewpoint, feed-forward neural networks may be viewed as approximation techniques for reconstructing input-output mappings in high-dimensional spaces. In neural network parlance, training (learning) is equivalent to finding a hyper-surface in a multidimensional space that provides the best fit for the training data and generalisation is equivalent to the use of this multidimensional surface to interpolate within the domain of the data where there are no examples. Both projection and kernel based neural networks (see section 2.2) have been shown to posses reliable approximation properties for the reproduction of multivariate non-linear functions. Wang et al [1996] showed that a projection based feed-forward neural network with two hidden layers employing sigmoidal basis functions is capable of approximating many non-linear functions. Kernel based radial Basis Function (RBF) networks have also been shown to posses good approximation capabilities. Poggio and Girosi [1990a&b] report that among all feedforward networks RBF networks posses the best approximation property. Hunt et al [1992] present further theoretical support for RBF networks.
The training of projection based networks always reduces to the solution of a large-scale non-linear optimisation problem. Such problems are usually very time consuming and often encounter severe convergence problems. In contrast, the training of RBF networks with specified non-linearities reduces to the solution of an over-determined set of linear equations which can be solved by a variety of highly stable techniques. In this chapter we focus on kernel based neural networks with particular emphasis on RBF networks. These networks have a close connection with the well studied subject of multivariate function approximation and enjoy a firm theoretical foundation.

This chapter starts with an introduction to multivariate linear regularisation theory to pinpoint the relevance of radial basis functions (RBFs). Linear differential operators and their corresponding Green's functions, which play an essential role in the theoretical development, are introduced briefly. It is shown that the solution of the multivariate regularisation problem can be represented as a single hidden layer network, known as the Regularisation network. The basic concepts introduced in section 2.6 are used to develop a new working equation for the efficient calculation of the optimal regularisation level of a Regularisation network.

Regularisation networks are traditionally constructed using isotropic Gaussian basis functions. An illustrative example is employed to investigate the effect of the regularisation level and the value of the isotropic spread $\sigma$ on the performance of a Regularisation network. It is shown that a Regularisation network only performs
adequately with appropriate choice of isotropic spread and the optimal level of regularisation.

The effective degrees of freedom, $df$, of a Regularisation network is a function of the regularisation level $\lambda$ and the isotropic spread $\sigma$, an important point which has received little attention. A simple measure of the effective degrees of freedom is introduced and is used to establish a convenient procedure for selecting the appropriate value of the isotropic spread $\sigma$. To the best of our knowledge, this approach has not been addressed previously and leads to significant improvement in the performance of the Regularisation network.

3.1- Multivariate Linear Regularisation Theory

The benefits of stabilising or regularising the estimation of the model parameters were illustrated in the previous chapter with reference to a univariate problem. In this section we present a brief introduction to the linear regularisation theory for multivariate problems. This is a vast and complex subject relying on fairly advanced mathematical concepts. The aim is to provide a readable introduction rather than a rigorous mathematical treatment, the latter can be found in the classic text by Courant and Hilbert [1970] and the articles by Poggio and Girosi [1990a,b].
We start by considering the strict interpolation problem and the remarkable theorem due to Michelli [1986] which pinpoints the importance of radial basis functions in multivariate function approximation and neural networks. The strict interpolation problem may be stated as:

"Given a set of N inputs \((\mathbf{x}_i \in \mathbb{R}^P, i = 1,\ldots,N\) ) and the corresponding outputs \((y_i, i = 1,\ldots,N\) ) find a continuous multivariate function \(F(\mathbf{x})\) which maps the inputs to the output and satisfies the N interpolating conditions:

\[
F(\mathbf{x}_i) = y_i, \quad i = 1,\ldots,N
\]

Let us expand \(F(\mathbf{x})\) as a linear summation of \(N\) radial basis functions each centred at a distinct data point,

\[
F(\mathbf{x}) = \sum_{j=1}^{N} w_j \phi_j(\| \mathbf{x} - \mathbf{x}_j \|)
\]

Here \(\phi_j(r)\) represents a radial function whose argument is a measure of the (Euclidean) distance from the known centre located at \(\mathbf{x}_j\), \(r = \| \mathbf{x} - \mathbf{x}_j \|\), and the \(w_j\)'s are the weighting coefficients. Equations (1) and (2) can be combined and stated in the compact form,

\[
F = \Phi w
\]

where \(F = [F(\mathbf{x}_1), F(\mathbf{x}_2),\ldots,F(\mathbf{x}_N)]^T\), \(w = [w_1,\ldots,w_N]^T\) and \(\Phi\) is the \(N \times N\) interpolation matrix with elements,

\[
\phi_j = \phi(\| \mathbf{x}_i - \mathbf{x}_j \|)
\]
The theorem originally proved by Michelli [1986] for multiquadrics basis functions states that (in exact or infinite precision arithmetic):

"if $x_1, x_2, \ldots, x_N$ are $N$ distinct points in $\mathbb{R}^P$ the interpolation matrix $\Phi$ is nonsingular."

All that is required is that the input vectors are distinct and the elements of the interpolation matrix are based on radial functions centred on the known distinct data points. In theory, therefore, the system (3) can always be solved to obtain the unique optimal weights:

$$ w^* = \Phi^{-1} y $$

and the continuous function (2) with the optimal weights will satisfy the $N$ interpolating conditions (1) exactly. We should remark here that in practice we always deal with finite rather than infinite precision arithmetic and as a result the interpolation matrix $\Phi$ may turn out to be numerically singular. This does not violate Michelli’s theorem, it only tells us that we must use greater precision for our calculation. The strict interpolation problem always has a solution in terms of radial basis functions centred at the data points irrespective of the size of the data set $N$ or the dimensions of the input vector $x$.

A large class of radial basis functions are covered by Michelli’s theorem [Powell, 1987a&b; Haykin, 1999]. The functions shown in Table 3.1 and Figure 3.1 are of particular interest in the studies of multivariate function approximation and its implementation as RBF networks. The Gaussian and Inverse Multiquadric functions are localised and Michelli has proved that for such basis functions the interpolation matrix $\Phi$
is positive definite. The Multiquadric and Thin Plate Spline functions are global and for such basis functions the interpolation matrix $\Phi$ is indefinite with $N-1$ negative and one positive eigenvalues [Haykin, 1999]. In principle, global radial basis functions which grow to infinity with their arguments construct a smoother fit than local radial basis functions. On the other hand local radial basis functions can extract more specialised features due to their locality.

Figure 3.1 – Examples of radial basis functions (RBF) with $r = \| x - 3 \|$. 
Table 3.1 – Various types of Radial Basis Functions

<table>
<thead>
<tr>
<th>Radial Basis Functions</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gaussian</td>
<td>( \phi(x;t,\sigma) = \exp\left(-\frac{|x-t|^2}{\sigma^2}\right) )</td>
</tr>
<tr>
<td>Multiquadrics</td>
<td>( \phi(x;t,\sigma) = \left[|x-t|^2 + \sigma^2\right]^{1/2} )</td>
</tr>
<tr>
<td>Inverse Multiquadrics</td>
<td>( \phi(x;t,\sigma) = \frac{1}{\left[|x-t|^2 + \sigma^2\right]^{1/2}} )</td>
</tr>
<tr>
<td>Thin plate splines</td>
<td>( \phi(x;t,\sigma) = \left[\frac{|x-t|}{\sigma}\right]^{2n} \ln\left(\frac{|x-t|}{\sigma}\right) )</td>
</tr>
</tbody>
</table>

3.1.1 - Ill-Posed Problems and the Need for Regularisation

Given a set of \( N \) distinct data points (exemplars) we can always construct a continuous multivariate function (a hyper surface) which interpolates the observed data exactly. However, this interpolating function may have little or no bearing to the true underlying relationship (mapping) between the multivariate inputs and the output. Recovering the true underlying function from a limited noisy data set is an inherently ill posed problem [Tikhonov and Arsenin, 1977].

Consider a mapping process that transforms the input vector \( x \) contained in a given domain \( \mathbb{R}^P \) to the output \( \tilde{y} \) confined to the range \( \mathbb{R}^1 \). A schematic of such a mapping \( \{\tilde{y}_i = F(x_i) : \mathbb{R}^P \rightarrow \mathbb{R}^1, i = 1,\ldots,N\} \) is shown below [Haykin, 1999],
The reconstruction of the true underlying mapping from a discrete data set may be considered well posed if the following conditions are satisfied:

a) **Existence:** for every input vector $\mathbf{x} \in \mathbb{R}^P$ there exists an output $\mathbf{y} = F(\mathbf{x}) \in \mathbb{R}^1$.

b) **Uniqueness:** for any pair of input vectors $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^P$ we have $F(\mathbf{x}_1) = F(\mathbf{x}_2)$ if and only if $\mathbf{x}_1 = \mathbf{x}_2$.

c) **Continuity:** the mapping is continuous, in the sense that for any $\varepsilon > 0$ a scalar $\delta = \delta(\varepsilon)$ can be found such that $\| F(\mathbf{x}_1) - F(\mathbf{x}_2) \| < \varepsilon$ implies that $\| \mathbf{x}_1 - \mathbf{x}_2 \| < \delta$.

In general, one or more of the above conditions are violated making the reconstruction of the true underlying mapping an inherently *ill posed* problem. First, a distinct output may not exist for every input vector. Second, there may be insufficient information in the data set to enable the reconstruction of a unique mapping, the continuity condition is then violated. Finally, real data are inevitably corrupted with measurement noise. If the noise level is large, an imprecise input vector $\mathbf{x} \in \mathbb{R}^P$ may give rise to an output $\mathbf{y}(\mathbf{x}) \in \mathbb{R}^1$ and
both the uniqueness and the continuity conditions are again violated. In short, the reconstruction problem is inherently *ill-posed* because quite large noisy data sets may actually contain little information about the true underlying process generating the data. There is no magic cure for lack of information and the ill-posed problem can only be turned into a well posed one by introducing additional *a priori* knowledge or assumptions about the true underlying process. Regularisation in its broadest sense is a means for introducing such *a priori* knowledge or belief.

3.1.2- Linear Differential Operators and Green's Function

The general solution of linear regularisation problems relies heavily on concepts of linear differential operators and the close relationship between the solution of partial differential and integral equations. We pause here to briefly introduce the basic concepts involved, rigorous proofs are given by Courant and Hilbert [1970].

Consider a process described by the differential equation,

\[ L[\tilde{y}(x)] = \psi(x) \]  

(6)

where \( L[.\] is a linear differential operator. The response \( \tilde{y}(x) \) for a given forcing function \( \psi(x) \) and prescribed linear boundary conditions must be obtained by solving the partial differential equation (6) which constitutes a complex boundary value problem. The solution of (6) can be expressed in terms of the integral transformation of the right hand side,
\[ \bar{y}(x) = \int_{\mathbb{R}^p} G(x, \xi) \psi(\xi) d\xi \]  

(7)

where \( G(x, \xi) \) is the Green's (or influence) function corresponding to the linear differential operator \( L[\,] \) and must be constructed to have the following properties,

a) For a fixed \( \xi \), \( G(x, \xi) \) is a function of \( x \) and satisfies the boundary conditions.

b) Except at the point \( x = \xi \) where \( G(x, \xi) \) has a singularity, the derivatives of \( G(x, \xi) \) with respect to \( x \) are all continuous. The maximum order of the derivatives of \( G(x, \xi) \) is determined by the order of the linear operator \( L[\,] \).

c) Viewed as a function of \( x \), \( G(x, \xi) \) satisfies the partial differential equation,

\[ L[G(x, \xi)] = \delta(x - \xi) \]  

(8)

where \( \delta(x - \xi) \) is the Dirac delta function centred at the point \( x = \xi \). That is \( G(x, \xi) \) satisfies the partial differential equation

\[ L[G(x, \xi)] = 0 \]  

(9)

everywhere except at the point \( x = \xi \), where \( G(x, \xi) \) has a singularity.

The Green's function \( G(x, \xi) \) plays a role for a linear differential operator \( L[\,] \) that is similar to that of the inverse matrix for a matrix equation. In essence the solution of the partial differential equation (6) with prescribed boundary conditions on \( \bar{y}(x) \) and its derivatives is transformed to the solution of the integral equation (7). This would of course require the determination of the Green’s function \( G(x, \xi) \) and solving a boundary
value problem governing $G(x, \xi)$ which is often simpler than the original boundary value problem.

The starting point for the solution of (6) by the Greens method is the introduction of the adjoint operator $L^* [.]$ associated with the linear operator $L[.]$. One way to define $L^* [.]$ is to consider an arbitrary function $v(x)$, form the product $v(x)L[\overline{y}(x)]$ and integrate it over the domain of interest:

$$\int_{\Delta} v(x) L[\overline{y}(x)] dx = [\text{boundary terms}]_{\Delta} + \int_{\Delta} \overline{y}(x) L^*[v(x)] dx \quad (10)$$

We note that $[\text{boundary terms}]_{\Delta}$ is obtained by repeated integration by parts and involves the values of $\overline{y}(x)$ and its derivatives, some of which may be specified at the domain boundary. It also contains the values of the arbitrary function $v(x)$ and its derivatives over the domain boundary which are free and can be assigned any value. The boundary values on $v(x)$ can therefore be chosen such that the $[\text{boundary terms}]_{\Delta}=0$. This leads to the conclusion that for any pair of arbitrary functions $\overline{y}(x)$ and $v(x)$ which are sufficiently differentiable we can always find appropriate boundary conditions such that:

$$\int_{\Delta} v(x) L[\overline{y}(x)] dx = \int_{\Delta} \overline{y}(x) L^*[v(x)] dx \quad (11)$$

Equation (11) is called the Green's identity, it provides the mathematical basis for defining the adjoint operator $L^* [.]$ associated with operator $L[.]$. If the operator $L^* [.] = L[.]$ and the boundary conditions on $\overline{y}(x)$ and $v(x)$ turn out to be identical, the operator $L[.]$ is called self-adjoint. Loosely speaking, the adjoint operator $L^* [.]$ plays a
role similar to a matrix transpose in a matrix equation and a self-adjoint operator \( L[.\] is equivalent to a symmetric matrix. The above brief introduction to the Greens method permits the handling of the multivariate linear regularisation problem in a very abstract and general way. Detailed description of the Greens method is given by Courant and Hilbert [1970] and its application to engineering problems is considered by Greenberg [1971].

3.1.3 - Solution of the Multivariate Linear Regularisation Problem

The basic problem of multivariate linear regularisation which was first proposed by Tikhonov [1963] can be stated as:

"Given a set of \( N \) noisy exemplars \( \{ x_i, y_i, i = 1, \ldots, N \} \) determine the continuous multivariate function \( F(x) \) which gives the best approximation to the true underlying relationship between the inputs and the output."

The problem is addressed by forming the following regularisation functional and attempting to minimise it over a set of admissible functions,

\[
\min_F \mathfrak{F}(F) = A(F) + \lambda B(F) \tag{12}
\]

The first term \( A(F) \) is constructed to measure the fidelity of the solution \( F_\lambda(x) \) to the data and the second term \( B(F) \) is used to penalise the \( F_\lambda(x) \) for disagreement with the \textit{a priori} knowledge or belief about the true underlying relationship. The functional \( A(F) \)
representing deviation from the observed data (the standard error term) is usually taken as,

\[ A(F) = \frac{1}{2} \sum_{i=1}^{N} (y_i - F(x_i))^2 \]  

(13)

and the regularising (stabilising) term \( B(F) \) is defined as,

\[ B(F) = \frac{1}{2} \int_{\mathcal{P}} (D[F(x)])^2 \, dx \]  

(14)

where \( D[.\]\) is a linear differential operator and the multidimensional integration is over the domain of the input vector \( x \). The \textit{a priori} knowledge or belief about the true underlying relationship is embedded in the linear differential operator \( D[.] \). The regularisation functional to be minimised can be expressed as,

\[ \min_{F} \mathcal{J}(F) = \frac{1}{2} \sum_{i=1}^{N} (y_i - F(x_i))^2 + \frac{1}{2} \lambda \int_{\mathcal{P}} (D[F(x)])^2 \, dx \]  

(15)

where \( \lambda \) is the regularisation parameter which controls the compromise between the \textit{a priori} information about the true underlying relationship and the closeness to the data.

To develop a working equation for minimising (15) we must first introduce a measure for the variation of functional \( \mathcal{J}(F) \) with respect to the function \( F \). The appropriate measure is the Frechet differential of a functional which is defined as [Dorny, 1975],

\[ d\mathcal{J}(F, h) = \lim_{\beta \to 0} \left[ \frac{d\mathcal{J}(F + \beta h)}{d\beta} \right] \]  

(16)

where \( h(x) \) is an arbitrary function of the input vector \( x \) and \( \beta \) is a scalar and the ordinary rules of differentiation apply under the limit. The necessary condition for a
function $F_\lambda$ to be a minimiser of the functional (12) is that its Frechet differential vanishes [Haykin, 1999],

$$d\mathcal{A}(F,h) = dA(F,h) + \lambda dB(F,h) = 0$$  \hspace{1cm} (17)

The Frechet differential of the standard error term $dA(F,h)$ given by:

$$dA(F,h) = \frac{1}{2} \lim_{\beta \to 0} \frac{d}{d\beta} \sum_{i=1}^{N} \left[ y_i - F(x_i) - \beta (x_i) \right]^2$$

$$= -\sum_{i=1}^{N} (y_i - F(x_i)) h(x_i)$$  \hspace{1cm} (18)

It is convenient at this point to introduce the function $\delta_{x_i}$ defined as the multivariate Dirac delta distribution of $x$ centred at the point $x_i$:

$$\delta_{x_i} = \delta(x - x_i) = \begin{cases} 1 & \text{if } x = x_i \\ 0 & \text{if } x \neq x_i \end{cases}$$  \hspace{1cm} (19)

Using the sifting property of the Dirac delta function $\delta_{x_i}$ the summation in equation (18) can be replaced with an equivalent integral as,

$$dA(F,h) = -\int \left[ h(x) \sum_{i=1}^{N} (y_i - F(x)) \right] \delta_{x_i} d\vec{x}$$  \hspace{1cm} (20)

The Frechet differential of the regularisation term $dB(F,h)$ can be expressed as,

$$dB(F,h) = \frac{1}{2} \lim_{\beta \to 0} \frac{d}{d\beta} \left[ D[F(x)] + \beta h(x) \right]^2 d\vec{x}$$

$$= \lim_{\beta \to 0} \left[ R^p \left( D[F(x)] + \beta h(x) \right) D[h(x)] \right] d\vec{x}$$

$$= \left[ R^p D[F(x)] D[h(x)] \right] d\vec{x}$$  \hspace{1cm} (21)

Using the Green’s identity (11) and taking:
\( v(x) = D[F(x)] \) and \( L[\tilde{y}(x)] = D[h(x)] \)

(22)

equation (21) can be expressed in the equivalent form,

\[
dB(F,h) = \int_{R^p} h(x) D^* [D[F(x)]] d\lambda
\]

(23)

We are now in a position to add the two contributions \( dA(F,h) \) and \( dB(F,h) \) to arrive at the necessary optimality condition:

\[
d\mathcal{J}(F,h) = dA(F,h) + \lambda dB(F,h)
\]

(24)

It follows that for a given \( \lambda \) the Frechet differential \( d\mathcal{J}(F,h) \) is zero for an arbitrary function \( h(x) \) if and only if the function \( F_\lambda(x) \) satisfies the following (Euler-Lagrange) partial differential equation:

\[
D^* D[F_\lambda(x)] = \frac{1}{\lambda} \sum_{i=1}^{N} (y_i - F_\lambda(x)) \delta(x - x_i)
\]

(25)

We can now use the ideas presented in Section 3.1.2 and express the solution to the regularisation problem (25) in terms of the Green's function corresponding to the linear differential operator \( L = D^* D \) as,

\[
F_\lambda(x) = \int_{R^p} G(x, \xi) \left\{ \frac{1}{\lambda} \sum_{i=1}^{N} (y_i - F_\lambda(x)) \right\} \delta(\xi - \xi_i) d\xi
\]

(26)

\[
= \frac{1}{\lambda} \sum_{i=1}^{N} (y_i - F_\lambda(x_i)) \int_{R^p} G(x, \xi) \delta(\xi - \xi_i) d\xi
\]

Finally on using the sifting property of the Dirac delta function we arrive at,

\[
F_\lambda(x) = \sum_{i=1}^{N} \left( \frac{y_i - F_\lambda(x_i)}{\lambda} \right) G(x, \xi_i)
\]

(27)
We may therefore conclude that the minimising solution $F_\lambda(x)$ can be represented in terms of a linear superposition of $N$ Green functions $G(x, x_i)$ centred at the individual data points $x_i$ with the weights $w_i = \frac{y_i - F_\lambda(x_i)}{\lambda}$ serving as the coefficients of the expansion. The influence of the regularisation parameter $\lambda$ is embedded in the unknown weights $w_i$'s.

To determine the unknown weights $w_i$'s, let $y = [y_1, y_2, ..., y_N]^T$ represent the vector of measured responses, $w_\lambda = [w_1, w_2, ..., w_N]^T$ represent the vector of the unknown weights and $F_\lambda = [F_\lambda(x_1), F_\lambda(x_2), ..., F_\lambda(x_N)]^T$ be the vector of minimising solution. Equation (27) can then be written in the matrix form,

$$E_\lambda = Gw_\lambda$$

(28)

where $G$ is the $N \times N$ symmetric Green's matrix with elements,

$$G_{ij} = G(x_i, x_j)$$

(29)

and:

$$w_\lambda = \frac{1}{\lambda} (y - E_\lambda)$$

(30)

Eliminating $F_\lambda$ between (28) and (30) leads to the following system of linear equations,

$$(G + \lambda I_N)w_\lambda = y$$

(31)

that can be solved to obtain the unknown weight vector $w_\lambda$. It should also be noted that equation (28) is not the complete solution of (25). In fact as it will be shown in chapter 5, all the functions that lie in the null space of the operator $D$ are invisible to the regularisation term of (14).
3.2 – The Regularisation Network

Equation (28) which is the solution of the multivariate linear regularisation problem can be symbolised as the network shown in Figure 3.2 which is known as the Regularisation network [Poggio and Girosi, 1990a & 1990b]. The network consists of a single hidden layer with $N$ neurones and the activation function of the $j^{th}$ hidden neurone is a Green's function $G(x,x_j)$ centred at a particular data point $x_j$. The influence of the regularisation parameter $\lambda$ is embedded in the unknown synaptic weights $w_j$'s. Equation (39) provides the unknown synaptic weights.

![Figure 3.2 – The Regularisation network](image)

99
Poggio and Girosi [Poggio and Girosi, 1990b] pointed out that from the viewpoint of approximation theory the Regularisation network has the following desirable properties:

a) The Regularisation network is a *universal-approximator* which can approximate arbitrarily well any multivariate continuous function, given a sufficiently large \( N \) number of hidden units.

b) Since the approximation scheme derived from regularisation theory is linear in the unknown weights it follows that the corresponding merit function is convex (unimodal) and therefore a unique solution exists for the linear weights.

c) The solution computed by the regularisation network is optimal in the sense that the weights obtained minimise the regularisation functional defined by (15).

The Regularisation network expressed by (28) is in its most general form. Special forms of the Green's function emerge based on the *a priori* knowledge embedded in the operator \( D \). For example, if the operator \( D \) is selected to be *translationally invariant*, the corresponding Green's function \( G(x, x_j) \) depends only on the difference between the arguments \( x \) and \( x_j \),

\[
G(x, x_j) = G(x - x_j)
\]  

(32)

Furthermore, if the stabiliser \( D \) is selected to be both *translationally and rotationally invariant*, the corresponding Green's function will depend only on the Euclidean norm of the difference vector \( (x - x_j) \),

\[
G(x, x_j) = G(\| x - x_j \|)
\]  

(33)
3.2.1 – Regularisation Network with Isotropic Spreads

As a more specific example, if the stabiliser $D$ and its adjoint $D^*$ are defined as,

$$D = \sum_{n=0}^{\infty} \left( \frac{\sigma_i^{2n}}{n!2^n} \right)^{\frac{\gamma}{2}} \left[ \frac{\partial}{\partial x_1} + \frac{\partial}{\partial x_2} + \ldots + \frac{\partial}{\partial x_p} \right]^n$$

$$(34)$$

$$D^* = \sum_{a+b+\ldots+k=n} (-1)^n \left( \frac{\sigma_i^{2n}}{n!2^n} \right)^{\frac{\gamma}{2}} \left[ \frac{\partial}{\partial x_1} + \frac{\partial}{\partial x_2} + \ldots + \frac{\partial}{\partial x_p} \right]^n$$

$$(35)$$

The solution of equation (8) with $L = D^* D$ via the multidimensional Fourier transform technique [Poggio and Girrosi, 1990b] leads to a Green’s function in the form of a multidimensional *factorisable* Gaussian basis function which is both *translationally and rotationally invariant* and has an infinite number of continuous derivatives.

$$G(x,x_j) = \exp \left[ -\frac{(x-x_j)^2}{2\sigma_j^2} \right]$$

$$(36)$$

The $\sigma_j$ appearing in (36) denotes the *isotropic* spread of the $j^{th}$ Green’s function which is assumed identical for all input dimensions.
3.2.2 - Regularisation network with non-isotropic spreads

The Regularisation networks developed above uses isotropic spreads (Euclidean norm) in the argument of its Gaussian basis functions. This restriction can be removed by introducing a general \textit{weighted-norm} for the vector \((x - x_j)\) defined as,

\[
\| x - x_j \|_{C_j}^2 = (C_j(x - x_j))^T (C_j(x - x_j)) = (x - x_j)^T C_j^T C_j (x - x_j) = (x - x_j)^T \Sigma_j^{-1} (x - x_j)
\]

(37)

Here, \(C_j\) is a \(p \times p\) norm-weighting matrix with \(p\) the dimension of the input space and \(\Sigma_j^{-1} = C_j^T C_j\) is the variance-covariance matrix which controls both the spread and the orientation of the radial basis function \(G(\| (x - x_j) \|_{C_j})\). Three increasingly more general definitions of matrix \(\Sigma_j^{-1}\) can be identified.

a) Isotropic spreads: \(C_j = \sigma_j^{-1} I\)

In this case, the \textit{receptive field} of \(G(\| (x - x_j) \|_{C_j})\) consists of a hyper-sphere centred at \(x_j\) with its radius determined by \(\sigma_j\). Figure 3.3a illustrates a bivariate example centred at \((3, 3)\) with \(\sigma = 1\).

b) Diagonal norm weighting matrix: \(C_j = \text{diag}(\sigma_j^{-1}_1, \sigma_j^{-1}_2, \ldots, \sigma_j^{-1}_p)\)

In this case, the receptive field of \(G(\| (x - x_j) \|_{C_j})\) is a hyper-ellipse centred at \(x_j\) with its axes parallel to those of the input space and the spread along each axis determined by \(\sigma_{j,k}\). Figure 3.3b illustrates a bivariate example centred at \((3, 3)\) with \(C = \text{diag}(1.5, 0.5)\).
c) General case:

In the most general case, $\Sigma^{-1} = C_j^T C_j$ is a full positive definite matrix which establishes both the spread and orientation of the receptive field of $G(\|x - x_j\|_{C_j})$.

Figure 3.3c illustrates a bivariate example centred at (3, 3) with $C = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$.

a) Gaussian with isotropic spreads

b) Gaussian with diagonal $C_j$

c) Gaussian with full $C_j$

**Figure 3.3**—The Gaussian receptive field for different choices of the norm weighting matrix.
We note here that a full variance-covariance matrix $\Sigma_j^{-1}$ can always be transformed to the diagonal form by using the similarity transformation,

$$\Sigma_j^{-1} = Q_j \Lambda_j Q_j^T$$

$$= (\Lambda_j^{1/2} Q_j^T)^T (\Lambda_j^{1/2} Q_j^T)$$

$$= C_j^T C$$  \hspace{1cm} (38)$$

The columns of the orthogonal matrix $Q_j = [q_{j,1}, q_{j,2}, \ldots, q_{j,p}]$ are the eigenvectors of $\Sigma_j^{-1}$ while the diagonal elements of $\Lambda_j = \text{diag}[\lambda_{j,1}, \lambda_{j,2}, \ldots, \lambda_{j,p}]$ are the eigenvalues of the variance-covariance matrix. The rotation of the receptive field is controlled by the elements of the rotation matrix $Q_j$ and the widths of the receptive field along each individual input dimension is determined by $\lambda_{j,k}$. Introducing the transformation $(\hat{x} - \hat{x}_j) = (x - x_j)Q_j$ reduces the general case of the full $\Sigma_j^{-1}$ to the diagonal norm weighting matrix case.

Equations (34) and (35) show the stabilising operators leading to the isotropic Gaussians (36). The stabilising operators $D$ and $D^*$ for the other two cases are slightly different. For the general case of a symmetric and positive definite variance-covariance matrix $\Sigma^{-1}$ with $p \times p$ elements of $\sigma_{ij}$, the self-adjoint operator $L_{\Sigma^{-1}}$ is defined as

$$L_{\Sigma^{-1}} = (D^* D)_{\Sigma^{-1}} = \sum_{k=0}^{\infty} \left( \frac{(-1)^k}{k! 2^k} \right) \nabla^2 k$$  \hspace{1cm} (39)$$

where,
\[ \nabla^2 = \sum_{i=1}^{P} \sum_{j=1}^{P} \sigma_{ij}^{-1} \frac{\partial^2}{\partial x_j \partial x_i} \]  

and the associated Green's function is the solution of the following differential equation.

\[ L_{\Sigma^{-1}} G(x, x_j) = \delta(x - x_j) \]  

Using the multidimensional Fourier transform technique [Poggio and Girosi, 1990b] it can be shown that the solution of (41) is given by,

\[
\begin{align*}
G(x, x_j) &= \exp\left( -\frac{1}{2} \| x - x_j \| C_j \right) \\
&= \exp\left( -\frac{1}{2} (x - x_j)^T C_j C_j (x - x_j) \right) \\
&= \exp\left( -\frac{1}{2} (x - x_j)^T \Sigma_j^{-1} (x - x_j) \right)
\end{align*}
\]  

3.2.3 – Regularisation network with linear superposition of different basis functions

The multivariate linear regularisation technique is easily extended to a network employing a linear superposition of different Green’s functions corresponding to different differential operators. The Green’s functions maybe Gaussians with different definitions of the norm weighting matrices or they may belong to different classes of radial basis functions such as Thin plate splines, Multiquadrics or inverse Multiquadrics.

Let us assume that the function \( F \) obtained by minimising (15) is itself the linear superposition of \( m \) components \( f_l, l = 1, 2, \ldots, m \) with each component approximated by a
linear superposition of $M_l, l = 1, 2, ..., m$ Green's function with $\sum_{l=1}^{m} M_l = N$. With this assumption, the functional (15) can be expressed as,

$$\min_{F} \mathcal{J}(F) = \frac{1}{2} \sum_{i=1}^{N} \left( y_i - \sum_{l=1}^{m} f_l(x_i) \right)^2 + \frac{1}{2} \sum_{l=1}^{m} \lambda_l (D_l[f_l(x)])^2$$ (43)

It can be shown [Poggio and Girrosi, 1990b] that the Euler-Lagrange equations associated with (43) can be expressed as,

$$\sum_{l=1}^{m} \lambda_l D_l^* D_l \left[ f_{\lambda_l}(x) \right] = \sum_{i=1}^{N} \left( y_i - \sum_{l=1}^{m} f_{\lambda_l}(x) \right) \delta(x - x_i)$$ (44)

The function $F_\lambda(x)$ which minimises the functional (43) can be expressed as a linear superposition of linear superpositions of the Green’s functions $G_l, l = 1, 2, ..., m$ corresponding to the stabiliser $D_l, l = 1, 2, ..., m$. Using a similar mathematical manipulation to that described in section 3.1.3 the solution $F_\lambda(x)$ can be expressed as,

$$F_\lambda(x) = \sum_{l=1}^{m} \sum_{j=1}^{M_l} w_{j, \lambda_l} G_l(x_j x_j)$$ (45)

$$(y_i - \sum_{k}^{m} f_{\lambda_k}(x_i)) - f_{\lambda_j}(x_i)$$

where $w_{j, \lambda_l} = \frac{1}{\lambda_l} \frac{M_l}{\sum_{l=1}^{m} M_l}$ and $\lambda = [\lambda_1, \lambda_2, ..., \lambda_m]$ denotes $m$ separate regularisation parameters each operating on a component of $F_\lambda(x)$. Figure 3.4 illustrates a schematic representation of the Regularisation network described by (45).
In theory, the performance of a Regularisation network can be improved by using a suitable combination of various Green's functions with different stabilising operators. In practice, however, this approach becomes quite complex and proves computationally demanding. In particular, the number of unknown parameters increases significantly. Moreover, we need to estimate an appropriate level of regularisation for each of the components of $F_{\lambda}(x) = \sum_{l=1}^{m} f_{l}(x)$ which is by no means a trivial task. We do not consider such complex Regularisation network in this study.
3.3 - Automatic selection of the regularisation parameter (Revisited)

The selection of an appropriate level of regularisation is an important task in the training of any Regularisation network. The general derivation of the cross validation criterion for automatic selection of the regularisation level based on the data was presented in section 2.6. This derivation can not, however, be directly applied to the specific case of the Regularisation Networks. In this section we present a similar approach leading to an automatic selection procedure for Regularisation Networks. To our knowledge the working equation obtained has not been previously reported.

Consider the linear system corresponding to a standard Regularisation network,

\[(G + \lambda I) w_\lambda = y\]  \hspace{1cm} C.f.(31)

The solution of this system for any \(\lambda\) is given by,

\[w_\lambda = (G + \lambda I)^{-1} y\]  \hspace{1cm} (46)

Cross validation works by leaving out the data points one at a time and recomputing the best linear weights with the remaining \(N-1\) data points. Let \(e_k\) be the \(k^{th}\) unit vector and define \(D_k = I - e_k e_k^T\) as the \(N \times N\) diagonal matrix with all diagonal elements equal to 1 except the \(k^{th}\) element which is equal to zero. The optimal linear parameters based on \((N-1)\) data points excluding the \(k^{th}\) observation is determined as,

\[w_k(\lambda) = (D_k G + \lambda I)^{-1} y\]  \hspace{1cm} (47)

Next consider reinstating the \(k^{th}\) point and calculating the residual using \(\hat{u}_k(\lambda)\):

\[\| y - G w_k(\lambda) \|^2 = \| D_k [y - G w_\lambda] \|^2 + (G^T w_k(\lambda) - y_k)^2\]  \hspace{1cm} (48)
where \( y_k \) is the \( k^{th} \) observation and \( g_k^T \) is the \( k^{th} \) row of the matrix \( G \). The term
\[
(g_k^T w_k(\lambda) - y_k)^2
\]
represents the increase in the sum of squared errors resulting when the \( k^{th} \) row of matrix \( G \) and the \( k^{th} \) element of the vector \( y \) are "reinstated". The (leave one out) cross validation estimate of \( \lambda \) is obtained by minimising the criterion,
\[
\min_{\lambda} CV(\lambda) = \frac{1}{N} \sum_{k=1}^{N} (y_k - g_k^T w_k(\lambda))^2
\]
(49)

Minimising \( CV(\lambda) \) as it stands is not a practical proposition since for each value of \( \lambda \) we must solve \( (N-1) \times (N-1) \) system \( N \) times to obtain \( w_k(\lambda), k=1,2,\ldots,N \). Fortunately, this can be avoided following the algebraic manipulation below which enables expressing \( CV(\lambda) \) in terms of \( w_\lambda \) rather than \( w_k(\lambda) \). Since \( D_k^T D_k = D_k^T = D_k \) and \( D_k G = G - e_k g_k^T \), equation (47) can be written as,
\[
w_k(\lambda) = [(G + \lambda I) - e_k g_k^T]^{-1} y
\]
(50)

Using the Sherman-Morrison formula [Press et al, 1992] equation (50) can be written as
\[
w_k(\lambda) = \left[ (G + \lambda I)^{-1} + \frac{e_k g_k^T (G + \lambda I)^{-1}}{1 - g_k^T (G + \lambda I)^{-1} e_k} \right] y
\]
(51)

Substituting for \( (G + \lambda I)^{-1} y = w_\lambda \) leads to,
\[
w_k(\lambda) = \left[ I_N + \frac{e_k g_k^T}{1 - g_k^T (G + \lambda I)^{-1} e_k} \right] w_\lambda
\]
(52)
Noting that \( g_k^T (G + \lambda I)^{-1} e_k \) is scalar and multiplying (52) by \(-g_k^T\) and then adding
\[
y_k = e_k^T y
\]
leads to,
\[
y_k - g_k^T w_k(\lambda) = e_k^T y - \left[ g_k^T + \frac{g_k^T g_k^T (G + \lambda I)^{-1} e_k}{1 - g_k^T (G + \lambda I)^{-1} e_k} \right] w_\lambda
\]  
(53)

Taking the common denominator in (53) and substituting for \( g_k = e_k G \) and \( w_\lambda = (G + \lambda I)^{-1} y \) leads to,
\[
y_k - g_k^T w_k(\lambda) = e_k^T y - \left[ e_k e_k^T G (G + \lambda I)^{-1} y \frac{1}{1 - e_k^T G (G + \lambda I)^{-1} e_k} \right]
\]  
(54)

It is convenient here to define the smoother matrix
\[
H(\lambda) = G(G + \lambda I)^{-1}
\]  
(55)
and write (54) as,
\[
y_k - g_k^T w_k(\lambda) = e_k^T y - \left[ e_k^T H(\lambda) y \frac{1}{1 - e_k^T H(\lambda) e_k} \right]
\]  
(56)

Noting that \( 1 - e_k^T H(\lambda) e_k \) is a scalar and taking the common denominator leads to,
\[
y_k - g_k^T w_k(\lambda) = \left[ e_k^T \left[ I_N - H(\lambda)(I_N + e_k e_k^T) \right] y \right] \frac{e_k}{e_k^T (I_N - H(\lambda)) e_k}
\]  
(57)

Substituting (57) into (49) gives the desired result,
\[
CV(\lambda) = \frac{1}{N} \sum_{k=1}^{N} \left[ e_k^T \left[ I_N - H(\lambda)(I_N + e_k e_k^T) \right] y \right]^2 \frac{e_k}{e_k^T (I_N - H(\lambda)) e_k}
\]  
(58)
The computational burden for calculating $CV(\lambda)$ for a given $\lambda$ is now reduced to inverting a single $N \times N$ matrix $(G + \lambda I)$ to obtain $H(\lambda)$. The computational efficiency can be further improved by using the similarity transformation of the design matrix $G$:

$$G = Q \Lambda Q^T$$  \hspace{1cm} (59)$$

where the columns of the orthogonal matrix $Q$ are the eigenvectors and the elements of the diagonal matrix $\Lambda = \text{diag} [\lambda_1, \lambda_2, \ldots, \lambda_N]$ are the eigenvalues of the matrix $G$. Substituting for $G$ into the smoother matrix $H(\lambda)$ and using the orthogonality property of $Q$ leads to,

$$H(\lambda) = Q \Lambda Q^T (Q \Lambda Q^T + \lambda I)^{-1}$$
$$= Q \Lambda Q^T (Q \Lambda Q^T + \lambda I)^{-1}$$
$$= Q \Lambda (\lambda + \lambda I_N)^{-1} Q^T$$  \hspace{1cm} (60)$$

or

$$H(\lambda) = Q \begin{bmatrix}
\frac{\lambda_1}{\lambda_1 + \lambda} & 0 & & \\
0 & \frac{\lambda_2}{\lambda_2 + \lambda} & & \\
& & \vdots & \\
0 & & & \frac{\lambda_N}{\lambda_N + \lambda} \\
\end{bmatrix} Q^T$$  \hspace{1cm} (61)$$

Given the orthogonal matrix $Q$ and the diagonal matrix $\Lambda$, the evaluation of $H(\lambda)$ and hence $CV(\lambda)$ for a given $\lambda$ is reduced to the trivial task of inverting an $N \times N$ diagonal matrix. The minimisation of $CV(\lambda)$ can therefore be carried out very efficiently using any univariate minimisation method.
3.4 – An illustrative example

The following bivariate example is employed to investigate the performance of Regularisation networks on reconstructing the true underlying trend from a set of noisy data. The true underlying function was considered as,

\[
y = 100(1 - 3.3x_1 + 2.9x_1^2)\exp\left[-\left(\frac{x_1 - 0.5}{0.25}\right)^2\right] \\
       (1 - 3.3x_2 + 2.9x_2^2)\exp\left[-\left(\frac{x_2 - 0.5}{0.25}\right)^2\right]
\]  

Figure 3.5 illustrates a 3D plot of the true underlying process in the range of \(0 \leq x_1, x_2 \leq 1\). The normalisation of the input space does not create any practical limitation.

**Figure 3.5** - Three-dimensional plot of the bivariate example.
Four hundred equi-spaced training exemplars were sampled from the above three-dimensional surface. The true outputs generated from equation (62) were then contaminated with random noises drawn from a Gaussian distribution of width 0.1. The generalisation performances of the trained Regularisation networks were finally computed on a $50 \times 50$ grid over the entire training range. Figure 3.6 shows the generalisation performance of a Regularisation network with an isotropic spread of $\sigma = 0.1$ at different regularisation levels of $\lambda = 0$, $\lambda = 10^{-5}$, $\lambda = 1$, and $\lambda = 10^5$.

Figure 3.6 – Generalisation performance of a Regularisation network with $\sigma = 0.1$ at various levels of regularisation (400 equispaced exemplars).
From figure 3.6 it is evident that regularisation is crucial for filtering out the noise and capturing the true underlying function. It is also clear that an optimal level of regularisation exists which produces the best generalisation performance for a given set of observations. Small values of the $\lambda$ lead to oscillatory solutions due to the fitting of the noise, while excessively large levels of regularisation parameter over-smooth the Regularisation network predictions. The cross validation technique described in section 3.3 provides a method for selecting the optimum regularisation level based on the data alone.

In the previous example the spread was arbitrarily fixed at $\sigma = 0.1$. The choice of the spread $\sigma$ has a profound influence on the generalisation performance of the regularisation network. This is shown in Figure 3.7 which compares the optimal generalisation performances of four regularisation networks with the isotropic spreads of $\sigma = 0.01$, $\sigma = 0.1$, $\sigma = 0.5$ and $\sigma = 1$. The optimum levels of regularisation were computed by minimising the CV criterion as defined by equation (58). It is obvious that choosing an improper value for the isotropic spread of a Regularisation network leads to a poor generalisation performance. Figure 3.7 also shows that above some threshold, the influence of the isotropic spreads on the generalisation performance of the Regularisation network diminishes significantly. The major difficulty therefore is in deciding on the appropriate threshold for $\sigma$ which is highly problem dependent. Figure 3.8 illustrates the variations of the cross validation criterion with the level of regularisation for the four Regularisation networks of Figure 3.7. It is interesting to note that initially the optimum level of regularisation $\lambda^*$ increases with increasing $\sigma$ but after some certain threshold
Figure 3.7 – Generalisation performances of Regularisation network for different spreads at optimum level of regularisation.

Figure 3.8 – Variations of $CV(\lambda)$ with $\log(\lambda)$ for different isotropic spreads.
\( \lambda^* \) starts decreasing as \( \sigma^* \) increases. This behaviour suggests a potential method for selecting an appropriate value for \( \sigma^* \).

Figure 3.9 shows the variations of the optimum level of regularisation \( \lambda^* \) and the corresponding cross validation criterion \( CV(\lambda^* ) \) with the isotropic spread of the Gaussian basis functions used in a Regularisation network. The optimum level of regularisation drops after certain threshold \( (\sigma^* = 0.17) \) due to the increasing smoothing effect of their larger isotropic spreads. Figure 3.10 shows the generalisation performances of a Regularisation network at the threshold spread of \( \sigma^* = 0.17 \) for no regularisation \( \lambda = 0 \) and the optimal level of regularisation \( \lambda = \lambda^* = 2.1 \). The Regularisation network with \( \sigma^* = 0.17 \) and \( \lambda^* = 2.1 \) recovers the true underlying trend from the noisy data remarkably well.

![Figure 3.9](image-url) - Variations of optimum regularisation level and the corresponding cross validation criterion with the isotropic spreads of Gaussian basis functions.
Figure 3.10 - Generalisation performances of the Regularisation networks at the threshold spread of \((\sigma^* = 0.17)\).

The above examples show that for relatively "large" and suitably distributed data with comparatively low noise to signal ratio, the Regularisation network with optimum level of regularisation and a proper choice of the isotropic spread can provide an excellent generalisation performance. However, in many practical applications the available data are usually limited, sparse and contaminated with relatively large noise. In such cases, the training data cover only parts of input space but the fitted model is usually required to generalise well inside the entire training range. The following example investigates the performance of Regularisation networks for such limited, sparse and noisy data sets.

One hundred training exemplars were sampled at random from the three-dimensional surface of Figure 3.5 within the five distinct clusters as shown in figure 3.11. The true outputs generated from equation (62) were then contaminated with random noises drawn from a Gaussian distribution of width 0.2. The true underlying function and its noisy
samples are shown in Figure 3.12. This data set was used to explore the generalisation performance of various regularisation networks.

**Figure 3.11** – Two dimensional plot of five distinct input space clusters.

\[(N_1 = 15, N_2 = 15, N_3 = 15, N_4 = 15 \text{ and } N_5 = 40)\]

**Figure 3.12** – True surface and 100 noisy training exemplars.
Figure 3.13 shows the optimal generalisation performance obtained for four Regularisation networks with isotropic spreads of $\sigma = 0.01$, $\sigma = 0.1$, $\sigma = 0.5$ and $\sigma = 1$.

Figure 3.13 – Generalisation performances of the Regularisation network for different spreads at optimum levels of regularisation.

(100 clustered exemplars)
It is clear that the generalisation performance is very poor for small spread of $\sigma = 0.01$. The basis functions are simply too narrow and provide insufficient overlap to capture the underlying surface. The performance improves on increasing $\sigma$ but there are large differences between the true surface (Figure 3.12) and the recovered surface for the four cases shown. It is therefore interesting to see if an appropriate threshold for $\sigma$ can be found.

Figure 3.14 shows the variations of the optimum level of regularisation $\lambda^*$ and the corresponding cross validation criterion $\text{CV}(\lambda^*)$ with the isotropic spread of the Gaussian basis functions used in the Regularisation network. The figure suggests a threshold of $\sigma^* = 0.18$ and Figure 3.15 shows the generalisation performance for no regularisation ($\lambda = 0$) and optimal regularisation ($\lambda = \lambda^* = 1.58$) at $\sigma^* = 0.18$. It is clear that regularisation is essential to prevent spurious oscillation caused by over-fitting of the noisy data. More significantly, using an appropriate value of $\sigma$ and an optimal level of regularisation determined by cross validation criterion recovers the true underlying surface remarkably well from the above set of limited, sparse and noisy data.
Figure 3.14 – Variations of the optimum regularisation level and the corresponding cross validation criterion with the isotropic spread of the Gaussian basis functions.

![Graph showing variations of optimum regularisation level and cross validation criterion with log(σ)](image)

Figure 3.15 - Generalisation performance of a Regularisation networks with the threshold spread of $\sigma^* = 0.18$.

![Graphs showing generalisation performance at λ = 0 and λ = λ* = 1.58](image)
We close this discussion of Regularisation networks by giving a justification for the existence of a threshold value for the isotropic spread $\sigma$ based on an approximate measure of the degrees of freedom which can be sustained by the data. For a model with $M$ parameters representing given $N$ observations, the effective degrees of freedom is equal to $N - M$. Recalling the definition of the smoother matrix as,

$$H(\lambda) = G(G + \lambda I)^{-1}$$

For model comparisons the approximate degrees of freedom which gives an indication of the amount of fitting that $H$ does is defined as the $\text{tr}(H)$, which is the sum of the eigenvalues of matrix $H$. Figure 3.6 illustrates the variations of the optimal levels of regularisation $\lambda^*$ and the corresponding approximate degrees of freedom $df(\lambda^*) = \text{tr}(H(\lambda^*))$ with the isotropic spread of the Gaussian basis functions used in the Regularisation network.

![Figure 3.16](image.png)  

**Figure 3.16** – Variations of the optimum regularisation level and the corresponding approximate degrees of freedom with the isotropic spread of the Gaussian basis functions.
It is clear that the threshold $\sigma$ occurs at a point where the approximate degrees of freedom has a minimum. Using the threshold $\sigma$ therefore enables us to select the approximate degrees of freedom required to fit the underlying surface. Smaller $\sigma$ introduce larger degrees of freedom leading to spurious oscillations while larger $\sigma$ limits the degrees of freedom and leads to over-smoothing.

3.5 – Conclusions

This chapter was aimed exclusively at an important class of feed-forward neural networks with a single hidden layer (Regularisation network) which have a solid mathematical foundation. The basic concepts of multivariate linear regularisation theory were considered briefly and it was shown that the solution can be represented as a feed-forward network employing radial basis activation functions (RBFs). The number of neurones in the hidden layer of a Regularisation network is equal to the number of data points and the centre of each radial basis function is restricted to a particular exemplar.

In the majority of reported applications, the Regularisation network has been employed with Gaussian radial basis functions with a constant isotropic spread. The training of such networks requires the calculation of the synaptic weights, the selection of the isotropic spread and the computation of the optimal level of regularisation. The synaptic weights appear linearly and their calculation presents no major difficulty. The optimal value of the regularisation parameter, $\lambda^*$, is however highly correlated with the
isotropic spread $\sigma$, an obvious point which has surprisingly received little attention to date.

The leave one out cross validation criterion technique of section 2.6 was modified in this chapter and a new working equation was developed for the efficient computation of the optimum regularisation parameter $\lambda^*$ for a given spread $\sigma$. An illustrative example was used to clearly demonstrate the strong correlation between $\lambda^*$ and $\sigma$. A simple but significant contribution of this chapter is the development of a convenient procedure for de-correlating these parameters and selecting the optimal values of $\lambda^*$ and $\sigma^*$.

The effective degrees of freedom, $df(\lambda, \sigma)$, of a Regularisation network is a function of both the regularisation level $\lambda$ and the isotropic spread $\sigma$. A readily calculable measure of the approximate degrees of freedom of a Regularisation network was introduced which may be used to de-couple $\lambda^*$ and $\sigma$. Plotting $df(\lambda^*, \sigma)$ against $\sigma$ provides a curve which exhibits a clear minimum. This minimum is an approximate measure of the degrees of freedom which can be reasonably sustained by the noisy data set and can be used to provide the best value for the isotropic spread $\sigma^*$. The use of the effective degrees of freedom for this purpose leads to significant improvement in the performance of the Regularisation network and to our knowledge has not been previously reported.
Chapter 4

Optimisation of Generalised RBF Networks

The computational cost of a Regularisation network grows as $N^3$ where $N$ is the number of training exemplars, this is a direct consequence of the one-to-one correspondence between the number of the hidden neurones and the number of distinct data points. To ease the computational burden it is essential to break the one-to-one correspondence by using a smaller number of neurones $M << N$, such networks have been labelled as Generalised RBF networks by Poggio and Girosi [1990a&b]. The reduction in the number of neurones inevitably limits the approximating power of the network and it is then necessary to distinguish the centres of the radial basis activation functions from the exemplars. Consequently, the training of a Generalised RBF network reduces to a challenging large-scale optimisation problem which has received considerable attention over the past decade. This chapter starts with a thorough review of the literature to place the material presented below in the context of previous work.

The relevant aspects of multivariate linear regularisation theory are used to develop the working equations of a Generalised RBF network. The training of a Generalised RBF network requires the calculation of the synaptic weights and estimation of the centres and spreads of the RBFs which establish the receptive field of the neurones. Given the non-linear parameters, i.e. the centres and the spread(s), the calculation of the synaptic
weights reduces to an over-determined linear system which can be handled by a variety of efficient and stable techniques. Computationally efficient methods for establishing the optimum level of regularisation can also be developed by a combination of the leave one out cross validation criterion and generalised singular value decomposition using the procedures outlined in chapters 2 and 3.

This chapter focuses on the calculation of the optimal non-linear parameters of a Generalised RBF network, which is by no means a trivial task. A number of heuristics have been proposed to circumvent the non-linear optimisation altogether. An illustrative example is used to highlight the limited reliability of the heuristic approach and the need for a direct solution of the non-linear problem. Several procedures may be considered in order to optimise the Generalised RBF network through a succession of smaller and more manageable sub-problems. A number of techniques such as the unsupervised $k$-means clustering algorithm [Moody and Darken, 1989] and supervised Orthogonal Least Square method [Chen $et$ $al.$, 1991] have been developed to optimise the location of the centres of a Generalised RBF network with fixed spreads. In contrast, the optimisation of the spreads which determine the width and the orientation of the receptive field of the neurones, has received little attention to date. This is largely because the majority of the reported applications have concerned problems such as speech and vision recognition, classification or time series prediction with very large and dense data sets. The networks developed in such cases usually contain many neurones and the details of the receptive fields are then less important so long as the entire input domain is adequately covered. In applications involving limited and sparse data sets, such as those frequently encountered
in chemical engineering problems, the networks have a small number of neurones. The proper selection of the spreads is crucial to the performance of such small networks and is a main topic covered in this chapter. In particular, a novel sequential back-fit algorithm is developed to optimise the neurones one at a time. The new algorithm is tested against an illustrative problem and its promising performance is also demonstrated for a simple chemical engineering process.

4.1 – Literature Review

The Generalised RBF networks were initially invented in 1988 by Broomhead and Lowe [1988]. Two years later, Poggio and Girrosi provided a rigorous theoretical foundation for both Regularisation and Generalised RBF networks [Poggio and Girrosi, 1990a&b]. The last decade saw more than one thousand articles on RBF networks with the majority (~ 80%) concerned with the application of standard RBF networks to specific engineering problems. The remaining articles were related to the fundamental aspects or learning strategies of the RBF networks with special emphasis on classification or function approximation problems. This section presents a brief review of the articles dealing with the fundamental aspects of RBF networks. A concise review of the specific applications of neural networks in chemical engineering is presented in Appendix A.

Broomhead and Lowe [1988] introduced for the first time the idea of using radial basis functions as neural networks. They also suggested the idea of least square
interpolation (instead of strict interpolation) by distinguishing between the data points and the radial basis function centres, which led to the Generalised RBF networks. This reduction in the number of neurones was viewed as some sort of regularisation by Broomhead and Lowe [1988]. To preserve the linearity of the learning procedure, they only investigated the computation of the linear synaptic weights for a given set of non-linearities $\phi_j$'s, with known centres and spreads. The centres were either selected uniformly within the domain of the training data or they were selected at random from the training exemplars. No special procedure was recommended for selecting the spreads of the radial basis functions or the number of centres of the generalised RBF network. The recommended approach although innovative was far from producing the optimal RBF networks.

Moody and Darken [1989] presented two hybrid learning techniques for selecting the Generalised RBF network centres based on the self-organised $k$-means clustering algorithm. Both on-line and batch clustering techniques were able to recognise well the data clusters in the input space. However, clustering algorithms are more probability density oriented and are best suited to classification problems. The main draw back of the $k$-means clustering algorithms for function approximation problems is that the selection of the RBF centres depends only on the distribution of the input training exemplars without taking into account how the output varies with the variation of inputs. Thus, the performance of the $k$-means clustering algorithm drastically deteriorates when there is a region with sparse data but the output show high frequency variations. Moody and Darken [1989] also recommended a “global first nearest neighbour” heuristic for
selection of the RBF isotropic spreads which also has limited utility for practical function approximation applications.

Bruzzone and Prieto [1998] modified the above approach for classification problems by introducing the class-membership factor for the training data. Sohn and Ansari [1997&1998] presented another novel algorithm called scatter based clustering (SBC) which recognises well the input clusters of the training exemplars and allocates enough centres to cover all clusters. Although most of these methods are extensively applied to function approximation problems by other users, we do believe that they are more suitable for classification problems and should be applied with great caution to function approximation tasks.

Poggio and Girosi [1990a] employed the standard techniques of multivariate regularisation theory to transform the inherently ill-posed problem of reconstructing a continuous hyper-surface via Generalised RBF network for a finite set of training exemplars to a well-posed one. They introduced the Hyper Basis Functions (HyperBFs) network which as it will be shown in the next chapter is a generalisation of the Generalised RBF network of Broomhead and Lowe [1988]. Poggio and Girosi [1990a] were also the first to point out the crucial role of spreads whenever different types of inputs are present. Although the normalisation (scaling) of the input variables can be used to alleviate the need for non-isotropic spreads, the selection of appropriate spreads remains a major task in RBF networks and this problem will be discussed in more detail subsequently.
In their next fundamental article, Poggio and Girosi [1990b] considered various aspects of the single hidden layer RBF networks. They derived the so-called “Regularisation networks” as the solution of the multivariate regularisation problem. The complete solution of the Regularisation network including the global terms which lie in the null space of the stabiliser operators were discussed in more details. Various operators were derived for different radial basis functions and the regularisation approach was extended to the Generalised RBF networks. A gradient descent optimisation technique was suggested for stepwise optimisation of the (non-linear) centres and spreads and the (linear) synaptic weights of the RBF networks in the absence of regularisation. It will be shown in section 4.4 that although simultaneous optimisation of the Generalised RBF network parameters leads to promising results with a small number of centres, it is computationally demanding and encounters convergence problems for large input space dimensions due to the curse of dimensionality.

Thau [1991] reviewed the above article of Poggio and Girosi [1990b] and employed the generalised RBF networks with constant isotropic spreads for a number of object recognition applications. The reported results indicated that using the gradient descent technique for selection of the RBF centres substantially improves the performance of the RBF network. Our investigations also verify this issue, however, the proposed gradient descent algorithm is extremely slow for practical complex applications. Furthermore, simultaneous optimisation of the entire RBF network parameters may lead to redundant centres especially when the number of centres is relatively large. The redundancy can be
alleviated by resorting to the singular value decomposition (SVD) technique for computation of the linear weights instead of the recommended gradient descent technique.

Chen et al. [1991] presented the Orthogonal Least Square (OLS) algorithm for the selection of the generalised RBF network centres with constant isotropic spreads in the absence of regularisation. The initial regressor vectors can be constructed using all training exemplars as the centres of the Generalised RBF network. Classical Gram-Schmidt orthogonalisation technique was then employed to select the “best” centres that maximise the increment to the explained variance or energy of the desired output. Due to the orthogonalisation procedure, the OLS algorithm leads to a set of over-determined and well-conditioned linear equations for computing the linear weights. A detailed exploration of the OLS algorithm will be considered in section 4.3.1.3.

Sherstinsky and Picard [1996] argued against the efficiency of the OLS algorithm and provided some examples where the OLS algorithm did not produce the smallest possible network for a given approximation error. Our investigations show that the method is quite fast and performs better than random selection of the centres. However, The constructed network usually requires a large number of centres to provide an acceptable training performance, which leads to poor generalisation for noisy data sets. To overcome this difficulty, Chen [1995] and Chen et al. [1996] combined subset selection via the OLS algorithm with some sort of zero order regularisation for multiple output RBF networks. In this article, a Fast version of Regularised Orthogonal Least Square (FROLS) was
introduced which provided better generalisation performance than the simple OLS algorithm but has the following limitations.

The FROLS algorithm uses pre-specified isotropic spreads and cannot be easily adapted for the optimisation of non-isotropic spreads. FROLS is also limited to the zero order regularisation which is often inferior to other regularisation techniques such as the second order method of Bishop [1991]. FROLS has more in common with the ridge regression version of RBF networks presented by Orr [1998]. Chen et al [1995] also suggested a two layer learning algorithm to combine a genetic algorithm with FROLS for the simple univariate example of $F(x) = \sin(2\pi x); 0 \leq x \leq 1$. At the lower level, the FROLS algorithm was employed to find the RBF centres and the corresponding linear weights, while the genetic algorithm was used at the higher level to compute the required regularisation parameter and the isotropic spreads of the Gaussian basis functions.

Bishop [1991] presented a second order smoothing technique for filtering out the noise from a set of univariate noisy data and improving the generalisation performance of the Generalised RBF networks. Our investigation shows that the regularisation method of Poggio and Girosi [1990a&b] is more robust and leads to better results for complex applications. Bishop [1991] minimised the test error criteria for calculation of the optimal regularisation parameter. The leave one out cross validation technique [Golub, 1996] seems a more efficient way of finding the optimum regularisation level and employs both training and test exemplars in the training phase.
Wong [1991] argued against the suitability of the commonly used Gaussian radial basis function networks for predicting chaotic time series and believed the reason that other researches (such as Broomhead and Lowe [1988] and Moody and Darken [1989]) have generally reported good result is that their training data sets may not contain enough high frequencies due to insufficient sampling of the input space. We believe that using a sufficient number of Gaussian basis functions with appropriate non-isotropic spreads can dramatically alleviate this problem.

Leonard et al. [1992] derived the Validity Index network (VI net) from radial basis function networks which fits functions and calculates the confidence intervals for its predictions, indicating local regions of poor fit and extrapolation. The underlying RBF network of the VI net employed the nearest neighbour clustering technique [Moody and Darken, 1989] for selection of the RBF centres and uses the $P$ nearest neighbour heuristic for estimation of the RBF network spreads. The optimum values of $P$ and $H$ (number of hidden neurones) were computed via the $S$-fold cross validation technique [Schenker and Agarwal, 1996]. Our investigations show that the above training algorithm performs well for simple applications. However, for complex applications with noisy data, the above algorithm should be equipped with more efficient techniques for selection of the RBF centres and the corresponding spreads as well as using regularisation technique to filter out the noise. In the VI net, additional neurones were added to the output layer of the underlying RBF network to compute the confidence limits of the model predictions and the local density of the training data. The predicted confidence limits provide valuable information about the validity of the underlying RBF network predictions indicating the
local regions of poor fit. While, the local density neurones identify the extrapolation regions outside the training data clusters.

Tao (1993) recommended the normalisation of the hidden layer neurones outputs for smoother interpolation and presented a simple back-propagation learning algorithm for recursive selection of the RBF centres with isotropic spreads and the corresponding linear weights. An isotropic basic spread was introduced as the averaged distance between the underlying kernel and the centres of its two nearest kernel neighbours. Various percentages of this basic spread were employed for both normalised and unnormalised RBF networks in the absence of regularisation. The reported results indicated that the normalisation process decreases the dependency of the RBF networks on the choice of spreads. To increase the chances of seeding the peaks for the random starting points of the k-means clustering algorithm [Moody and Darken, 1989], higher chances of getting seeded were designated for points with higher outputs, which does not seem a sensible approach for a general hyper-surface.

Poggio et al. [1993] proposed the Generalised Regularisation Networks (GRN) as an extension of the Regularisation Network (RN) for a broad class of approximation schemes such as Additive Models (AM) and Projection Pursuit Regression (PPR) technique. They considered new stabilisers other than the common radial stabilisers. The translationally and rotationally invariant radial stabiliser operators which lead to the RBF networks used a weighted norm in the form of
\[ \| x_i - t_j \|_{C_j} = (x_i - t_j)^T C_j^T C_j (x_i - t_j) \] with the norm-weighting matrix \( C_j \). While,
for "additive stabilisers" a linear transformation of the inputs in the form of $\mathbf{c}_j \mathbf{x}_i$ ($\mathbf{c}_j$: projection vector) replaced the weighted norm and lead to the Projection Pursuit Regression (PPR) method [Freedman and Stutzle, 1981].

Kuo and Melsheimer [1994] compared the performance of the "global two nearest neighbour" heuristic with the estimation of the optimal isotropic spreads of the RBF networks via Genetic algorithm in the absence of regularisation. The RBF centres were selected using $k$-means clustering algorithm [Moody and Darken, 1989] and the linear weights were computed via OLS algorithm [Chen, Cowan and Grant, 1991]. Estimations of the open loop responses of two CSTR and batch reactors were considered as two practical application examples. They reported that the optimisation of the RBF spreads (even isotropic) leads to essentially less complex (fewer centres) and tremendously more accurate RBF networks. We shall demonstrate in this chapter that full optimisation of the spreads coupled with the regularisation technique produces even better results.

Girosi and Chan [1995] proposed a method to incorporate some prior knowledge (other than smoothness) such as radial symmetry of the model function ($f(\mathbf{x}) = f^* (\|\mathbf{x}\|)$) directly in the supervised learning technique. The reported results show that the inclusion of the available a priori information about the form of the final solution in the formulation of the RBF networks, substantially reduces the generalisation error for a given set of training exemplars. In an alternative approach reviewed in Chapter 135
2, such \textit{a priori} information can be included as constraints in the regularised merit function to force the solution to have the desired \textit{a priori} property.

Maffezzoni and Gubian [1995a] proposed a novel self-organised learning algorithm for placement of the RBF network centres which reduces to the standard \textit{k-means} clustering algorithm as one of its extremes as $\beta \to 0$. The reported results indicate that the new algorithm with non-zero $\beta$, leads to substantially different positions of the centres compared with standard \textit{k-means} clustering algorithm. The overall performances of the RBF networks were not investigated in the article. Our investigations verify the reported results but indicate that the new method provides no definite improvement on the overall performance of the RBF networks over the standard \textit{k-means} clustering algorithm for multivariate function approximation purposes.

In their next article, Maffezzoni and Gubian [1995b] proposed a three step learning strategy for training the RBF networks in the absence of regularisation. In the first step, the RBF centres were selected via modified \textit{k-means} clustering algorithm using a special entropy cost function. Next the isotropic spreads of each centre were computed as the average distance of the training exemplars form that centre. Finally, the synaptic weights were computed via back-propagation technique. To avoid ill-conditioning problems, Maffezzoni and Gubian [1995b] imposed an upper bound on the magnitude of the synaptic weights. We believe that using the singular value decomposition (SVD) technique with an appropriate threshold as described in Chapter 2 provides a more elegant solution to the \textit{ill-conditioning} problem. These authors also pointed out that by
increasing the overlap between the adjacent centres (larger isotropic spreads), the model response for a particular point depends on many centres which leads to a better generalisation performance for noisy data sets. Although we agree that larger spreads lead to smoother predictions, the regularisation approach provides a more direct way of dealing with noisy data sets. Furthermore, as illustrated in Chapter 2, larger spreads in general amplify the ill-conditioning problem.

Bostel [1995] combined the gradient descent optimisation technique with perturbation methods for training the RBF networks in the absence of regularisation. First, the linear weights were updated via conventional Least Mean Square (LMS) algorithm for a set of initial centres and spreads. Then the RBF centres and the corresponding diagonal spreads were computed via a perturbation method. The recommended technique is relatively fast for simple examples and does not require gradients for updating the network centres and spreads. Constant step-lengths were used for the perturbation technique, which may affect the convergence for more complex problems. An efficient step-size control mechanism seems more beneficial. Furthermore, the perturbation technique employed for updating the centres and the spreads applies only to a special class of Gaussian basis functions and can not be easily extended to other types of Generalised RBF networks.

Cheng and Lin [1995] compared three different orthonormalisation techniques for simultaneous multidimensional scaling and rotation of the radial basis functions. They considered a single ten dimensional target Gaussian centred at origin with different scalings in different dimensions [3,7.5]. Eleven selected axis pairs were rotated by 45
degrees. The starting basis function was fixed at the origin and initialised with scaling factors of 5. and zero degrees rotation angles in all dimensions. Even, for this relatively simple example, the Newton's method failed to converge while the Gram-Schmidt and Gradient descent techniques required large number of iterations for acceptable convergence. Our investigations show that the recommended approach encounters severe convergence problems when dealing with more than one centre.

Tsai et al. [1996a&b] attempted to keep the form of the sigmoid activation function used with MLP but modified its argument from a linear projection to a radial one. They argued that the conventional Gaussian basis functions used in RBF networks are unable to capture the constant values. As it will be presented in Chapter 5, the RBF network representing the complete solution of the multivariate regularisation problem are not only capable of approximating constant values via bias but they are able to approximate any global trend by employing the appropriate non-zero polynomials that live in the null space of the self-adjoint radial stabilising operator $D$ [Courant and Hilbert, 1970].

Tsai et al. [1996a&b] also employed robust objective functions instead of the conventional Least Square (LS) technique to deal with the outliers problem. As illustrated in Chapter2, the use of robust probability density functions such as Lorentzian rejects the outliers in the favour of a smoother fit. On the other hand, the linearity of the problem (even for linear synaptic weights) is violated on using robust objective functions which amplifies the computational burden. Our investigations show that use of the LS criterion with an appropriate regularisation technique has a similar effect on rejecting the outliers,
while preserving the linearity of the problem for the synaptic weights. Tsai et al. [1996a&b] also recommended a memory queue to prevent the repetitive selection and deletion of the outliers as centres. The selected centres with small synaptic weights were deleted in a pruning step. This is conceptually troubling, since a combination of two or more centres with large weights of opposite sign may also be redundant. An effective pruning algorithm should recognise this problem and delete such redundant combination of centres simultaneously which is by no means a trivial task.

Zhang et al. [1996] presented a novel clustering algorithm for selection of the RBF network centres which considered both inputs and output. An augmented vector was constructed using scaled inputs and outputs. This augmented vector was utilised in the k-means clustering algorithm to select the RBF centres in a recursive manner. Isotropic spreads were used for the RBF kernels. A singular value decomposition (SVD) based recursive least square optimisation technique was employed to avoid the ill conditioning problem in the calculation of the RBF synaptic weights. The recommended method was reported to be fast and lead to smaller training phase mean squared error for a univariate example compared with the standard k-means clustering algorithm. The generalisation performance of the new method was not discussed in the article.

Russel and Fausett [1996] compared the RBF and sigmoidal back-propagation (multilayer perceptron) network. Different non-linearities with isotropic spreads were used in the RBF networks in the absence of regularisation. Gradient descent optimisation technique with constant step-size was used for simultaneous updating of the centres,
spreads and the synaptic weights of the RBF networks. The reported results for a number of relatively simple examples show that RBF networks require less neurones to attain the same training error as the sigmoidal network (SN). Furthermore, for a given relatively large number of neurone, the RBF network (with gradient descent learning) requires more CPU time than sigmoidal network (SN) but leads to a dramatically smaller final training error. The article does not provide any information about the generalisation capabilities of the RBF or SN networks.

Kiernan et al. [1996] presented the simple but interesting idea of partitioned k-means clustering algorithm that leads to a variety of isotropic spreads for the RBF networks. The recommended method selects at any stage one half of the undefined nodes independently via k-means clustering algorithm [Moody and Darken, 1989] and computes the corresponding spreads as the mean distance to the nearest $j$ centres. The process terminates when the number of undefined centres drops to or below $j$. Our investigations show that this method may lead to a variety of isotropic spreads that have both desirable global and local features. However, the selected centres in the successive steps are unnecessarily close to each other which deteriorates the overall performance of the RBF networks.

Gomm and Williams [1997] extended the recursive orthogonal least square (ROLS) method of Bobrow and Murray [1993] for training multiple input multiple output (MIMO) RBF networks with thin plate spline (TPS) basis functions. The reason for selecting TPS was stated that "unlike possible basis functions including the Gaussian, this
basis function (TPS) does not require the choice of an additional width parameter”. This statement does not seem correct, because in general, a thin plate spline (TPS) basis function must be expressed as 
\[ \phi(x_i, t_j) = \left( \frac{\| x_i - t_j \|}{\sigma_j} \right)^{2n} \ln\left( \frac{\| x_i - t_j \|}{\sigma_j} \right), \]
which not only includes the width parameter \( \sigma_j \) but has an additional power factor \( n \). The main advantage of the conventional OLS algorithm [Chen et al., 1991] is that it automatically provides the network size for a given error criteria. However, it is relatively slow for extremely large data sets. The ROLS algorithm selects the pre-specified number of RBF centres via *k-means* clustering algorithm and uses QR decomposition for updating the linear synaptic weights. ROLS is faster than OLS especially for large data sets, but it does not provide the optimum network size. We do believe that using the singular value decomposition (SVD) technique is more advantageous than ROLS for the computation of the linear synaptic weights.

Karayiannis [1997] replaced the exponential Gaussian function by a generator function in the form of 
\[ \phi(x) = (g_0(x))^{-m}, \]
where \( g_0(x) \) is any monotonic increasing or decreasing generator function. A simple algorithm based on the gradient descent optimisation technique was recommended to compute the linear weights and the reformulated RBF centres for multiple input multiple output (MIMO) networks. Fixed and equal learning parameters (\( \eta \)) were considered for updating both linear weights and the non-linear centres which may not be the best idea in many practical applications. The recommended algorithm emphasises on optimising the location of the centres but the importance of the widths of the radial functions has been neglected. The reported results
indicate that "the proposed algorithm guarantees fast learning and very satisfactory
generalisation ability". In another similar article [Karayiannis, 1998], Karayiannis
proposed batch and online learning algorithms for training a similar reformulated RBF
network.

Hong and Billings [1997] presented a fast backward elimination algorithm for
pruning the RBF networks in the absence of regularisation. In this approach, all training
exemplars were considered as the RBF centres or the initial centres were selected via
self-organised methods such as \(k\)-means clustering algorithm [Moody and Darken, 1989].
QR decomposition of the \(N \times M\) design matrix (with isotropic spreads) was then
performed and Givens transformation was employed to construct the new QR
decomposition resulting from permutation of \(j^{th}\) column \((j=1,M-1)\) of the design matrix
with the last column. The centre which produced the minimum increment of error
variance (IEV) was pruned and the loop continued until the value of some prediction risk
indicator parameter \((\varepsilon)\) reached the prescribed threshold. As already mentioned, a
combination of two or more RBF centres with large but opposite sign synaptic weights
may be redundant. An effective pruning algorithm should recognise this and delete such
redundant combination of centres simultaneously. Deletion of one of these centres affects
the performance of the RBF network significantly while the simultaneous removal of the
entire redundant combination of centres has essentially no effect on the overall
performance. The pruning algorithm of Hong and Billings [1997] deletes one centre at a
time and is therefore unable to distinguish multiple redundant centres.
Lowe [1997] introduced a method for characterising the degrees of freedom of the RBF networks by exploiting a relationship to the theory of linear smoothers [Hastie and Tibshirani 1990]. Traditionally, the number of hidden units or the total number of adjustable parameters of the RBF networks was considered as the measure of network’s complexity, which is clearly naive. The effective degrees of freedom (\( df \)) provides a more sensible measure. For linear smoothers [Hastie and Tibshirani 1990], the trace of the smoother matrix is usually selected as the effective degrees of freedom. Lowe believes that “empirical evidence suggests that the precise choice of the kernel width is relatively unimportant”. Our investigations verify this issue as far as it concerns the magnitude of the degrees of freedom (\( df \)). However, this does not mean that the spreads have no effect on the overall performance of the RBF network. As we know, locations of the centres have a crucial effect on the training and generalisation performances of the RBF networks. In practice, the number of centres determines the maximum degrees of freedom and the location of the RBF centres have a minor effect on the magnitude of degrees of freedom. For most practical applications, the effective degrees of freedom is also highly related to the value of regularisation parameter (\( \lambda \)). In fact, the choice of regularisation parameter specifies the effective degrees of freedom (or \textit{vice versa}) for a given network architecture. Surprisingly, Lowe [1997] did not consider regularisation even for his simple noisy sine wave example.

Zhang and Morris [1998] proposed an orthogonal algorithm for sequential selection of the neurones of single hidden layer neural networks. The centres of the RBF network were selected in such a manner that at each step, the new information introduced by
adding a centre was orthogonal to the space spanned by the previous ones. The Gram-Schmidt orthogonalization procedure coupled with gradient-descent optimisation technique was employed to compute the RBF centres and the corresponding isotropic spreads at each step. Levenberg-Marquardt optimisation technique was actually used instead of the gradient descent method to accelerate the convergence. Least square regression formula was employed for calculation of the RBF linear weights and the Akaike information criterion (AIC) was used to compromise between the model accuracy and the model complexity. Using the effective degrees of freedom as described in the previous chapter may provide a better and more readily understood criterion for this purpose. A special kind of regularisation similar to the ridge regression was employed to improve the generalisation performance of the constructed RBF network. A more detailed consideration of the promising sequential algorithm of Zhang and Morris [1998] will be presented in section 4.5.2.

McLoone and Irwin [1998] compared various optimisation techniques for simultaneous optimisation of linear and non-linear parameters of RBF networks in the absence of regularisation. They reported that “no optimisation method can produce guaranteed results for global simultaneous optimisation of the RBF network parameters”. This is because non-linear optimisation of the RBF networks is an inherently ill-conditioned problem due to the curse of dimensionality and their sensitivity to the gradients of the network parameters. Our observations verify these findings but as will be shown subsequently, novel sequential back-fit learning strategy presented in this chapter can alleviate the so-called curse of dimensionality significantly. McLoone and Irwin
[1998] also reported that isotropic (spherical) spreads provide better performance than diagonal (elliptical) spreads. Our investigations are at odd with this observation and we believe that the above statement is problem dependent and highly sensitive to the choice of the diagonal spreads. We emphasis that in general, flexible diagonal spreads provide better performance than even flexible isotropic spreads. However, we do agree with McLoone and Irwin [1998] argument that “the widths are the major source of ill-conditioning in RBF networks”. McLoone and Irwin [1998] also proposed an efficient hybrid learning algorithm for training RBF networks by separating the linear and non-linear network parameters. The non-linear parameters (centres) were computed via non-linear optimisation techniques while the linear parameters were computed using singular value decomposition (SVD) technique. As they reported, the hybrid method provides better conditioning of the Hessian matrix and leads to superior convergence and performance when compared with simultaneous optimisation of both linear and non-linear parameters.

Yingwei et al. [1998] presented a version of Resource Allocation Network (RAN) algorithm which combines the growth criterion of RAN with a pruning strategy based on the relative contribution of each hidden unit to the overall network output. The proposed algorithm starts with no hidden neurones and grows by allocating new hidden units based on the novelty in the observations which arrive sequentially. The decision as to whether a new (i \textsuperscript{th}) exemplar has novelty depends on the following conditions,

\begin{align}
\| \mathbf{x}_i - \mathbf{t}_{lr} \| &> e_i \label{eq:1} \\
| e_i | &= | y_i - f(x_i) | > e_{\min} \label{eq:2}
\end{align}

145
where, $t_{ir}$ is the nearest centre to $x_i$, $e_i$ and $e_{\text{min}}$ are two prescribed thresholds. If the above two conditions are satisfied then a new centre will be added. The weight, centre and the isotropic spreads of the new ($j^{th}$) centre are computed as $w_j = e_i$, $t_j = x_i$ and $\sigma_j = \kappa \left( \| x_i - t_{ir} \| \right)$. Where $\kappa$ is an overlap factor that determines the amount of overlap of the hidden units in the input space. They recommended that larger spreads should be selected initially and by increasing the number of observations, more basis functions with smaller widths should be chosen to fine tune the approximation. The case where the new exemplar has no novelty was simply dealt with by adjusting the parameters of the existing network without adding a centre.

In this approach [Yingwei et al., 1998], some of the added neurones may end up with less contribution to the network output. The proposed algorithm detects these inactive neurones and removes them in the course of training. The recommended algorithm provides acceptable performance for relatively simple problems. In more complex applications, the above method usually leads to sub-optimal placement of centres because it restricts the RBF centres to the position of data points. Evidently, it would be desirable to position more centres (with maybe smaller spreads) in regions where the output is changing sharply and fewer centres in the flat regions. Using regularisation techniques coupled with non-isotropic spreads may also lead to further improvements especially in the presence of noisy data.

Orr [1998] presented a simple technique based on testing a number of trial values for isotropic spreads of the RBF network and selecting the one associated with the smallest
value of Generalised Cross Validation (GCV) criterion. All training exemplars were considered as centres and all centres had equal isotropic spreads. He introduced the idea of using similarity transformation to speed up the computation of the generalised cross validation (GCV) criterion. The GCV criterion was selected to optimise the value of the ridge regression parameter. As emphasised in the article, GCV criterion suffers from multiple minima even for the toy problem presented in the paper. Our investigations show that using standard cross validation (CV) technique [Golub, 1996] instead of GCV leads to a unimodal cost function at least for the toy problem presented in the article. Clearly, the ridge regression technique which penalises large synaptic weights is not as robust as the regularisation technique of Poggio and Girosi [Poggio and Girosi, 1990]. However, the latter method requires more advance methods such as generalised singular value decomposition (GSVD) technique coupled with cross validation (CV) to speed up the automatic selection of the optimal regularisation parameter.

Heims and Heuvelen [1998] extended the idea of normalised RBF networks [Tao, 1993] for multiple output function approximation problems. Self organised clustering algorithm of Moody and Darken [1989] was employed to specify the locations of the RBF centres and the values of the corresponding isotropic spreads. Moore-Penrose pseudo inverse \( \Phi^+ \) of the non-square, non-symmetric design matrix \( \Phi \) was used to compute the optimal weight vector \( w \) [Broomhead and Lowe 1988, Golub, 1996]. Tao [1993] reported that normalisation of the outputs of the RBF neurones lead to a smoother overall prediction, and provides better generalisation performance. Our investigations also verify this issue for the simple univariate example considered by Tao [1993].
However, more advanced techniques such as multivariate regularisation are required to provide an acceptable generalisation performance for complex multivariate applications.

The above literature review summarised the fundamental articles on learning strategies of the RBF networks with special interest in function approximation problems. We may conclude that enormous interest has been focused on RBF networks in the last decade and different aspects of these networks have been intensively studied from various points of views.

4.2 - Generalised Radial Basis Function Networks

A major drawback of the Regularisation network is the one-to-one correspondence between the number of the input exemplars $N$ and the number of hidden neurones. The estimation of the linear synaptic weights demands the inversion of an $N \times N$ matrix with a computational cost proportional to $N^3$ and becomes prohibitive for large $N$. The likelihood of ill-conditioning is also in general higher for larger matrices. The selection and optimisation of the widths of the radial basis functions for a regularisation network also presents major difficulties. The number of width parameters to be tuned is $N$ for a Regularisation network with isotropic spreads. For an input vector of $x \in \mathbb{R}^p$, the number of width parameters increases to $N \times p$ for a network with diagonal spreads and to $N \times p \times p$ for a network with full norm weighting matrices. For modest values of $p = 4$ and $N = 200$, the full optimisation of the spreads for a Regularisation network reduces to a large-scale non-linear optimisation involving 3200 variables. In order to
ease the computational burden, it is necessary to break the one-to-one relationship between the number of exemplars and the number of neurones. The basic idea is to search for a sub-optimal solution in a lower dimensional space (with respect to the number of neurones) that approximates the Regularised network solution of the previous chapter.

The approximate solution $\tilde{F}(x)$ is expanded on a finite basis,

$$\tilde{F}(x) = \sum_{j=1}^{M} w_j \phi_j(x)$$

(3)

where the $\phi_j(x)$'s are a new set of linearly independent basis functions, and the $w_j$'s constitute a new set of synaptic weights. With radial basis functions in mind we may choose,

$$\phi_j(x) = G(||x - t_j||), \quad j = 1,2,...,M$$

(4)

with the set of centres $[t_j, j = 1,2,...,M]$ either pre-specified or determined during the learning process. Note that the Regularisation network is recovered by setting $M = N$ and $[t_j = x_j, j = 1,2,...,N]$. Restricting ourselves to radial basis functions, the approximate solution can be expressed as,

$$\tilde{F}(x) = \sum_{j=1}^{M} w_j G(||x - t_j||)$$

(5)

The network that represents the approximate solution (5) is usually referred to as the Generalised RBF network and is shown schematically in Figure 4.1.
Given the centres \( t_j \)'s (and any other non-linear parameters that enter the argument of the Green's functions) the determination of the linear weights can proceed as follows. We start by defining a regularisation functional,

\[
\min_{\tilde{F}} \mathcal{S}(\tilde{F}) = A(\tilde{F}) + \lambda B(\tilde{F}) \\
= \frac{1}{2} \sum_{i=1}^{N} \left( y_i - \sum_{j=1}^{M} G(x_i, t_j) w_j \right)^2 + \frac{1}{2} \lambda \int_{\mathcal{R}^p} |\tilde{D} [\tilde{F}(x)]|^2 dx \tag{6}
\]

The first term \( A(\tilde{F}) \) representing the fidelity to the observed data can be expressed as,

\[
A(\tilde{F}) = \frac{1}{2} (y - Gw)^T (y - Gw) \tag{7}
\]

where \( y = [y_1, y_2, \ldots, y_N]^T \), \( w = [w_1, w_2, \ldots, w_M]^T \) and \( G \) is the \( N \times M \) Green's matrix with elements,

\[
G_{ij} = G(x_i, t_j), \quad i = 1, \ldots, N \text{ and } j = 1, \ldots, M \tag{8}
\]
Note that unlike the Regularisation network, the Green matrix \( G \) is no longer square and symmetric. The second term \( B(\bar{F}) \) incorporating our \textit{a priori} knowledge or belief about the true underlying function can be expanded as follows:

\[
B(\bar{F}) = \int_{R^P} D[\bar{F}(x)] D[\bar{F}(x)] dx
\]
\[
= \int_{R^P} \left( \sum_{j=1}^{M} \left( G(x, t_j) w_j \right) \right) \left( \sum_{k=1}^{M} w_k \delta(x - t_k) \right) dx
\]
\[
= \sum_{j=1}^{M} \sum_{k=1}^{M} G(t_k, t_j) w_j w_k
\]
\[
= w^T G_0 w
\]

Note that in the second line of (9) we have made use of the definition of the adjoint operator (see Chapter 3) and in the third line we have used the fact that the Green's function \( G(x, t_k) \) satisfies the partial differential equation \( D^*D[G(x, t_k)] = \delta(x - t_k) \).

The matrix \( G_0 \) is an \( M \times M \) matrix with elements,

\[
(G_0)_{ij} = G(t_i, t_j), i = 1, ..., M \text{ and } j = 1, ..., M
\]

Using equations (7) and (9) the regularisation functional (6) may be expressed as:

\[
\min_{w} \mathcal{S}(\bar{F}(w)) = \frac{1}{2} (y - Gw)^T (y - Gw) + \frac{1}{2} \lambda w^T G_0 w
\]

Applying the necessary optimality conditions \( \partial \mathcal{S}(w) / \partial w = 0 \) it can be easily shown that the minimising weights satisfy the linear system,

\[
\begin{pmatrix} G^T G + \lambda G_0 \end{pmatrix} w_\lambda = G^T y
\]

The regularised solution of the Generalised network is therefore given by,
\[ F_\lambda(x) = \sum_{j=1}^{M} G(x, t_j) w_{\lambda j} \tag{13} \]

with the optimal weights determined by the linear system (12). The influence of the regularisation parameter \( \lambda \) is embedded in the linear synaptic weights. As expected, the approximate solution of the Generalised RBF network coincides with that of the Regularisation network for the special case \( M=N \) and \( t_j = \bar{x}_j, j = 1, \ldots, N \). In this case \( G_0 = G \) is a square, symmetric, and non-singular matrix and multiplying equation (12) by \( G^{-1} \) leads to \( (G + \lambda I_N)w_A = y \), which was derived in the previous chapter for the optimal synaptic weights of the Regularisation network.

The optimum level of regularisation can be computed using the leave one out cross validation (CV) criterion described in Chapter 2. The only difference is that the matrix \( \Psi \) required by the generalised singular value decomposition (GSVD) technique is no longer known explicitly and must be obtained from the similarity transformation of the matrix \( G_0 \),

\[
G_0 = Q \Lambda Q^T = (\Lambda^{1/2}Q^T)^T(\Lambda^{1/2}Q^T) = \Psi^T \Psi \tag{14}
\]

That is \( \Psi = \Lambda^{1/2}Q^T \) where, the columns of the orthogonal matrix \( Q \) are the eigenvectors and the elements of the diagonal matrix \( \Lambda \) are the eigenvalues of matrix \( G_0 \).
4.3 - Design and Training of Generalised RBF Networks

In this section we give a brief review of the important issues which arise in the design and training of Generalised RBF networks and have a direct bearing on their performance. The first issue which is readily answered is the number of hidden layers. Radial basis function networks with multiple hidden layers can be easily constructed and a few applications have been reported [Craddoc and Warwick, 1996; Hirasawa, Matsuoka, Ohbayashi and Murata, 1997]. Such networks are, however, totally non-linear and lose the “best approximation” property enjoyed by single hidden layer RBF networks which are linear in the synaptic weights and therefore have a unique optimal solution. There is therefore little advantage in using multiple hidden layer RBF networks and such networks are not pursued further in this study.

The second issue concerning conventional single hidden layer RBF networks is the number of neurones in the hidden layer. The practical objective is to produce the network of smallest size which meets the required recall and generalisation performance. This is a problem in model selection and a variety of trial and error procedures have been proposed which may be divided into two groups. The first group involves training networks of different size and then establishing the sensitivity of the various networks by a posteriori sensitivity analysis and pruning techniques (see Scheneker and Agarwal [1996] and Reed [1993] for a review). In the second approach an attempt is made to build up the optimal network sequentially, that is by adding the best neurones one at a time. Chen et al [1991] have proposed a procedure based on the orthogonal least squares technique and Zhang and Morris [1998] have made significant contributions to sequential learning techniques.
A novel sequential back-fit training algorithm with promising results is developed in this chapter and will be presented in section 4.5.2.1.

For a single hidden layer Generalised RBF network with a given number of neurones we must address the difficult problem of obtaining the optimal network parameters. The parameters can be divided into two groups: the synaptic weights which appear linearly and the parameters appearing in the argument of the non-linear activation functions of the neurones. In the case of RBF networks, the non-linear parameters are the centres $t_j$'s, which control the locations, and the variance-covariance matrices $\Sigma_j$'s, which control both the orientation and the widths of the receptive fields of the neurones. Given the non-linear parameters, the determination of the optimal synaptic weights reduces to the solution of an over-determined linear system of equations and selecting the appropriate level of regularisation which is fairly straightforward. It is the proper selection of the non-linear parameters which ultimately reduces to an extremely challenging multivariate non-linear optimisation problem.

4.4 – Independent selection of the Generalised RBF network spreads and centres

The centre and spreads of an RBF appear simultaneously and the adjustment of either alters the receptive field of a neurone. The training of the non-linear parameters of an RBF network can therefore be considered as:
a) independent tuning of the spreads for given centres
b) independent tuning of the centres for given spreads, or
c) simultaneous tuning of centres and spreads.

A number of tuning heuristics have been suggested for dealing with cases (a) and (b) and their performances for an illustrative example will be explored in this section. The simultaneous training of the centres and spreads is a complex optimisation problem and will be deferred to section 4.5.

4.4.1 — A priori selection of spreads for given centres

A number of heuristics have been proposed for the selection of the Generalised RBF networks with specified centres. Haykin [1999] suggests the following heuristic for isotropic spreads of the Generalised RBF network,

\[ \sigma = \frac{d}{\sqrt{2M}} \]  

(15)

where \( M \) is the number of hidden neurones and \( d \) is the maximum distance between the centres. Moody and Darken [1989] proposed the global first nearest neighbour heuristic as,

\[ \sigma = \langle \Delta t_{\alpha\beta} \rangle \]  

(16)

where \( \Delta t_{\alpha\beta} \) is the Euclidean distance between centre \( \alpha \) and its nearest neighbour \( \beta \) and \( \langle \rangle \) indicates the global average over all such pairs. Leonard et al. [1992] used a \( P \) nearest neighbour heuristic for the selection of the RBF spreads. The optimum values of
$P$ and $M$ were computed via the $S$-fold cross validation technique [Schenker and Agarwal, 1996]. Kuo and Melsheimer [1994] compared the performance of the “global two nearest neighbour” heuristic with the estimation of the optimal isotropic spreads of RBF networks via Genetic algorithm in the absence of regularisation. Tao [1993] introduced the isotropic basic spread as the averaged distance between the centres and the centres of its two nearest neighbours. Various percentages of this basic spread can be used. In a similar approach, Yingwei et al. [1998] used a fraction of the distance between each centre and its nearest neighbour as the isotropic spread of that centre. Maffezzoni and Gubian [1995b] proposed that the isotropic spreads of each centre should be computed as the average distance of the training exemplars form that centre. Kieman et al. [1996] computed the isotropic spread of each centre as the mean distance of that centre to its nearest $j$ centres.

It is also possible to develop simple heuristics for a network with diagonal rather than isotropic spreads. As a simple procedure, the spreads in each dimension maybe selected as the standard deviation of the training exemplars,

$$
\sigma_{j,p} = \sqrt{\frac{\sum_{i=1}^{N}(x_{i,p} - \bar{x}_p)^2}{N}} \quad \text{for} \quad j = 1, 2, \ldots, M
$$

(17)

This heuristic leads to different values for the spreads in each input dimension but equal spreads are assigned to all centres for a given input dimension. As an alternative for a network with specified centres, the spreads in each input dimension can be taken as,

$$
\sigma_{j,p} = \sqrt{\frac{\sum_{i=1}^{N}(x_{i,p} - t_{j,p})^2}{N}} \quad \text{for} \quad j = 1, 2, \ldots, M
$$

(18)
This results in different values of $\sigma_{j,k}$ for different centres.

For all of the above heuristics the centres are assumed given and the spreads are computed once and then kept fixed during the calculation of the synaptic weights. It is of course also possible to keep the centres fixed and perform a stepwise or simultaneous optimisation of the spreads and the synaptic weights. We shall have more to say on this in section 4.5, which considers the full optimisation of the centres, spreads and synaptic weights.

### 4.4.2 – A priori and supervised selection of centres for given spreads

Given the spreads of the radial basis functions, the centres of a Generalised RBF network can be chosen using an unsupervised (self-organised) procedure which does not rely on the measured output values. Well-known unsupervised procedures for centre selection include:

- **Pre-specified centres**: the network centres are placed *a priori* via some simple heuristic. For example they are evenly distributed across the input domain or they are selected at random from the training exemplars. Once selected, the centres remain fixed through the rest of the learning process.

- **Self-organised selection of the centres**: the optimum locations of the network centres are selected based on the distribution of the data in the input space regardless of the measured output values. The standard *k*-nearest-neighbour Clustering Algorithm is a major example of this approach. Once selected, the centres remain fixed through the rest of the training.
In a more sophisticated approach we may consider the supervised selection of the centres. The optimum positions of the network centres are now considered a function of both the inputs and the measured outputs. The Orthogonal Least Squares (OLS) algorithm is one example of supervised selection of the Generalised RBF network’s centres. It is of course also possible to keep the spreads fixed and perform a stepwise or simultaneous optimisation of the centres and synaptic weights. We shall consider the simultaneous optimisation of centres, spreads and weights in section 4.5.

4.4.3 Results for an illustrative example

The next few subsections present the results obtained for an illustrative example using the various simple heuristics and a priori procedures available. These provide the comparison basis for the more sophisticated learning strategies presented in section 4.5. The example used throughout this chapter is the same as that included in chapter 3 and involves the reconstruction of the underlying surface from a set of 100 noisy exemplars shown in Figure 4.2.

Figure 4.2 – True surface and 100 noisy training exemplars.

158
4.4.3.1- Pre-specified Selection of Centres

Given the spreads $\sigma_j$'s, the centres may be chosen to be uniformly (evenly) distributed throughout the whole input space. The performance of this simple heuristic for $M=9$ and $M=49$ evenly spaced centres and different choices of spreads is shown in Figures 4.3 and 4.4. The spreads were calculated for isotropic spreads by equation (15), (16) or for diagonal spreads using equation (18).

Figure 4.3 shows the generalisation performance in the absence of regularisation and Figure 4.4 illustrates the performance obtained at the optimum level of regularisation. In each case, the optimal value of the regularisation parameter was computed by minimising the cross validation (CV) criterion using the generalised singular value decomposition (GSVD) technique. Figure 4.3 clearly shows the over-smoothing effect caused by decreasing the number of centres from $M=49$ to $M=9$. In the absence of regularisation neither of the isotropic heuristics (15) or (16) is capable of capturing the essential features of the underlying surface, see Figure 4.3. In contrast, heuristic (18) for diagonal spreads with $M=49$ recognises the general features of the underlying surface but is too oscillatory without regularisation. Figure 4.4 shows that regularisation has little effect for a small number of centres $M=9$. For a large number of centres, $M=49$, and an optimum level of regularisation all three heuristics recognise the general features of the underlying surface, compare Figures 4.2 & 4.4, with heuristic (18) performing marginally better.
Figure 4.3 – Performance of Generalised RBF networks with Equi-spaced centres.
(No Regularisation)
Figure 4.4 – Performance of Generalised RBF networks with Equi-spaced centres.
(with optimum level of regularisation)
Figures 4.5 and 4.6 present the results obtained for $M = 10$ and $M = 50$ centres selected at random from the 100 training exemplars. The spreads were again computed using heuristics (15), (16) and (18) as indicated on the diagrams. The optimal level of regularisation was obtained by minimising the cross validation criterion. Figure 4.5 shows that the random selection of the Generalised RBF centres performs better than equispaced centres when the number centres is relatively small ($M = 10$). This is in keeping with Haykin’s comment [Haykin, 1999] that random selection of the Generalised RBF centres from the training exemplars acts as some sort of regularisation. For this example, the isotropic spreads computed via equation (15) provide a better performance for a limited number of centres, $M = 10$, while the heuristic (18) for the calculation of the diagonal spreads is superior for a large number of centres, $M = 50$. 
Figure 4.5 - performances of Generalised RBF networks with random centres.
(Without regularisation).
Figure 4.6 – performances of the Generalised RBF networks with random centres

(With optimum level of regularisation)
4.4.3.2 – Self Organised Selection of Centres

For a given set of exemplars, the centres may be chosen in a self-organised way which recognises any data clusters in the input space and attempts to place the centres within these clusters. Moody and Darken [1989] have presented a simple technique which can be applied either adaptively or in a batch mode. In the adaptive approach, \( M \) initial centres are selected at random from the training exemplars. At each step of the adaptation an exemplar (\( x_i \)) is selected at random and the nearest centre to it \( t_j \) is moved by an amount,

\[
\Delta t_j = \eta (x_i - t_j)
\]

where \( \eta \) is a small positive constant. This simple procedure is then repeated until the centres cease to move. To illustrate various aspects of the self-organised algorithm, one hundred input exemplars were randomly generated within five distinct clusters \{ \( N_1 = 5, N_2 = 10, N_3 = 20, N_4 = 30, N_5 = 35 \) \} with equal radii and centres at \( C_1 = (0.1,0.1), C_2 = (0.1,0.9), C_3 = (0.9,0.1), C_4 = (0.9,0.9) \) and \( C_5 = (0.5,0.5) \).

Figure 4.7 gives two examples of the trajectories showing the movement of the RBF network centres (\( M=5 \)) from initial positions selected at random to their final destinations. This simple algorithm recognises well the clusters in the input space and positions the network centres approximately in the middle of the clusters. Our simulations show that the initial position of the centres is not practically important for large clusters but small clusters (with less than about 5 percent of the total exemplars) are sensitive to the initial position of the centres. If initially a centre is not selected within these small clusters there is a chance that larger clusters dominate the smaller ones and attract all the centres. This is demonstrated by the trajectories shown in Figure 4.8 with the smallest cluster at the bottom left not attracting a centre. Our
observations also show that the adaptive clustering algorithm rarely positions the centres between (or outside) the input clusters.

**Figure 4.7** – Selecting the RBF centres via the Adaptive clustering algorithm \( (M = 5) \).

**Figure 4.8** – Larger clusters dominating the smaller one in the Adaptive clustering algorithm \( (M = 5) \).
A major drawback of the adaptive technique is that the number of trials (random selections) for convergence can not be established \textit{a priori}. An alternative batch procedure was also suggested by Moody and Darken [1989] to overcome this drawback. The so-called \(k\)-means clustering algorithm computes a set of \(k\) centres which minimise the following merit function,

\[ K = \sum_{j=1}^{k} \sum_{i=1}^{N} M_{ji} \left\| t_j - x_i \right\|^2 \]  

where, \(K\) represents the total squared distance in the input space between the \(N\) training exemplars \((x_i; \ i=1,2,...,N)\) and the nearest of the \(k\) centres to each exemplar. \(M_{ji}\) is defined as a cluster membership function (with 0 and 1 entries) and identifies the centre to which a given exemplar belongs. The minimisation of (20) can be performed by any multidimensional optimisation method. We have used a variety of optimisation techniques including the first order gradient descent method, the Powell method, the Conjugate Gradient technique and the Variable Metric method with good results. The merit function (20) is convex and hence unimodel and most optimisation techniques perform well on finding that single minimum. Figure 4.9 compares the final position of the five RBF network centres obtained by the batch and adaptive methods using various optimisation techniques. Initial centres were selected at random from the training exemplars and the adaptive and the batch methods converged to the same final positions.
Figure 4.9 - Comparison of adaptive and batch $k$ means clustering algorithms using different optimisation techniques.

Figure 4.10 illustrates the performance of two Generalised RBF networks with $M = 10$ and $M = 50$ centres selected by the adaptive clustering algorithms using various choices of spreads in the absence of regularisation. Figure 4.11 shows the results obtained at the optimum level of regularisation. Initial centres were selected at random from the training exemplars and the number of iterations for the adaptive method was sufficiently large to ensure convergence. The results obtained indicate that for our illustrative example there is no specific improvement in the generalisation performance of the RBF networks compared to the much simpler random selection of the centres.
Figure 4.10—Performance of Generalised RBF networks with centres selected via the adaptive clustering method without regularisation ($\eta = 0.05$).
Figure 4.10 – Performance of Generalised RBF networks with centres selected via the adaptive method at optimum regularisation level ($\eta = 0.05$).
4.4.4 – Supervised Selection of Centres

All the procedures examined so far were for an RBF network of a known size. The optimum number of centres must then be computed via other methods such as cross validation or pruning algorithms. The Orthogonal Least Square (OLS) algorithm first applied by Chen et al [1991] chooses the RBF centres as a subset of the training exemplars one by one until an adequate network has been constructed. The form of the radial basis functions and their widths are known and kept fixed during the learning process. Chen et al [1991] claim that the OLS method although computationally more demanding provide a network with minimal number of neurones. Sherstinsky and Picard [1996] argued against the efficiency of the OLS algorithm and provided some examples where it does not produce the smallest possible network for a given approximation error. Nonetheless, the OLS algorithm represents a first attempt at combining model selection and learning and tuning procedures and merits a closer look.

Consider the solution of the regularisation network,

\[ F(x) = \sum_{j=1}^{N} w_j G(x; x_j) \]  \hspace{1cm} (21)

Applying this equation for each exemplar \( \bar{x}_i, i = 1,2,\ldots,N \) yields a system,

\[
\begin{bmatrix}
    y_1 \\
    y_2 \\
    \vdots \\
    y_N
\end{bmatrix} =
\begin{bmatrix}
    G(x_1; x_1) & G(x_1; x_2) & \cdots & G(x_1; x_N) \\
    G(x_2; x_1) & G(x_2; x_2) & \cdots & G(x_2; x_N) \\
    \vdots & \vdots & \ddots & \vdots \\
    G(x_N; x_1) & G(x_N; x_2) & \cdots & G(x_N; x_N)
\end{bmatrix}
\begin{bmatrix}
    w_1 \\
    w_2 \\
    \vdots \\
    w_N
\end{bmatrix} +
\begin{bmatrix}
    e_1 \\
    e_2 \\
    \vdots \\
    e_N
\end{bmatrix}
\]  \hspace{1cm} (22)

where \( e_i \) is the mismatch between the network prediction and the measured value for the \( i^{th} \) exemplar. Each column of the matrix \( G \) is referred to as a regressor and the system (22) can be expressed as,
\[
y = Gw + e
\]  
(23)

where \( y = [y_1, y_2, \ldots, y_N]^T \), \( w = [w_1, w_2, \ldots, w_N]^T \), \( e = [e_1, e_2, \ldots, e_N]^T \) and \( G \) is a matrix with the column partitioning,

\[
G = [g_1, g_2, \ldots, g_N]
\]  
(24)

with the \( j^{th} \) column (regressor) given by,

\[
g_j = [G(x_1, x_j), G(x_2, x_j), \ldots, G(x_N, x_j)]^T
\]  
(25)

All the regressors can be evaluated and stored up front and form a candidate set of regressors. Selecting the appropriate set of RBF centres via the OLS algorithm is equivalent to the selection of a subset of significant regressors from this candidate set. In the OLS method an attempt is made to build an orthogonal decomposition of the matrix \( G \),

\[
G = QR
\]  
(26)

where \( R \) is an \( N \times N \) upper triangular matrix with diagonal elements of unity and \( Q \) is an \( N \times N \) matrix with orthogonal columns. Chen et al [1991] used the classical Gram-Schmidt procedure to compute one column of \( R \) at a time and find the corresponding column of the matrix \( Q \) afterwards. At the first step, the regressor which has the maximum error reduction ratio, which is defined as,

\[
[err]_i = \frac{y^T g_i g_i^T y}{g_i^T y_y^T y} , \quad i = 1, 2, \ldots, N
\]  
(27)

is selected as the first column of \( Q \). On the \( k^{th} \) step \( (k > 1) \), the \( k^{th} \) column of matrix \( Q \) is constructed to be orthogonal to the previously selected columns with maximum error reduction ratio. The procedure is terminated at the \( M_s^{th} \) step when,
with $0 < \rho < 1$ a user specified tolerance. Chen et al [1991] recommended that the
tolerance $\rho$ is selected as the ratio $\sigma_e^2 / \sigma_y^2$, where $\sigma_e^2$ is the variance of the residuals
and $\sigma_y^2$ is the variance of the measured outputs. The latter variance is known for a
given training set but the variance of the residuals is unknown. However, an estimate
of $\sigma_y^2$ can be computed during the learning process. Figure 4.12 shows the progress
of the OLS technique for the illustrative example of Figure 4.2 for different types of
basis functions.

![Figure 4.12 - Effect of shape of the non-linearity on the tolerance (\(\rho\)).](image)

Figure 4.13 illustrates the generalisation performance of the Generalised RBF
networks with the centres selected via the OLS algorithm for different values of
spreads using isotropic Gaussian basis functions with and without regularisation. For comparison purposes, a tolerances of $\rho = 0.05$ was used for all cases considered. The results in Figure 4.13 indicate that the OLS algorithm requires relatively large number of centres to produce a reliable performance with an appropriate choice of spreads and an optimal level of regularisation.

![Graphs showing performance of Generalised RBF networks with centres selected via the OLS algorithm and isotropic spreads ($\rho = 0.05$)].

**Figure 4.13** – Performance of Generalised RBF networks with centres selected via the OLS algorithm and isotropic spreads ($\rho = 0.05$).

(a): no regularisation
(b): optimal level of regularisation
4.5 – Training of Generalised RBF networks viewed as an optimisation problem

The training of a Generalised RBF network can be viewed as the minimisation of the following merit function,

$$\min_{w, T, \Sigma} \mathcal{J}(\hat{w}, T, \Sigma) = \frac{1}{2} (y - Gw)^T (y - Gw) + \frac{1}{2} \lambda w^T G_0 w$$  \hspace{1cm} (29)$$

where \( w = [w_1, w_2, \ldots, w_M]^T \) is the linear weights vector, \( T \) is the \( M \times p \) matrix of the RBF centres, \( \Sigma \) is the \( M \times p \times p \) tensor of the Generalised RBF variance-covariance matrices and \( \lambda \) establishes the level of regularisation. The direct minimisation of (29) with respect to \( w, T, \Sigma \) and the selection of the appropriate regularisation level \( \lambda \) constitutes a highly non-linear and difficult problem and is not a practical proposition.

There are two basic procedures for tackling this difficult problem which are considered below. The first approach employs a hierarchical methodology for the simultaneous optimisation of the linear and non-linear parameters. The second approach handles the optimisation problem sequentially by optimising the neurones one at a time. In both procedures the appropriate level of regularisation is determined by minimising the leave one out cross validation (CV) criterion as described in Chapter 2.

4.5.1 – Hierarchical simultaneous optimisation of Generalised RBF networks

Instead of direct minimisation of (29) with respect to \( w, T, \Sigma \), which constitutes a highly non-linear and difficult problem, we may take advantage of the fact that the synaptic weights \( w \) appear linearly in (29) and adopt the following hierarchical approach:
S1) For fixed $T$ and $\Sigma$ (i.e. $G$ and $G_0$) find the optimal $\lambda^*$ and $w_{\lambda^*}$ from,

$$\min_w \mathcal{S}(\bar{F}(w)) = \frac{1}{2} (y - Gw)^T (y - Gw) + \frac{1}{2} \lambda w^T G_0 w$$  \hspace{1cm} (30)$$

using the leave one out cross validation criterion.

S2) With $w = w_{\lambda^*}$, find improved $T$ and $\Sigma$ from,

$$\min_{T, \Sigma} \mathcal{S}(\bar{F}(T, \Sigma)) = \frac{1}{2} (y - Gw_{\lambda^*})^T (y - Gw_{\lambda^*})$$  \hspace{1cm} (31)$$

S3) Repeat steps S1 and S2 until the non-linear parameters $T$ and $\Sigma$ cease to change.

The minimisation in step S1 presents no difficulty. For given non-linear parameters, the optimal level of regularisation $\lambda^*$ and the corresponding synaptic weight $w_{\lambda^*}$ is obtained by minimising the leave one out cross validation (CV) criterion. Efficient and stable methods employing the generalised singular value decomposition (GSVD) were developed in this study and can be used to good effect.

The major difficulty is in the computation of the optimal non-linear parameters $T$ and $\Sigma$ in step S2. The non-linear objective function (31) can be re-written as,

$$\mathcal{S}(\bar{F}(T, \Sigma)) = \frac{1}{2} (y - Gw_{\lambda^*})^T (y - Gw_{\lambda^*})$$

$$\min_{T, \Sigma} = \sum_{i=1}^N (y_i - \sum_{j=1}^M w_{j^*}^i G(x_i; t_j, \Sigma_j^{-1}))^2$$  \hspace{1cm} (32)$$

Minimisation of (32) can be performed using various optimisation techniques ranging from the simple first order gradient descent method to the more advanced higher order
techniques such as modified Newton, Gauss Newton, Modified Gauss Newton or Levenberg – Marquardt methods. All of these optimisation techniques are, however, only capable of finding a local minimum of the objective function (32). For illustration purposes we shall use the gradient descent method with a controlled step-size to optimise a Generalised RBF network with diagonal spreads.

The Gaussian basis function with a diagonal norm-weighting matrix is given by,

$$G(\| \mathbf{x}_i - t_k \|_{C_k}) = \prod_{l=1}^{p} \exp[- \frac{(x_{i,l} - t_{k,l})^2}{\sigma_{k,l}^2}]$$

and the gradients of the merit function (32) with respect to the non-linear parameters $t_{k,l}$ and $\sigma_{k,l}$, can be expressed as:

\[\begin{align*}
\text{a) Gradients with respect to centre } t_{k,l} \\
\frac{\partial \mathcal{J}}{\partial t_{k,l}} &= - w_k \sum_{i=1}^{N} e_i \left[ \frac{\partial}{\partial t_{k,l}} G(\| \mathbf{x}_i - t_k \|_{C_k}) \right]; \quad k=1,2,\ldots,M; \quad l=1,2,\ldots,p \\
\text{with:} \\
\frac{\partial}{\partial t_{k,l}} G(\| \mathbf{x}_i - t_k \|_{C_k}) &= 2 \cdot \frac{(x_{i,l} - t_{k,l})}{\sigma_{k,l}^2} \prod_{m=1}^{p} \exp[- \frac{(x_{i,m} - t_{k,m})^2}{\sigma_{k,m}^2}]
\end{align*}\]

\[\begin{align*}
\text{b) Gradients with respect to spread } \sigma_{k,l} \\
\frac{\partial \mathcal{J}}{\partial \sigma_{k,l}} &= - w_k \sum_{i=1}^{N} e_i \left[ \frac{\partial}{\partial \sigma_{k,l}} G(\| \mathbf{x}_i - t_k \|_{C_k}) \right]; \quad k=1,2,\ldots,M; \quad l=1,2,\ldots,p \\
\text{with:}
\end{align*}\]
Improved values of $t_{k,l}$ and $\sigma_{k,l}$ are obtained using the following first order update formulas:

$$t_{k,l}(n+1) = t_{k,l}(n) - \eta_1 \frac{\partial \Omega(n)}{\partial t_{k,l}(n)} ; \quad k = 1, 2, ..., M; \quad l = 1, 2, ..., p \quad (38)$$

$$\sigma_{k,l}(n+1) = \sigma_{k,l}(n) - \eta_2 \frac{\partial \Omega(n)}{\partial \sigma_{k,l}(n)} ; \quad k = 1, 2, ..., M; \quad l = 1, 2, ..., p \quad (39)$$

where, $\eta_1$ and $\eta_2$ are learning rate parameters. The learning rate parameters should be small to ensure the stability of the iterative procedure. On the other hand, small learning rate parameters lead to tedious and slow learning. A step-length control mechanism is therefore essential to ensure both stable and quick training. A simple step-length control mechanism was used to satisfy these contradictory requirements. The hierarchical optimisation procedure can be arranged in alternative ways. For example, we may iterate step S2 to convergence before re-applying step S1. Our results indicate that such a procedure freezes the non-linear parameters and affects the overall convergence adversely. The best results were obtained by using a single update in step S2 and reusing the updated value in step S1 immediately. This ensures that the synaptic weights $w_{ij}$ do not change appreciably between iterations and yields a faster and more stable algorithm.

The performance of the above procedure for the illustrative example of Figure 4.2 was considered for two networks with $M = 5$ and $M = 10$ centres and diagonal spreads. Initial centres were selected via the adaptive k-means clustering algorithm [Moody and
Darken, 1989] and the initial spreads were computed via heuristic (18). For comparison purposes the optimisation loop was interrupted after 20000 iterations for all cases.

Figures 4.14 and 4.15 show: (a) the true surface, (b) the initial predictions, (c) the final prediction without regularisation and (d) the final prediction with optimal regularisation for $M = 5$ and $M = 10$ respectively.

Figure 4.14 – Generalisation performance of Generalised RBF networks using simultaneous optimisation of the non-linear parameters via the gradient descent method. ($M = 5$ neurones, diagonal spreads)
Figure 4.15 – Generalisation performance of Generalised RBF networks using simultaneous optimisation of the non-linear parameters via the gradient descent method. ($M = 10$ neurones, diagonal spreads)
Figures 4.14 and 4.15 clearly illustrate that optimising the non-linear parameters provides substantial improvement in the generalisation performance of a Generalised RBF network. A major draw back to the gradient descent method is the slow speed of training. Executing 20000 iterations for a regularised network with $M = 10$ neurones and diagonal spreads took 4.3 hours on a Pentium II (300 MHz) machine. We may resort to higher order non-linear optimisation methods to speed up the training but such methods are, in general, less stable than the gradient descent technique. Improving the speed of training, particularly for large networks, demands a different approach which is considered next.

4.5.2 - Sequential optimisation of Generalised RBF networks

The hierarchical simultaneous optimisation of the RBF network parameters leads to a substantially better performance with a small number of neurones. However, the number of parameters increases rapidly with the number of neurones. For the general case of a network with full norm weighting matrices, the number of parameters to be optimised is $M(p^2 + p + 1)$, where $M$ is the number of centres and $p$ is the dimension of the input space. For modest values of $M=20$ and $p=5$, the number of parameters to be optimised is more than six hundred. The gradient descent technique is not appropriate for the simultaneous optimisation of such a large number of parameters and proves too time consuming. In an alternative approach, the optimisation of the non-linear parameters may be handled sequentially by considering one neurone at a time. At each stage, the optimisation is then reduced to the determination of $(p^2 + p + 1)$ parameters which is more manageable.
Zhang and Morris [1998] proposed a general sequential orthogonal approach for building up and training a single hidden layer neural network. The procedure starts with a single hidden neurone and sequentially increases the number of hidden neurones until the model error is sufficiently small. For a given set of \( y = [y_1, y_2, \ldots, y_N]^T \) and a network with \( M \) neurones, the network residuals \( e = [e_1, e_2, \ldots, e_N]^T \) are given by,

\[
\begin{align*}
    y &= \hat{y} + e_M \\
    &= Gw + e_M \\
    &= \sum_{j=1}^{M} w_j g_j + e_M
\end{align*}
\]

where we have partitioned the Green’s matrix as \( G = [g_1, g_2, \ldots, g_M] \). Equation (40) can be considered in a sequential way as,

\[
\begin{align*}
    y &= w_1 g_1 + e_1 \\
    e_1 &= w_2 g_2 + e_2 \\
    \ldots &= \ldots \\
    e_{n-1} &= w_n g_n + e_n \\
    \ldots &= \ldots \\
    e_{M-1} &= w_M g_M + e_M
\end{align*}
\]

which enables a sequential approach to the optimisation of the network. The first neurone is used to model the relationship between input and output data. Additional neurones are introduced to model the relationship between the input data and the model residuals. In Zhang and Morris’s approach [1998] the performance of the network is assessed against a set of test data after the addition of each new neurone and the procedure terminates when no significant improvement can be obtained.
The contribution of the $n^{th}$ neurone to the overall response, $w_n g_n$, may be decomposed into two parts. This is achieved by splitting the vector $g_n$ into two components: one which is in the space spanned by all the neurones added so far, i.e. $g_1, g_2, \ldots, g_{n-1}$, and another which is orthogonal to this space which will be denoted by $R_n$. This allows us to express the relation between $e_n$ and $e_{n-1}$ as,

$$e_{n-1} = w_n R_n + w_n (g_n - R_n) + e_n$$

(44)

The new information introduced by the $n^{th}$ neurone is contained in $w_n R_n$ and the objective is to maximise this information by adjusting the parameters of the $n^{th}$ neurone. The contribution $w_n (g_n - R_n)$ lies in the space spanned by $g_1, g_2, \ldots, g_{n-1}$ and can be accommodated by adjusting the weights $w_1, w_2, \ldots, w_{n-1}$, after the optimisation step. For the purpose of optimisation, therefore, we need only consider,

$$e_{n-1} = w_n R_n + e_n$$

(45)

It is clear from (45) that:

$$\| e_n \|^2 = (e_{n-1} - w_n R_n)^T (e_{n-1} - w_n R_n)$$

$$= e_{n-1}^T e_{n-1} - 2 w_n e_{n-1}^T R_n + w_n^2 R_n^T R_n$$

$$= \| e_{n-1} \|^2 - 2 w_n e_{n-1}^T R_n + w_n^2 \| R_n \|^2$$

(46)

Noting that the first term is fixed and the last term is positive, the task is reduced to finding a neurone such that its corresponding $R_n$ is most aligned with $e_{n-1}$. This is achieved by adjusting $w_n$, $\ell_n$ and $\Sigma_n$ such that,
The efficient implementation of the above sequential training requires the efficient calculation of a set of orthogonal bases $R_1, R_2, \ldots, R_n$, for the space spanned by $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n$. A number of techniques are available for the stepwise construction of orthogonal bases for a matrix, notably the classical and modified Gram-Schmidt orthogonalisation methods [Press et al, 1992; Golub and Van Loan, 1996]. Zhang and Morris [1998] employed the classical Gram-Schmidt orthogonalisation method for the ease in calculating the network error gradients.

Minimisation of (47) with respect to $w_n$ is simply carried out by applying the optimality condition as,

$$
\frac{\partial \| e_n \|^2}{\partial w_n} = -2e_{n-1}^T R_n + 2w_n R_n^T R_n = 0
$$

The optimal values of the non-linear parameters $t_n$ and $\Sigma_n$ can be computed using a variety of non-linear optimisation methods. Zhang and Morris [1998] employed the Levenberg-Marquadt optimisation technique for this purpose. Once the training procedure is terminated, the output layer weights of the neurones must be readjusted in order to accommodate the effects of the $w_n(\varepsilon_n - R_n)$ terms. Zhang and Morris [1998] used the OLS method to readjust the weights at termination.
The crucial importance of regularisation for improving the generalisation performance of Generalised RBF networks with noisy data was also recognised by Zhang and Morris [1998]. To accommodate regularisation, they suggested the addition of a term to the objective function (47),

$$
\min_{w_n, t_n, \Sigma_n} J_n(w_n, t_n, \Sigma_n) = \| e_n \|^2 + \frac{\lambda}{R_n^TR_n}
$$

(50)

to penalise very small values of $\| R_n \|$. They recommended that the entire procedure is executed for several different values of the regularisation parameter $\lambda$ and the appropriate level of regularisation is chosen a posteriori. This can prove to be too time consuming and is perhaps the weakest link in the otherwise excellent algorithm of Zhang and Morris [1998]. The orthogonal sequential algorithm summarised below draws heavily on the procedure of Zhang and Morris [1998] but enables the automatic selection of the regularisation level $\lambda$ using the leave one out cross validation technique developed in this study. Its main difference with the procedure of Zhang and Morris [1998] is that the number of neurones $M$ is fixed before hand. The entire procedure is run for several different values of $M$ making full use of the powerful leave one out cross validation technique to select the appropriate level of regularisation automatically. The final model is chosen as the optimally regularised models providing the best compromise between model complexity and model accuracy.

The major steps of the orthogonal sequential algorithm developed in this study are summarised below:
S0) Select the number of neurones $M$ and set $e_0 = \gamma$

S1) Initialise the first neurone: $t_1^0$ and $\Sigma_1^0$

$k=0$
Repeat

Calculate $g_1^k$
Set $R_1^k = g_1^k$

$w_1^k = \frac{(e_0^k R_1^k)}{(R_1^k T R_1^k)}$

$t_1^{k+1} = t_1^k - \eta \frac{\partial J_1^k}{\partial t_1^k}$

$\Sigma_1^{k+1} = \Sigma_1^k - \eta \frac{\partial J_1^k}{\partial \Sigma_1^k}$

$k=k+1$
Until convergence

$e_1 = e_0 - w_1 R_1$

S2) For $n=2,...,M$

Initialise the $n^{th}$ neurone: $t_n^0$ and $\Sigma_n^0$

$k=0$
Repeat

Calculate $g_n^k$

Use Gram-Schmidt orthogonalisation procedure to find $R_n^k$

$w_n^k = \frac{(e_{n-1}^k R_n^k)}{(R_n^k T R_n^k)}$

$t_n^{k+1} = t_n^k - \eta \frac{\partial J_n^k}{\partial t_n^k}$

$\Sigma_n^{k+1} = \Sigma_n^k - \eta \frac{\partial J_n^k}{\partial \Sigma_n}$

$k=k+1$
Until convergence

$e_n = e_{n-1} - w_n R_n$

find the optimal regularisation level $\lambda_n^*$ and adjust the synaptic weights $w_1^*,...,w_n^*$ by minimising the leave one out cross validation criterion.

* recalculate $e_1^*,...,e_n^*$

End for
The steps indicated by ‘Φ’ are new to the procedure of Zhang and Morris [1998] and deserve further comment. The regularisation procedure suggested by Zhang and Morris [1998] is in effect similar to a type of ridge regression which attempts to penalise large synaptic weights. We have already demonstrated in Chapters 2 and 3 that a wide range of more efficient regularisation procedures are available. The sequential orthogonal algorithm developed here make use of the regularisation procedure outlined in section 4.2 which is a natural and efficient choice for Generalised RBF networks with known non-linear parameters.

Figure 4.16 shows the generalisation performance of the above sequential algorithm for Generalised RBF networks with an isotropic spread applied to the illustrative example of Figure 4.2. The initial centres were selected via the adaptive k-means clustering algorithm and heuristic (15) was used to initialise the isotropic spreads. The modified Newton optimisation program [Gill and Murray, 1997] was used in place of the gradient descent method to obtain the optimal non-linear parameters at each sequential stage. The optimum level of regularisation was computed by minimising the cross validation criterion. The sequential orthogonal algorithm is much faster than the hierarchical simultaneous optimisation of the non-linear parameters. In the case of a Generalised RBF network with \( M = 10 \) neurones and isotropic spreads the sequential orthogonal algorithm required 54 seconds on a Pentium II 300 MHz compared to 4.3 hours required by the simultaneous optimisation method of section 4.5.1.
Figure 4.16 – Generalisation performances of the sequential orthogonal algorithm for training the Generalised RBF networks with isotropic spreads.
4.5.2.1 - Sequential back-fit algorithm for constructing the entire RBF network

A major difficulty with the orthogonalisation procedure is that the $n^{th}$ neurone is optimised with the non-linear parameters of neurones 1 to $n-1$ (i.e. the basis vectors $g_1, g_2, \ldots, g_{n-1}$) considered fixed. This enables us to use the highly efficient Gram-Schmidt orthogonalisation procedure [Press et al, 1992; Golub and Van Loan, 1996] to determine $g_n$. It must be recognised however that by freezing the non-linear parameters of neurones 1 to $n-1$, we preclude the possibility of a better network with $n$ neurones but free non-linear parameters. This situation can be avoided, at the expense of increased computational cost, by resorting to a sequential back-fit algorithm which avoids the use of orthogonalisation. In this approach the neurones are optimised one at a time but an opportunity is made to readjust the parameters of previously optimised neurones. The optimal level of regularisation and the linear weights of the entire Generalised RBF network are again calculated via minimisation of the leave one out cross validation criterion.

At each inner step of the back-fit procedure, the non-linear parameters of the $j^{th}$ neurone are adjusted to describe the residual error of all other neurones as closely as possible. That is we consider the minimisation of the objective,

$$
\min_{\{\epsilon_j, \Sigma_j\}} || \epsilon_j ||^2 = \frac{1}{2} (r_j - w_j g_j)^T (r_j - w_j g_j)
$$

(51)

where the $r_j$ represents the model residuals excluding the $j^{th}$ neurone and is given by,
The entire back-fit procedure can be summarised as follows,

S1) Select the number of neurones \( M \) and initialise the centres and the spreads of the entire Generalised RBF network.

S2) Do (back-fit loop)

Check termination criteria, exit back-fit loop if termination criteria are satisfied

Do \( j = 1, \ldots, M \)

find the optimal regularisation level \( \lambda^*_n \) and adjust the synaptic weights

\[
\begin{align*}
& w_{1, \lambda^*, \ldots, w_{M, \lambda^*y}} \\
& b
\end{align*}
\]

minimising the leave one out cross validation criterion.

calculate \( r_j \) as given by (52)

\( k = 0 \)

Repeat

\[
\begin{align*}
& \xi_{1}^{k+1} = \xi_{1}^{k} - \eta \frac{\partial J_{1}^{k}}{\partial \xi_{1}^{k}} \\
& \Sigma_{1}^{k+1} = \Sigma_{1}^{k} - \eta \frac{\partial J_{1}^{k}}{\partial \Sigma_{1}^{k}} \\
& k = k + 1
\end{align*}
\]

Until convergence

Continue

Continue
The optimal network structure is selected \textit{a posteriori} by repeating the above sequential back-fit procedure for different values of $M$. The most suitable network structure is chosen as that value of $M$ providing the best compromise between model complexity and model accuracy. In our applications we used the Akaike's information criterion (AIC) which presents a trade off between model accuracy and model complexity and may be expressed as,

$$AIC(\chi) = N \log(\nu) + M \chi$$ \hspace{1cm} (53)

Here, $N$ is the number of training exemplars, $\nu$ is the variance of the model residuals, $M$ is the number of model parameters which is taken as the number of neurones and $\chi$ is the critical value of the chi-square distribution with one degrees of freedom. Leontarities and Billings [1987] suggest $\chi = 4$ as a convenient choice which corresponds to a significance level of 0.0456.

The performance of the sequential back-fit algorithm for training Generalised RBF networks with 5 and 10 neurones and diagonal spreads is shown in Figure 4.17. Initial centres were selected via the \textit{adaptive k-means clustering} algorithm and heuristic (18) was used to initialise the spreads. The modified Newton optimisation program [Gill and Murray, 1997] was used in place of the gradient descent method in the inner loop. The optimum level of regularisation at each stage was computed by minimising the leave one out cross validation criterion using the generalised singular value decomposition (GSVD) technique. The promising generalisation performance of the sequential back-fit algorithm is shown clearly in Figure 4.17. The algorithm is relatively fast and for a fully regularised
network with $M = 10$ neurones requires about 23 minutes on a Pentium II 300 MHz compared to 4.3 hours required by the simultaneous optimisation method of section 4.5.1.

**Figure 4.17** – Generalisation performances of the sequential back-fit algorithm for 5 and 10 neurones with different levels of regularisation.
Figure 4.18 shows the variation of the Akaike's information criterion (53) for $\chi = 4$ with the number of neurones. Figure 4.19 compares the generalisation performances of the Generalised RBF networks corresponding to minima of the AIC curve with and without regularisation. The excellent performance of the sequential back-fit algorithm is highlighted by the fact that a faithful reconstruction of the true surface is obtained with only 2 neurones using 100 noisy data points.

**Figure 4.18** – Variation of the Akaike's information criterion (AIC) with the number of hidden neurones of the Generalised RBF network.
Figure 4.19 – Generalisation performances of the sequential back-fit algorithm for the networks corresponding to the minima of the AIC curve with and without regularisation.
4.6 - A Steady State Isothermal Continuous Stirred Tank Reactor

We close this chapter with the application of the sequential back-fit algorithm to a simple chemical engineering process. The process is taken as a steady state, isothermal, constant volume ideal continuous stirred tank reactor in which a reactant A is converted to a product R.

\[ Q_i, C_{A_i}, T_i \]

\[ C_{R_i} = 0 \]

\[ Q_o, C_{A_o}, T_o \]

\[ C_{R_o} \]

\[ A \leftrightarrow R \]

Figure 4.20 – Schematic representation of a CSTR for the reaction \( A \leftrightarrow R \)

The concentration of the product \( C_{R_o} \) leaving this reactor is a function of the inlet concentrations \( C_{A_i}, C_{R_i}, T_o \) and the reactor residence time \( \tau \):

\[ C_{R_o} = f(C_{A_i}, C_{R_i}, T_o, \tau) \] (54)

We assume that the kinetics of the reaction are unknown and the only information available is several direct measurements of the input and output variables. The objective is to train a neural network to predict the functional relationship in (54) based on this limited and noisy data set. For the purpose of illustration, we shall assume that the data is
taken at a fixed and known residence time $\tau$, and the reactor feed contains no product, $C_{R_i} = 0$. This gives a 3D problem which helps in visualising the results.

Synthetic experimental data was simulated for a fixed residence time of $\tau = 60 s$ and a feed of pure $A$ for the following first order reversible reaction:

$$
A \rightleftharpoons R
$$

(55)

with

$$
k_f = 5 \times 10^3 \exp\left[-\frac{5033}{T}\right] \quad s^{-1}
$$

(56)

$$
k_r = 1 \times 10^6 \exp\left[-\frac{7549}{T}\right] \quad s^{-1}
$$

(57)

The steady state mass balance on this reactor reduces to,

$$
C_{R_o} = \frac{\tau k_f}{(1 + \tau (k_f + k_r))} C_{A_i}
$$

(58)

Figure 4.21 shows the surface of $C_{R_o}$ as a function of $C_{A_i}$ and $T_o$ in the range $0 \leq C_{A_i} \leq 1$ and $0.3 \leq T/1000 \leq 1$. The training data were obtained by adding random Gaussian deviates of spread 0.2 to the true surface at 100 randomly chosen points and are also shown on Figure 4.21.
The sequential back-fit training algorithm was employed to train Gaussian RBF networks with a different number of neurones. Initial centres were selected via the adaptive $k$-means clustering algorithm and the initial spreads were computed using the diagonal heuristic (18). Figure 4.22 illustrates the variations of the Akaike's information criterion (53) with the number of the neurones for the converged networks. Figure 4.23 compares the true underlying surface and the prediction of a Generalised RBF network with $M = 5$ corresponding to the minimum of the AIC. It is evident that the sequential back-fit algorithm produces a small network ($M = 5$) with reliable predictions over the entire input domain.
Figure 4.22 - Variations of the Akaike’s information criterion (AIC) with the number of neurones of the converged Generalised RBF networks.

Figure 4.23 – Comparison of the true response and the predictions of the optimal Generalised RBF network trained by the sequential back-fit algorithm ($M^* = 5$, fully regularised).
4.7- Conclusions

The one-to-one correspondence between the number of hidden neurones of the Regularisation network and the number of data points \( N \) can prove troublesome. The computational burden of a Regularisation network grows as \( N^3 \) and may prove prohibitive for large data sets. The so-called Generalised RBF network breaks this one-to-one correspondence and searches for a sub-optimal solution in a lower dimensional space with respect to the number of hidden layer neurones. This is achieved by using a smaller number of neurones \( M \ll N \) and distinguishing between the training exemplars and the hidden neurones of the RBF network. The reduction in the number of hidden neurones inevitably limits the approximation power of the network and the training of a Generalised RBF network reduces to a challenging large-scale optimisation problem which has received considerable attention over the past decade. This Chapter started with a thorough review of the literature to place this study in the context of the previous work.

The multivariate linear regularisation theory was extended to derive the working equations of a Generalised RBF network. For a given network size \( M \), the training of a Generalised RBF network requires the calculation of the synaptic weights, estimation of the centres and spreads of the radial basis activation functions and the estimation of the level of regularisation. Given the centres, spreads and the level of regularisation, the calculation of the linear synaptic weights reduces to the solution of an over-determined set of linear equations which can be handled by a variety of efficient and stable techniques. The combination of the leave one out cross validation criterion and the generalised singular value decomposition as described in Chapter 2 coupled with the use
of similarity transformation may be used to establish the optimum level of regularisation. The major difficulty is the optimisation of the centres and spreads which appear non-linearly. A number of heuristics have been proposed to circumvent this large-scale optimisation altogether. An illustrative example was used in this chapter to highlight the limited reliability of the heuristic approach and the need for a direct solution of the non-linear optimisation problem.

Several procedures have been developed to optimise the location of the centres of a Generalised RBF network with fixed spreads, notably the unsupervised $k$-means clustering algorithm [Moody and Darken, 1989] and the supervised Orthogonal Least Square method [Chen et al, 1991]. In contrast, the optimisation of the spreads which determine the width and the orientation of the receptive field of the neurones, has received little attention to date. This is largely because the majority of the reported applications have concerned problems such as speech and vision recognition, classification or time series prediction with very large and dense data sets. The networks developed in such cases usually contain many neurones and the details of the receptive fields are then less important so long as the entire input domain is adequately covered. In applications involving limited and sparse data sets, such as those frequently encountered in chemical engineering problems, the networks have a small number of neurones. The proper selection of the spreads is crucial to the performance of such small networks.

This chapter was focused on the calculation of the optimal non-linear parameters of a Generalised RBF network. In particular, a sequential approach was presented in which
the neurones were optimised one at a time. Two alternatives were considered, in the orthogonal sequential approach a neurone once optimised remained fixed. The performance of an orthogonal sequential algorithm for training the Generalised RBF networks was investigated for an illustrative example. Freezing the optimised neurones may lead to a sub-optimal network and a back-fit procedure was adopted to circumvent this problem. In particular, a novel sequential back-fit algorithm was developed which enabled the optimisation to proceed one neurone at a time without freezing the previously optimised neurones. The new algorithm was tested with very promising results and its application to a simple chemical engineering process was also considered.
Chapter 5

Separation of Local and Global Features: Adaptive Generalised RBF networks

5.1 - Introduction

In some applications the true surface underlying a noisy data set consists of distinctive local and global features. These dual features are interwoven together to form the overall behaviour of the underlying process. Existing multivariate regression techniques as well as conventional neural networks can only filter out the noise and are aimed at recovering the combination of the global and local features. Evidently, separation of these dual features can provide further insight about the governing mechanisms of the underlying process but few researchers have paid attention to this issue. In general, projection based neural networks or RBF networks with a global activation function are better suited for recovering gently varying (low frequency) global trends. In contrast, sharp (high frequency) local features are more readily captured with RBF networks employing localised activation functions. In this chapter, we develop an adaptive neural network aimed specifically at separating the local and global features embedded in the data set.
Powell [1987a&b] introduced the idea of coupling known order multivariate polynomials with traditional radial basis functions for the strict multivariable interpolation problem. The basic problem was considered as the reconstruction of a continuous hyper-surface $F(x) \in \mathbb{R}^1$ in a multidimensional space $x \in \mathbb{R}^p$ which satisfies the interpolation conditions $F(x_i) = y_i$, where $\{x_i, y_i; i = 1, 2, ..., N\}$ are a set of distinct predictors and the corresponding responses (outputs). Powell [1987a] used the following model to couple radial basis functions with known order polynomial bases,

$$F(x_i) = F_{local}(x_i) + F_{global}(x_i)$$

$$= \sum_{j=1}^{N} w_j \varphi_j \left( \| x_i - x_j \| \right) + \sum_{k=1}^{L} \tilde{w}_k \psi_k(x_i) \quad i = 1, 2, ..., N \quad (1)$$

Here, $F(x)$ is the overall model or estimate of the truth. The $\psi_k(x)$’s are the bases for the space of the algebraic polynomials of known degree at most $L$ which construct the global response $F_{global}(x)$. The $\varphi_j(x)$’s are a set of radial basis functions (such as Gaussians or Inverse Multiquadrics) building the local response $F_{local}(x)$. The constants $w_j$ and $\tilde{w}_k$ represent the linear interpolation coefficients and $\| . \|$ is intended as the Euclidean norm. The interpolation conditions $F(x_i) = y_i; i = 1, 2, ..., N$ provide $N$ constraints for the $N + L$ coefficients $w_1, ..., w_N, \tilde{w}_1, ..., \tilde{w}_L$. Powell [1987a] employed the additional constraints,

$$\sum_{i=1}^{N} w_i \psi_k(x_i) = 0, \quad k = 1, 2, ..., L \quad (2)$$

to provide sufficient constraints for the calculation of the $N + L$ linear coefficients.

Equations (1) and (2) can be combined into a system of linear equations,
\[ A\tilde{w} = \tilde{y} \]  

where \( \tilde{y} = [y_1, y_2, \ldots, y_N]^{T} \cdot \tilde{w} = [w_1, \ldots, w_N; \tilde{w}_1, \ldots, \tilde{w}_L]^{T} \) and the \((N + L) \times (N + L)\) matrix \( A \) is partitioned as,

\[
A = \begin{bmatrix}
\Phi & \cdots & \Psi \\
\vdots & \ddots & \vdots \\
\Psi^{T} & & 0
\end{bmatrix}
\]

with \( \Phi_{ij} = \phi_j(\|x_i - x_j\|) \); \( i = 1, 2, \ldots, N \) and \( j = 1, 2, \ldots, N \) (4)

and \( \Psi_{ik} = \psi_k(x_i) \); \( i = 1, 2, \ldots, N \) and \( k = 1, 2, \ldots, L \) (5)

Provided that the data points are distinct and the columns of the matrix \( \Psi \) are independent, it can be proven that the matrix \( A \) turns out to be non-singular for a number of radial basis functions [Micchelli, 1986; Powell, 1987a; Haykin 1999]. Equation (3) then yields a unique solution for the linear coefficients and the function (1) interpolates the data exactly.

The basic idea of Powell [1987a] may be modified and applied within the context of the multivariate regularisation theory. We recall that the solution of a Generalised RBF network is obtained by minimising the regularised functional,

\[
\min_{\tilde{F}_\lambda} \mathcal{J}(\tilde{F}_\lambda) = \frac{1}{2} \sum_{i=1}^{N} (y_i - \tilde{F}_\lambda(x_i))^2 + \frac{1}{2} \lambda \| D[\tilde{F}_\lambda(x)] \|^2
\]

(7)

where,

\[
\tilde{F}_\lambda(x) = \sum_{j=1}^{M} w_j G(x; t_j, \Sigma)
\]

(8)
and \( G(x; t_j, \Sigma) \) represents the Green's function corresponding to the operator \( D \). Now, the regularisation term \( \| D[\tilde{F}_\lambda(x)] \|^2 \) is blind to any function which lies in the null space of the operator \( D \). We may therefore add any number of such functions to \( \tilde{F}(x) \) without affecting the level of regularisation. With this in mind, we may seek to find an approximating function in the form,

\[
\tilde{F}_\lambda(x) = \sum_{j=1}^{M} w_j G(x; t_j, \Sigma) + \sum_{k=1}^{L} w_k \psi_k(x)
\]

(9)

with each \( \psi_k(x) \) lying in the null space of the operator \( D \):

\[
D[\psi_k(x)] = 0, \quad k = 1, \ldots, L
\]

(10)

This gives us the flexibility to model the local response \( F_{\text{local}}(x) \) by the first term in (9) and the global response \( F_{\text{global}}(x) \) with the second term in (9) consisting of null space terms. For example, we may choose the translationally and rotationally invariant operator given by equations 3.33 and 3.34 whose Green's function is the isotropic Gaussian

\[
G(x; t_j, \Sigma) = \exp\left[ -\frac{1}{2} (x - t_j)^T \Sigma^{-1} (x - t_j) \right]
\]

(11)

All multivariate polynomials of finite degree lie in the null space of such an operator and may be chosen to represent the global response. A schematic representation of a Generalised RBF network with additional null space terms is shown in Figure 5.1.
5.2 – Training of a Generalised RBF network with null space terms

We shall now consider the training of the network of Figure 5.1 which includes a number of null space terms explicitly. Let us assume for the moment that all the null space terms may be combined into a single function \( \psi(x) \). We shall relax this restriction subsequently without affecting the derivation. We therefore seek a solution of the form,

\[
\bar{F}_A(x) = \sum_{j=1}^{M} w_j G(x; \ell_j, \Sigma) + \psi(x)
\]  

\( (12) \)
which minimises the merit function (7) with $D[\psi(x)] = 0$. Substituting (12) in (7) yields,

$$\min_{\bar{F}_\lambda} \mathcal{S}(\bar{F}_\lambda) = \frac{1}{2} \sum_{i=1}^{N} \left( y_i - \psi(x_i) - \sum_{j=1}^{M} w_j G(x_i; t_j; \Sigma) \right)^2 + \frac{1}{2} \lambda \left\| D[\sum_{j=1}^{M} w_j G(x; t_j; \Sigma)] \right\|^2 \quad (13)$$

For a set of given non-linear parameters $t_j$'s and $\Sigma$, the first term in (13) can be expressed as,

$$A(w, \psi) = \frac{1}{2} [\psi - \psi - Gw]^T [\psi - \psi - Gw] \quad (14)$$

where $\psi = [y_1, y_2, ..., y_N]^T$, $\psi = [\psi(x_1), \psi(x_2), ..., \psi(x_N)]^T$ and $G$ is the Greens matrix with elements,

$$G_{i,j} = G(x_i; t_j, \Sigma) \quad (15)$$

The second (regularisation) term in (13) may be stated as,

$$B(w) = \frac{1}{2} \int_{\mathbb{R}^p} D[\bar{F}(x)] D[\bar{F}(x)] \, dx \quad (16)$$

Substituting for $\bar{F}_\lambda(x)$ from (12) and noting that $D[\psi(x)] = 0$, leads to,

$$B(w) = \frac{1}{2} \int_{\mathbb{R}^p} D[\sum_{j=1}^{M} w_j G(x; t_j; \Sigma)] D[\sum_{k=1}^{M} w_k G(x; t_k; \Sigma)] \, dx \quad (17)$$

Using the definition of the adjoint operator $D^*$ (see section 3.1.2) we may express (17) as,

$$B(w) = \frac{1}{2} \int_{\mathbb{R}^p} \left[ \sum_{j=1}^{M} w_j G(x; t_j; \Sigma) \right] D^* \left[ \sum_{k=1}^{M} w_k G(x; t_k; \Sigma) \right] \, dx \quad (18)$$

Noting that the Green's function $G(x; t_k; \Sigma)$ satisfies the partial differential equation [Courant and Hilbert, 1970].
\[ D^* D[G(x; t_k, \Sigma)] = \delta(x - t_k) \]  

we may write (18) as,

\[
B(w) = \frac{1}{2} \int \left[ \sum_{j=1}^{M} w_j G(x; t_j, \Sigma) \right] \left[ \sum_{k=1}^{M} w_k \delta(x - t_k) \right] dx
\]

Finally, using the sifting property of the Dirac delta function \( \delta(x - t_k) \) we have,

\[
B(w) = \frac{1}{2} \sum_{j=1}^{M} \sum_{k=1}^{M} w_j w_k G(t_k; t_j, \Sigma)
\]

Defining the \( M \times M \) matrix \( G_0 \) by,

\[
(G_0)_{k,j} = G(t_k; t_j, \Sigma)
\]

enables us to express the regularisation term in (13) as,

\[
B(w) = \frac{1}{2} w^T G_0 w
\]

Combining (14) and (23) we may write the regularised merit function (13) as,

\[
\min_{w, \psi(x)} \mathcal{S}(w, \psi(x)) = \frac{1}{2} [y - \psi - Gw]^T [y - \psi - Gw] + \frac{1}{2} w^T G_0 w
\]

In order to minimise (24) we require some knowledge of the null space term \( \psi(x) \) and three situations can be envisaged: a) the exact form of \( \psi(x) \) is known, b) the exact form of \( \psi(x) \) is not known but we may approximate it as a linear combination of a set of pre-specified basis functions, and c) we have no \textit{a priori} information about the form of \( \psi(x) \). The first two cases can be readily incorporated within the training machinery of Generalised RBF networks and are discussed here. The third case is more complex and its development is deferred to section 5.4.
In the simplest case we may know the exact form of $\psi(x)$, in which case the vector $\underline{\psi} = [\psi(x_1), \psi(x_2), \ldots, \psi(x_N)]^T$ is a known constant vector. We may then simply define the adjusted response vector $\tilde{y} = y - \underline{\psi}$ and train the network as in Chapter 4 using the adjusted $\tilde{y}$ in place of the measured overall response $y$. The overall prediction is obtained by adding back $\psi(x)$ to the output of the trained Generalised RBF network.

In a slightly more challenging situation we may not know the exact form of the null space term $\psi(x)$ precisely. We may, however, know or assume that the null space term is composed of a linear combination of a set of pre-specified basis functions,

$$\psi(x) = \sum_{k=1}^{L} \tilde{\psi}_k \psi_k(x)$$

(25)

where each individual $\psi_k(x)$ lies in the null space of the operator $D$ (i.e. $D[\psi_k(x)] = 0$, $k = 1, 2, \ldots, L$). We note here that we are completely free to choose the number of the null space terms $L$ and the form of each $\psi_k(x)$ as long as it belongs to the null space of the operator $D$. In this case, each element of the vector $\underline{\psi}$ is given by,

$$\psi(x_i) = \sum_{k=1}^{L} \tilde{\psi}_k \psi_k(x_i)$$

(26)

and we may write,

$$\underline{\psi} = \Psi \tilde{\underline{\psi}}$$

(27)

where $\tilde{\underline{\psi}} = [\tilde{\psi}_1, \tilde{\psi}_2, \ldots, \tilde{\psi}_L]^T$ and $\Psi$ is an $N \times L$ matrix with elements,

$$\Psi_{i,k} = \psi_k(x_i)$$

(28)
Substituting for $y$ from (27) into (24) yields,

$$\min_{w, \tilde{w}} \mathcal{S}(w, \tilde{w}) = \frac{1}{2} [y - \Psi \tilde{w} - Gw]^T [y - \Psi \tilde{w} - Gw] + \frac{1}{2} w^T G_o w$$  \hspace{1cm} (29)$$

Applying the optimality conditions $\frac{\partial \mathcal{S}}{\partial w} = 0$ and $\frac{\partial \mathcal{S}}{\partial \tilde{w}} = 0$ leads to the linear system,

$$\begin{bmatrix} G^T G & G^T \Psi \\
\vdots & \vdots \\
\psi^T G & \psi^T \Psi \\
\end{bmatrix} + \lambda \begin{bmatrix} G_o & 0 \\
\vdots & \vdots \\
0 & 0 \\
\end{bmatrix} \begin{bmatrix} \tilde{w} \\
\vdots \\
w \\
\end{bmatrix} = \begin{bmatrix} G^T \\
\vdots \\
\psi^T \\
\end{bmatrix} y$$  \hspace{1cm} (30)$$

which can be solved for any given $\lambda$ to yield the linear weights $w_\lambda$ and $\tilde{w}_\lambda$. To simplify the notation, we define the partitioned matrices,

$$\hat{G} = \begin{bmatrix} G \\
\psi \\
\end{bmatrix}$$  \hspace{1cm} (31)$$

$$\hat{G}_o = \begin{bmatrix} G_o \\
\vdots \\
0 \\
\end{bmatrix}$$  \hspace{1cm} (32)$$

and the partitioned vector

$$\bar{w} = [w_1, w_2, \ldots, w_N; \tilde{w}_1, \tilde{w}_2, \ldots, \tilde{w}_L]^T$$  \hspace{1cm} (33)$$

Equation (30) can then be represented in the compact form,

$$(\hat{G}^T \hat{G} + \lambda \hat{G}_o) \bar{w} = \hat{G}^T y$$  \hspace{1cm} (34)$$

and the optimum solution minimising (29) is given by,

$$\bar{w}_\lambda = \hat{G} \tilde{w}_\lambda$$  \hspace{1cm} (35)$$

where $\tilde{w}_\lambda$ satisfies (34). A chief advantage of this formulation is that we may select the appropriate level of regularisation automatically and efficiently by minimising the leave one out cross validation criterion using the generalised singular value decomposition of
matrices \( \mathcal{G} \) and \( \mathcal{G}_0 = \mathcal{Q}_0 \mathcal{A}_0 \mathcal{Q}_0^T \). The following illustrative example is used to demonstrate the improvement in the performance of Generalised RBF network obtained by the explicit incorporation of null space terms to capture the global trend.

5.3 – An illustrative example

To illustrate the performance of the Generalised RBF networks in the presence of the null space terms, we consider that the true surface is composed of two parts,

\[
F(x) = F_{\text{local}}(x) + F_{\text{global}}(x)
\]  

(36)

The local feature is taken the same as the example used in Chapters 3 and 4,

\[
F_{\text{local}}(x) = 100(1 - 3.3x_1 + 2.9x_1^2) \exp \left[ -\left( \frac{x_1 - 0.5}{0.25} \right)^2 \right] \\
(1 - 3.3x_2 + 2.9x_2^2) \exp \left[ -\left( \frac{x_2 - 0.5}{0.25} \right)^2 \right]
\]  

(37)

We note here that (36) has virtually no response outside the range \( 0 < x_1, x_2 < 1 \). The global contribution can be taken as any low frequency function, for example

\[
F_{\text{global}}(x) = -0.5x_1 + 0.3(x_2 - 0.5)^3
\]  

(38)

which has a response over the entire input domain. The training data points were generated on a uniform grid in the range of \(-1 < x_1, x_2 < 2\) and the noisy data were computed as,

\[
y(x_i) = F(x_i)_{\text{local}} + F_{\text{global}}(x_i) + \epsilon_i, \quad i = 1, 2, ..., N = 400
\]  

(39)
where the $\varepsilon_i$'s represent random deviates drawn from a Gaussian distribution with a spread of 0.2. Figures 5.2 and 5.3 show the true response and the 400 noisy training exemplars.

**Figure 5.2** – An illustrative bivariate example with distinct local and global features.

**Figure 5.3** – The true underlying function and the uniformly sampled noisy data.
5.3.1 – Generalised RBF network with precisely known null space term

For comparison purposes we shall first consider the performance of a standard Generalised RBF network without null space terms for the illustrative example of Figure 5.3 for a network with $M = 30$ neurones and an isotropic spread of $\sigma = 0.2$. The top panel in Figure 5.4 shows the result obtained when the centres are placed randomly over the entire input domain and the bottom panel is for all the centres confined to the region of the local feature. The standard Generalised RBF network performs inadequately in either case but for different reasons. With the centres distributed across the entire input domain, the response is highly oscillatory and neither the local nor the global feature is adequately captured. Placing all the centres within the local feature region captures the local feature reasonably well but misses the global trend completely.

The results in Figure 5.4 highlight the importance of the null space term $\psi(x)$ and the need for its direct inclusion in the training procedure. This is readily achieved for the case where $\psi(x)$ is known precisely. All that is needed is to subtract the known global contribution from the data and train a standard Generalised RBF network on the adjusted response vector. Figure 5.5 shows the prediction obtained by training a standard Generalised RBF network using $\tilde{y}_i = y(x_i) - [ -0.5x_{1,i} + 0.3(x_{2,i} - 0.5)^3 ]$, $i = 1, ..., N$. The bottom panel in Figure 5.5 shows that subtracting an exactly known null space term leads to excellent separation of the local and global features provided that the centres are chosen appropriately. We do not pursue the optimisation of the centres or spreads in this chapter. However, the sequential back-fit algorithm presented in Chapter 4 may provide a means for the positioning of a sufficient number of centres within the local feature region.
Figure 5.4 – Generalisation performances of a standard Generalised RBF network (σ = 0.2).

(a): $M = 30$ centres placed across the input domain

(b): $M = 30$ centres confined to the local feature region
Figure 5.5 – Generalisation performances of a standard Generalised RBF network trained against an adjusted response for precisely known $\psi(x)$ with $\sigma = 0.2$. The noisy data and the true underlying surface are shown in Figure 5.3.

(a) $M = 30$ centres placed across the input domain

(b) $M = 30$ centres confined to the local feature region
5.3.2 – Generalised RBF network with assumed bases for the null space term

In most practical situations we do not know the exact form of the null space term \( \Psi(x) \) but could attempt to approximate it by a linear combination of assumed bases. For example we may choose to represent the null space term \( \Psi(x) \) as the linear superposition of a series of univariate polynomials, one for each input dimension. Collecting all the constant term together we may express

\[
\psi(x) = \tilde{w}_0 + \sum_{i=1}^{p} \sum_{k=1}^{L_i} \tilde{w}_{k,i} x_i^k
\]  

(40)

We note here that we may choose a different number of basis functions \( L_i \) for each input dimension. The constant term \( \tilde{w}_0 \) in (40) represents the overall bias term which can not be broken down into the contribution from the different dimensions.

The generalisation performance of a hybrid Generalised RBF network with \( M = 30 \) neurones with an isotropic spread of \( \sigma = 0.2 \) for the illustrative example of Figure 5.3 are compared in Figure 5.6. In all cases, the null space term was described by (40) with \( L_1 = L_2 = 3 \). The results in the top panel of Figure 5.6 were obtained with the centres selected at random across the entire input domain and those in the bottom panel are for the centres confined to the local feature region. Figure 5.6 shows that choosing a set of univariate polynomial bases to approximate the null space term leads to excellent result provided the centres are confined to the local feature region. We note that for an inappropriate choice of centres, the Generalised RBF network is incapable of distinguishing the local and global features clearly. The optimum level of regularisation has a minor effect compared to the position of the centres.
Figure 5.6 – Performance of Generalised RBF networks with univariate polynomial bases as the null space terms ($L_1 = L_2 = 3$, $M = 30$, $\sigma = 0.2$).

The noisy data and the true surface are shown in Figure 5.3.
(a): centres selected across the input domain
(b): centres confined to the local feature region
We should point out here that the success of the Generalised RBF network using polynomial bases for the illustrative example of Figure 5.3 is not entirely surprising. The true global feature in this example is first order in $x_1$, and third order in $x_2$ and readily described by 3rd order polynomial. The performance will inevitably deteriorate if there is significant input interactions in the global term or when the true global feature is not well approximated by polynomial bases. Figure 5.7 shows a second illustrative example which has the same local feature given by,

\[
F_{local}(x) = 100(1 - 3.3x_1 + 2.9x_1^2)\exp\left[-\left(\frac{x_1 - 0.5}{0.25}\right)^2\right] \\
(1 - 3.3x_2 + 2.9x_2^2)\exp\left[-\left(\frac{x_2 - 0.5}{0.25}\right)^2\right]
\]

but with the global term taken as a sum of two univariate exponentials:

\[
F_{global}(x) = \exp(-1.2x_1) + 0.02\exp(-5x_2)
\]  \hspace{1cm} (41)

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure5.7.png}
\caption{The True surface and 400 noisy exemplars for an illustrative example with exponential global terms.}
\end{figure}
Figure 5.8 shows the generalisation performance of a Generalised RBF network with $M = 30$ neurones and an isotropic spread of 0.2 for the example of Figure 5.7, using univariate polynomial bases of up to order 3. With the centres distributed across the entire input domain (top panel) the surface recovered is oscillatory and shows a number of false local features. The results obtained with the centres confined to the region of the local feature (bottom panel) are surprisingly good. However, close examination shows that the global features show a systematic deviation from the true surface. This is because the global variation in the $x_2$ dimension is very sharp and not easily captured with a third order polynomial. Better performance could perhaps be obtained with higher order polynomial bases at the expense of a larger network. A better approach is developed in the next section by resorting to an adaptive additive model which enables the null space term in each input dimension to evolve in the course of training.
Figure 5.8— Performances of Generalised RBF networks for example of Figure 5.7, using third order univariate polynomial bases to describe the null space term ($M = 30$, $\sigma = 0.2$).

(a): centres selected across the input domain
(b): centres confined to the local feature region
5.4 – An Adaptive Generalised RBF network

In this section we develop an algorithm which enables the form of the null space terms to emerge automatically in the course of the training process. The procedure is developed under the assumption that there is no significant input interactions in the global trend. For an input vector of dimension $p$, we may then represent the null space term as,

$$
\psi(x) = \sum_{l=1}^{p} \psi_l(x_l)
$$

(42)

where each $\psi_l(x_l)$ is an arbitrary univariate function in the input dimension $x_l$. No assumptions are made regarding the form of the functions $\psi_l(x_l)$ other than it lies in the null space of the operator $D$. The approximating function is therefore taken as,

$$
\bar{F}_\lambda(x) = \sum_{j=1}^{M} w_j G(x, t_j, \Sigma) + \sum_{l=1}^{p} \psi_l(x_l)
$$

(42)

The first term in (42) is aimed at capturing the local features embedded in the data and the second must account for the global background. For a given set of non-linear parameters (i.e. centres and spreads), the regularisation functional to be minimised can be expressed as,

$$
\min_{\bar{F}_\lambda, \psi_l, l=1,...,p} \mathfrak{F}(\bar{F}_\lambda) = \frac{1}{2} \sum_{i=1}^{N} (y_i - \bar{F}_\lambda(x_i))^2 + \frac{1}{2} \lambda \| D[\bar{F}_\lambda(x)] \|^2
$$

(43)

Substituting for $\bar{F}_\lambda(x)$ from (42) into (43) and noting that $D[\psi_l(x_l)] = 0$, $l = 1,...,p$, leads to,
We must now find a way for selecting the functions \( \psi_l(x_l) \) among the infinite number of possibilities. This is achieved by insisting that the functions \( \psi_l(x_l) \) should satisfy some \textit{a priori} regularisation constraint, one for each \( \psi_l(x_l) \). In general, the global variations are gentle and of low frequency and it is therefore appropriate to penalise the form of each univariate function \( \psi_l(x_l) \) with,

\[
B_l(\psi_l(x_l)) = \int_{x_l} \left[ \frac{d^2 \psi_l(x_l)}{dx_l^2} \right] dx_l, \quad l = 1, 2, \ldots, p
\]  

(45)

This constraint in effect penalises the excessive curvature and should lead to smoother functions. We can add the constraint (45) with its own regularisation parameter \( \gamma_l \) for \( l = 1, 2, \ldots, p \) (we use \( \gamma_l \) here to avoid confusion with \( \lambda \)). The penalised regularisation functional takes the form,

\[
\min \sum_{w, \psi_l, l=1,\ldots,p} \mathcal{I}(\tilde{F}_\lambda) = \frac{1}{2} \sum_{i=1}^{N} \left( y_i - \sum_{l=1}^{p} \psi_l(x_{l,i}) - \sum_{j=1}^{M} w_j G(x_{l,i}, t_j, \Sigma) \right)^2 + \frac{1}{2} \lambda \left\| D \left[ \sum_{j=1}^{M} w_j G(x_{l,i}, t_j, \Sigma) \right] \right\|^2 + \frac{1}{2} \sum_{l=1}^{p} \gamma_l \left[ \frac{d^2 \psi_l(x_l)}{dx_l^2} \right]^2 dx_l
\]  

(46)

Next we must decide on suitable bases in order to parameterise the functions \( \psi_l(x_l) \). We could do this by using any set of bases, for example polynomial bases as in section 5.3.2. However, there are a number of inherent difficulties with the use of simple polynomials.
First, the appropriate order of the polynomials is not available \textit{a priori}. We could of course test a range of orders for each particular case and then somehow decide on the best order. Second, the data may not be particularly well fitted by a polynomial form. We may then have to resort to higher order polynomials to obtain close agreement between data and the model predictions. Higher order polynomials may fit the data closely but will almost always show large oscillations between the data points, such behaviour is to be avoided when the model function is intended for interpolation. Most significantly, a polynomial basis function has a contribution over the entire range of \( x_i \). This is unproductive because the data may be well fitted by a particular polynomial basis function over a limited range but it is rarely well fitted over the entire range. Much greater flexibility is achieved if the contribution of each basis function to \( \psi_i(x) \) is limited to a specific range and is zero outside this range. A useful example of functions with compact support is B-spline functions which have desirable computational properties and are particularly well suited to the task of univariate fitting, interpolation and smoothing.

5.4.1 - B-splines

B-splines were first introduced by Schoenberg [1946] and their properties were carefully considered by De Boor [1972]. This section highlights the important properties of B-splines. Consider a data set \((x_{i,i}, y_i; \ i=1,2,...,N)\) with the \( x_{i,i} \) sorted in ascending order and covering the interval \( x_{\min} = x_1 \leq x_j \leq x_{N} = x_{\max} \). B-splines are defined
with respect to a grid generated by placing \( \mu \) knots, \( T_k, \ k = 1, \ldots, \mu \) within the data interval as shown below,

\[
\begin{array}{ccccccccccccccc}
& & & & x_{i_{\min}} & T_1 & T_2 & T_3 & T_4 & T_5 & \ldots & T_\mu & x_{i_{\max}} & & \\
\end{array}
\]

The B-spline function of order one, denoted as \( B_k^1(x_i) \), is a discontinuous function defined over knots \( T_k \) and \( T_{k+1} \) as,

\[
B_k^1(x_i) = \begin{cases} 
1 & \text{if } T_k < x_i < T_{k+1} \\
0 & \text{otherwise} 
\end{cases}
\]

Higher order B-splines can be generated by the following recurrence relation from the lower order B-splines,

\[
B_k^q(x) = \left(\frac{x_i - T_k}{T_{k+q} - T_k}\right) B_k^{q-1}(x_i) + \left(\frac{T_{k+q} - x_i}{T_{k+q} - T_{k+1}}\right) B_{k+1}^{q-1}(x) \quad q \geq 2; \ k = 1, \ldots, \mu \quad (47)
\]

For example, the second order B-spline, \( B_k^2(x_i) \), is a piecewise linear function defined over knots \( T_k, T_{k+1} \) and \( T_{k+2} \):

\[
B_k^2(x_i) = \begin{cases} 
\frac{x_i - T_k}{T_{k+1} - T_k} & \text{if } T_k \leq x_i < T_{k+1} \\
\frac{T_{k+2} - x_i}{T_{k+2} - T_{k+1}} & \text{if } T_{k+1} \leq x_i < T_{k+2} \\
0 & \text{otherwise} 
\end{cases}
\]
The B-spline of order \( q \), \( B_k^q(x_l) \), is defined over the knots \( T_k, \ldots, T_{k+q} \) and is a piecewise continuous function pieced together from polynomials of order \( q-1 \) and so has continuous derivatives up to order \( q-2 \). The derivatives of B-splines can be recursively computed from,

\[
\frac{d}{dx} B_k^q(x_l) = \left( \frac{q-1}{T_{k+q-1} - T_k} \right) B_k^{q-1}(x_l) - \left( \frac{q}{T_{k+q} - T_{k+1}} \right) B_{k+1}^{q-1}(x_l)
\]  

(48)

The actual shape of a B-spline depends on the knot spacing used to define it.

For the case of a \( q \) order B-spline, we may parameterise the function \( \psi_l(x_l) \) with the linear sum,

\[
\psi(x_l) = C_{l,1} B_{l,1}^q(x_l) + C_{l,2} B_{l,2}^q(x_l) + \ldots + C_{l,\mu+q} B_{l,\mu+q}^q(x_l)
\]

\[
= \sum_{k=1}^{\mu+q} C_{l,k} B_{l,k}^q(x_l)
\]  

(49)

where, \( B_{l,k}^q(x_l) \) is the \( q^{th} \) order B-spline basis function and \( C_{l,k}, l = 1, \ldots, p, k = 1, \ldots, \mu + q \) are the model parameters to be determined by minimising a suitable merit function. The grid over which the B-splines are defined is established by placing \( \mu \) interior knots strictly within the data interval \( x_{l_{\text{min}}} = x_{l1} \leq x_l \leq x_{l_N} = x_{l_{\text{max}}} \) augmented by \( q \) additional coincident knots inserted at each end of the data interval. The number of parameters in the model is therefore \( \mu + q \) and the number of knots in the grid is \( \mu + 2q \). The arrangement and numbering of the knots is shown below:
The above arrangement is the most compact form for which \( \psi_l(x_l) \) is fully defined over the entire data range \( x_{l_{\text{min}}} = x_l \leq x_l = x_{l_{\text{max}}} \) in terms of \( \mu \) interior knots and the \( \mu + q \) parameters \( C_{l,k}, l = 1, ..., p, k = 1, ..., \mu + q \). Within this arrangement we are free to choose the order of the B-splines, \( q \), and the number and position of the interior knots \( \mu \). This gives the B-spline model considerable flexibility for function approximation purposes. In most practical applications we seek continuity of the function and its first two derivatives and 4^{th} order B-splines \( (q = 4) \) suffice for most cases. We also note here that for a point \( x_{l_i}, T_{l,k} \leq x_{l_i} \leq T_{l,k+1} \) we can evaluate \( \psi_l(x_{l,i}) \) using only \( q \) coefficients.

\[
\psi_l(x_{l_i}) = \sum_{j=k}^{k+q} C_{l,j} B_{l,j}^q(x_{l_i})
\]

since \( B_{l,m}^q(x_{l_i}) = 0 \) for \( m < k \) and \( m > k + q \) by construction, which gives B-splines considerable computational speed advantages. With the functions \( \psi_l(x) \) parameterised using (49), we can minimise (46) with respect to \( w \) and \( C_{l,1}, C_{l,2}, ..., C_{l,p} \) to find the \( \tilde{F}_\lambda(x) \) from,

226
\[ \bar{F}_\lambda(x) = \sum_{j=1}^{M} w_j G(x, t_j, \Sigma) + \sum_{l=1}^{p} \sum_{k=1}^{\mu+q} C_{l,k} B_{l,k}^q(x_l) \]  

(51)

We also note that a B-spline of a finite order \( q \) has at most \( q-2 \) continuous derivatives and therefore lies in the null space of the operator \( D \) associated with a Gaussian Green function. Evidently, a linear combination of such B-splines will also automatically reside in the null space of the operator \( D \).

5.4.2 – Estimation of the coefficients

With the above preliminaries we can now consider the calculation of the unknown coefficients \( w = [w_1, w_2, \ldots, w_M]^T \) and \( C_l = [C_{l,1}, C_{l,2}, \ldots, C_{l,\mu+q}]^T, \ l = 1, 2, \ldots, p \).

Substituting for \( \psi_l(x_l) \) from (49) into (46) leads to,

\[
\min_{w, \psi_l, l=1, \ldots, p} \mathcal{I}(\bar{F}_\lambda) = \frac{1}{2} \sum_{l=1}^{N} \left( y_l - \sum_{l=1}^{p} \sum_{k=1}^{\mu+q} C_{l,k} B_{l,k}^q(x_l) - \sum_{j=1}^{M} w_j G(x_l, t_j, \Sigma) \right)^2
\]

(52)

\[
+ \frac{1}{2} \sum_{l=1}^{p} \left[ \frac{d^2 \sum_{k=1}^{\mu+q} C_{l,k} B_{l,k}^q(x_l)}{dx_l^2} \right] dx_l
\]

We may write the merit function (52) as,

\[
\min_{w, \psi_l, l=1, \ldots, p} \mathcal{I}(\bar{F}_\lambda) = \frac{1}{2} \sum_{l=1}^{N} \left( y_l - \sum_{l=1}^{p} \sum_{k=1}^{\mu+q} C_{l,k} B_{l,k}^q(x_l) - \sum_{j=1}^{M} w_j G(x_l, t_j, \Sigma) \right)^2
\]

(53)

\[
+ \frac{1}{2} \sum_{l=1}^{p} \left[ \frac{d^2 \sum_{k=1}^{\mu+q} C_{l,k} B_{l,k}^q(x_l)}{dx_l^2} \right] dx_l
\]

\[ \Omega_{l,m,n} \text{ is given by}, \]

227
\[ \Omega_{m,n} = \int_{x_{\min}}^{x_{\max}} \left( \frac{d^2 B_m(x_j)}{dx_j^2} \right) \left( \frac{d^2 B_n(x_j)}{dx_j^2} \right) dx_j, \quad m = 1, \mu + q, \quad n = 1, \mu + q \]  

(54)

For a given set of regularisation parameters \( \lambda, \gamma_1, \ldots, \gamma_p \), the coefficients \( w_\lambda \) and \( C_{\gamma_l}, l = 1, \ldots, p \) may be readily calculated from (54). The estimation of the appropriate levels of regularisation is, however, less straightforward and two alternatives are possible. We could extend the leave one out cross validation technique described in section 2.6 to cover the present situation which involves multiple regularisation parameters involved for different purposes. This would, however, result in a multidimensional cross validation criterion, \( CV(\lambda, \gamma_1, \ldots, \gamma_p) \), whose minimisation with respect to \( \lambda \) and \( \gamma_l, l = 1, \ldots, p \) is not a simple task. In an alternative approach, we may take advantage of the back-fit procedure and establish the optimal regularisation parameters \( \lambda \) and \( \gamma_l, l = 1, \ldots, p \) one at a time.

In the first step, all null the space terms \( \psi_l(x_j) \) are assumed fixed and the optimum level of regularisation \( \lambda^* \) for the Generalised RBF network and the corresponding \( w_{\lambda^*} \) are computed by minimising,

\[
\min_{w} \mathcal{J} (\tilde{\mathbf{F}}_\lambda) = \frac{1}{2} \sum_{l=1}^{N} \left( y_l - \sum_{l=1}^{\mu+q} C_{l,k} B_{l,k}(x_l) - \sum_{j=1}^{M} w_j g(x_l, t_j, \Sigma) \right)^2 + \frac{1}{2} \lambda \left\| D \left[ \sum_{j=1}^{M} w_j g(x_l, t_j, \Sigma) \right] \right\|^2
\]  

(55)
In subsequent steps, the optimum level of regularisation $\gamma_i^*$ for each null space term and the corresponding linear coefficients $(C_i)^*$ are computed in turn by repeated minimisation of,

$$
\min_{C_i} \mathcal{J} (\hat{f}_i) = \frac{1}{2} \sum_{l=1}^{N} \left( (\eta)_l - \sum_{k=1}^{\mu+q} C_{l,k} B_{l,k}^q (x_l) \right)^2 + \frac{1}{2} \gamma_i \sum_{m=1}^{\mu+q} \sum_{n=1}^{\mu+q} C_{l,m}^* C_{l,n}^* \Omega_{l,m,n}
$$

where,

$$(\eta)_l = y_l - \sum_{i=1}^{p} \sum_{k=1}^{\mu+q} C_{i,k} B_{i,k}^q (x_l) - \sum_{j=1}^{M} \sum_{i=1}^{M} w_j G_k^{l,j} (x_l, x_i, \Sigma)$$

This enables us to use the univariate leave one out cross validation criterion of section 2.6 without any modification. The entire back-fit algorithm can be summarised as follows,
Training Algorithm for adaptive Generalised RBF network

S1) Select the number of neurones $M$ and specify the centres and spreads

S2) Initialise $\psi_l(x_l) = \psi_l^0(x_l), \ l = 1, 2, ..., p$

S3) Start {outer back-fit loop}

\[
\begin{align*}
&n = 0 \\
&\text{Repeat} \\
&\text{Find } \lambda_n^m \text{ and } w_n^m \text{ via (56) with the } \psi_l^0(x_l) \text{ considered as known null space terms.} \\
&m = 0 \\
&\text{Repeat \{inner back-fit loop\}} \\
&\text{Do } l = 1, ..., p \\
&\text{Find the residuals } (r_l^m)_l^m = y_l - \sum_{\substack{i=1 \\|i\|\neq l}}^{p} \sum_{k=1}^{\mu+q} C_{\|i\|,k} B_{\|i\|,k}^0 (x_i_l) - \sum_{j=1}^{M} w_{\lambda_j}^m G(x_i_l, t_j) \\
&\text{Find } \gamma_l^m \text{ and } C_{\gamma_l^m} \text{ via minimising (57)} \\
&\text{Continue} \\
&m = m + 1 \\
&\text{Until each individual function } \psi_l(x_l)'s \text{ does not change (inner back-fit loop)} \\
&n = n + 1 \\
&\text{Until overall termination criteria are satisfied (outer back-fit loop)}
\end{align*}
\]
Figure 5.9 shows the performance of an adaptive Generalised RBF network (with $M = 30$ neurones and $\sigma = 0.2$) trained by the back-fit algorithm for the illustrative example of Figure 5.7. We note that the exponential global features captured well in both input directions without any *a priori* assumption about the form of the null space term. With the centres distributed across the entire input domain, the reconstructed surface although quite faithful to the true underlying trend, shows a number of false local features. Confining all the centres within the local feature region removes such ripples and produces an excellent reproduction of the underlying surface with both the local and global features accurately captured.

**Figure 5.9** – Performance of the back-fit algorithm for training of an adaptive Generalised Generalised RBF networks for illustrative example of Figure 5.7 ($\sigma = 0.2$).

(a) $M = 30$ centres selected across the entire training domain

(b) $M = 30$ centres confined to the local feature region
5.5 – Conclusions

This chapter was aimed at highlighting the significant influence of the null space terms for the separation of the local and global features embedded in a noisy data set. Existing multivariate regression techniques as well as the conventional neural networks are aimed at filtering the noise and recovering the overall response which is the combination of the global background and the local features. Evidently, separation of these dual features can provide further insight about the governing mechanisms of the underlying process but few researchers have paid attention to this issue.

The ultimate aim of this chapter was to produce a neural network containing two types of neurone: i) a set of projection based neurones, which were better suited for capturing the gently varying global background, and ii) a set of localised kernel based neurones which were more appropriate for the recognition of the sharp local features. A simple approach was presented to de-couple the contribution of the projection based neurones from the kernel based neurones by insisting that all the activation functions of the projection based neurones should lie in the null space of the differential operator generating the activation function of the kernel based neurones.

A general consideration of various procedures for the inclusion of the null space terms in the training procedure of the Generalised RBF networks under a variety of assumptions was considered and an illustrative example was employed to highlight the important influence of the null space terms on the performance of the Generalised RBF networks. An initial attempt was made in this chapter to develop an Adaptive Generalised
RBF network to separate the low frequency global background from the high frequency local features. A new adaptive back-fit training algorithm was developed which enabled the null space terms to evolve automatically. It was demonstrated that the proposed algorithm performs well in the absence of strong global input interactions. To our knowledge, the inclusion of the adaptive null space terms in the training of the Generalised RBF networks has not been previously reported.

The illustrative examples presented in this chapter were deliberately chosen without input interactions in the global trend. The new algorithm must be modified to account for strong global input interactions. This is demonstrated by adding a \(-0.5x_1x_2\) term to the true global trend of Figure 5.7, the overall global feature is therefore,

\[
F_{\text{global}}(x) = \exp(-1.2x_1) + 0.02\exp(-5x_2) - 0.5x_1x_2
\]  

(57)

A set of 400 noisy data was generated from (37) and (57) and contaminated with the Gaussian noise of spreads 0.2. The right panel of Figure 5.10 shows the generalisation performance of back-fit algorithm for training the adaptive Generalised RBF networks when applied to this new problem. At the first glance, the performance may seem quite impressive but closer comparison of the predictions with the true surface reveals that the fitted model has practically ignored the term \(-0.5x_1x_2\) which contains the input interactions in the global background. Replacing the adaptive additive model with other more advanced adaptive methods such as the Projection Pursuit Regression method [Friedman and Stuetzle, 1981] which is capable of accounting for input interactions may lead to better results and is a useful area for future work.
Figure 5.10 – Performance of the Adaptive Generalised RBF network ($\sigma = 0.2$) for the problems without (left panel) and with (right panel) global input interactions.
(a) Truth
(b) $M = 30$ centres across the input domain
(c) $M = 30$ centres confined to the local feature region.
Chapter 6

Conclusions and Suggestions for Future Work

6.1 - Conclusions

The application of neural networks for the solution of practical problems has shown an exponential growth over the last decade. Most applications have, however, concerned problems with very large and dense data sets, for example speech and vision processing, pattern recognition and time series analysis. The training algorithms developed for such networks are not well suited for problems involving small to moderate data sets. Data acquisition is usually expensive and time consuming in chemical engineering and the amount of data available is often limited. This thesis was aimed primarily at the development of neural networks to describe the non-linear relationship between the inputs and the outputs of a stationary process based on a limited number of observations. This is a task of reconstructing a multidimensional hyper-surface and is well suited to feed-forward neural networks.

Feed-forward neural networks may be constructed using either kernel or projection based neurones. The kernel based neurones with localised receptive field are more suitable for capturing the high frequency local features while the projection based neurones with response over the entire input domain are well suited to extract low frequency global background. In this study we focused exclusively on a special class of neural networks known as Radial Basis Function (RBF) networks which have
a well developed and solid theoretical foundation [Poggio and Girosi, 1990a&b]. The theory of such networks was naturally extended in this thesis for separation of local and global features.

The broad classification of artificial neural networks and a brief review of their development was presented in Chapter 1. From a mathematical point of view, the feed-forward neural networks considered in this study bear a close relationship and share many aspects of multivariate function approximation theory. The primary aim of Chapter 2 was to highlight the importance of regularisation and the need for stabilisation in the handling of noisy data. The basic ideas were most clearly introduced in a simple univariate setting. Various stabilisation techniques such as singular value decomposition, ridge regression, penalised least square (all of which require a specific parametric model) were considered. The selection of the appropriate level of stabilisation in such methods is highly problem dependent and may therefore prove subjective. In addition, these methods are not readily extendable to problems with multivariate inputs. The linear regularisation theory, which does not require a specific model of the process, and provides the most effective means of handling the noise, was introduced in section 2.5 and illustrated for a univariate problem. The linear regularisation method can be naturally extended to multivariate problems.

The most important issue in any regularisation method is the selection of the optimum level of regularisation which may prove highly subjective. A convenient and obvious procedure for large data sets is to set aside a sufficient number of data points exclusively for a posteriori validation purposes. This proves ineffective for problems with limited observations, a situation frequently encountered in chemical engineering.
A promising approach, the leave one out cross validation (CV) technique which makes full use of all the available data was described in section 2.6. An efficient procedure based on the generalised singular value decomposition (GSVD) was developed which can be used for a wide variety of linear stabilisation operators. The performance of the proposed procedure was assessed for the recovery of the unknown energy distribution of a heterogeneous solid adsorbent. To our knowledge the combination of the leave one out cross validation technique and generalised singular value decomposition has not been previously reported and was used to good effect throughout this study.

Chapter 3 dealt exclusively with the multivariate linear regularisation problem and it was shown that the exact solution to this problem can be represented in terms of a single hidden layer feed-forward neural network labelled as the Regularisation network [Poggio and Girrosi, 1990a&b] with radial basis activation functions. The number of the hidden layer neurones of the Regularisation network is equal to the number of training exemplars and the centres of each radial basis function coincides with a particular data point. The widths and the orientations of the receptive fields of the radial basis functions, in particular Gaussian basis functions, are controlled by the choice of the spread(s). In the majority of the previous applications, Regularisation networks have been used with Gaussian basis functions with a constant isotropic spread. Training of such networks requires the optimisation of the regularisation level and the calculation of the synaptic weights for a given isotropic spread $\sigma$.

A new working equation was derived in section 3.3 for estimation of the optimal level of regularisation $\lambda^*$ based on the minimisation of the cross validation criterion.
An illustrative example was used in section 3.4 to demonstrate that the value of the isotropic spread $\sigma$ has a crucial effect on the performance of the Regularisation network, an obvious point which has received little attention to date. It was also shown that the effective degrees of freedom, $df$, of a Regularisation network is a function of the regularisation level $\lambda$ and the isotropic spread $\sigma$, and the optimum level of regularisation is highly correlated with the value of the isotropic spread $\sigma$. A convenient measure of the approximate degrees of freedom was used to establish a procedure for selecting the appropriate value of the isotropic spread $\sigma^*$. It was illustrated that the proposed procedure leads to a significant improvement in the performance of a Regularisation network with isotropic Gaussian basis functions.

The computational cost of a Regularisation network grows as $N^3$ where $N$ is the number of training exemplars, this is a direct consequence of the one-to-one correspondence between the number of the hidden neurones and the number of distinct data points. To ease the computational burden it is necessary to break the one-to-one correspondence by using a smaller number of neurones $M << N$, such networks have been labelled as Generalised RBF networks by Poggio and Girosi [1990a&b]. The reduction in the number of hidden neurones inevitably limits the approximation power of the network and the training of a Generalised RBF network reduces to a challenging large-scale non-linear optimisation problem which has received considerable attention over the past decade. Chapter 4 started with a thorough review of the literature to place this study in the context of the previous work.
The relevant aspects of multivariate linear regularisation theory were used to develop the working equations of a Generalised RBF network. For a given network size ($M$), the training of a Generalised RBF network requires the calculation of the synaptic weights, estimation of the centres and spreads of the radial basis activation functions and the optimisation of the level of regularisation. Given the centres, spreads and the level of regularisation, the calculation of the linear synaptic weights reduces to the solution of an over-determined set of linear equations which can be handled by a variety of efficient and stable techniques. The combination of the leave one out cross validation criterion and the generalised singular value decomposition as described in Chapter 2 coupled with the use of similarity transformation established the optimum level of regularisation. The major difficulty is therefore the optimisation of the centres and spreads which appear non-linearly.

A number of heuristics have been proposed to circumvent the non-linear optimisation altogether. An illustrative example was used in Chapter 4 to highlight the limited reliability of the heuristic approach and the need for a direct solution of the non-linear optimisation problem. Several procedures such as the unsupervised $k$-means clustering algorithm [Moody and Darken, 1989] and the supervised Orthogonal Least Square method [Chen at al, 1991] have been developed to optimise the location of the centres of a Generalised RBF network with fixed spreads. In contrast, the optimisation of the spreads which determines the width and the orientation of the receptive field of the neurones, has received little attention to date. In applications involving limited and sparse data sets, such as those frequently encountered in chemical engineering problems, networks have a small number of neurones and the proper selection of the spreads is crucial to the performance of such small networks.
Chapter 4 emphasised on the calculation of the optimal non-linear parameters of a Generalised RBF network. In particular, a sequential approach was presented in which the neurones were optimised one at a time. Two alternatives were considered, in the orthogonal sequential approach a neurone once optimised remained fixed. The performance of an orthogonal sequential algorithm for training the Generalised RBF networks was investigated for an illustrative example. Freezing the optimised neurones may lead to a sub-optimal network and a back-fit procedure was adopted to circumvent this problem. In particular, a novel sequential back-fit algorithm was developed which enabled the optimisation to proceed one neurone at a time without freezing the previously optimised neurones. The new algorithm was tested with very promising results and its application to a simple chemical engineering process was also considered.

In some applications the overall response is composed of sharp localised features superimposed on a slowly varying global background. Chapter 5 was aimed at highlighting the significant influence of the null space terms for the separation of the local and global features embedded in a noisy data set. The ultimate aim is to produce a neural network containing two types of neurone: i) a set of projection based neurones, which are better suited for capturing the gently varying global background, and ii) a set of localised kernel based neurones which are more appropriate for the recognition of the sharp local features. A simple approach was presented to de-couple the contribution of the projection based neurones from the kernel based neurones by insisting that all the activation functions of the projection based neurones should lie in the null space of the differential operator generating the activation function of the kernel based neurones.
Various procedures were considered for the inclusion of the null space terms in the training procedure of the Generalised RBF networks under a variety of assumptions. An illustrative example was employed to highlight the important influence of the null space terms on the performance of such networks. An initial attempt was also made to develop an Adaptive Generalised RBF network to separate the global and local features. A new adaptive back-fit training algorithm was developed which enabled the null space terms to evolve automatically. It was demonstrated that the proposed algorithm performs extremely well in the absence of strong global input interactions. To our knowledge, the inclusion of the adaptive null space terms in the training of the Generalised RBF networks has not been previously reported.

6.2 – Suggestions for Future Work

In the context of what we have done so far in this thesis, three major tasks remain and may be suggested for future work.

1) A novel sequential back-fit algorithm with promising results was presented in Chapter 4. A second order optimisation technique based on the Modified Newton method [Gill and Murray, 1997] was employed to optimise the non-linear parameters at each stage of the sequential back-fit procedure. More robust optimisation methods such as Genetic Algorithm (GA) may improve the reliability of the sequential back-fit algorithm further.
2) A serious limitation of the back-fit algorithm presented in Chapter 5 for training the Adaptive Generalised RBF networks is that the proposed method does not account for global input interactions explicitly. This may deteriorate the performance of the adaptive Generalised RBF network in the presence of strong input interactions in the global trend. The inclusion of global interaction terms provides a major area for future work. A potential approach is to use Projection Pursuit Regression [Friedman and Stuetzle, 1981] in place of adaptive additive models. Finally,

3) The illustrative examples presented in Chapter 5 illustrate that a method should be developed to assign a sufficient number of centres to each local region. The development of suitable algorithms for this purpose is another challenge for future.
Appendix A

Applications of Neural Networks in Chemical Engineering

The following section presents a concise survey of the articles published to date on application of neural networks in chemical and biochemical engineering. With a few exceptions, the emphasis has been on the application of the standard neural network programs as a tool for solving various engineering problems.

In the late eighties, a few scattered efforts were carried out to employ the learning and approximation capabilities of neural network to solve specific problems in control and fault diagnosis of chemical engineering processes. Himmelblau and Hoskins [1988] employed a Zone Node Layer (ZNL) artificial neural network approach for the fault diagnosis of a process consisting of three CSTR's in series. Venkatasubramanian and Chan [1989] used a binary-input network to diagnose faults of a fluidised catalytic cracking (FCC) process. Watanabe et al [1989] presented a network architecture to estimate the degree of failure for a system with three measurements and five faults.

The 1990's began with the pioneering work of Bhat and McAvoy [1990]. They used a novel multilayer feed-forward neural network with back-propagation learning as a modelling tool to control the pH in a CSTR. In the same year, Mavrovouniotis [1990] summarised previous works on neural networks with especial emphasis on fault
detection of chemical processes. In the same year, Thibault et al [1990] employed artificial neural networks for online prediction of fermentation variables.

In 1991, Psichogios and Ungar [1991] presented a stand-alone neural network approach for modelling direct and indirect based control problems. In the next year they introduced the concept of Hybrid neural networks by combining a priori knowledge about a fed-batch bio-reactor with a neural network which served as an estimator of unmeasured process parameters difficult to model from first principles [Psichogios and Ungar, 1992].

Reuter et al [1993] illustrated the application of neural networks to identify the process conditions and dynamic simulation of batch and continuous processing systems. Thompson and Kramer [1994] presented other hybrid models for synthesising chemical engineering processes. They introduced modular and semi-parametric approaches to combine prior knowledge and neural networks. In their approach, prior knowledge enters the hybrid model as a simple process model based on first principle equations. This simple model augments the predictions of the hybrid network in the regions of the input space where the training data is not available. As an industrial application, Chan and Nascimento [1994] applied direct industrial data to a neural network to simulate the performance of olefin polymerisation in high-pressure tubular reactors.

Bulsari [1995] collected several articles in his book which covers a broad spectrum, ranging from introductory material (chapters 1-4,19) to complex applications of neural

Karjala and Himmelblau [1996] used a recurrent neural network coupled with an extended Kalman filter for data rectification in process environments with badly autocorrelated measurement errors. Galvan \textit{et al} [1996] employed neural networks to approximate the reaction rate constant from a large set of experimental data for the following reactions:

a) Homogeneous esterification of propionic anhydride and 2-butanol catalysed by sulphuric acid,

b) Heterogeneous liquid-liquid mono-nitration of toluene by mixed acids.

fixed bed reactor to train the recurrent neural network. Zorzetto and Wilson [1996] proposed a hybrid strategy to predict concentrations of bio-mass and substrate for on line control of *Saccharomyces cerevisiae* production. Single hidden layer networks were used in the ANN section.

Kim *et al* [1997] used a radial basis function (RBF) network to linearise the relation between the output of the linear controller and the process output for non-linear continuous stirred tank reactor and a pH process with linear controllers. Shaw *et al* [1997] reported that recurrent dynamic neural networks performs well for predicting non-linear asymmetric dynamic response for a first order, exothermic, irreversible reaction in a well-mixed reactor. Krishnapura and Jutan [1997] proposed a new back propagation approach for training neural networks containing Auto Regressive Moving Average (ARMA) neurones. These new neurones which depend on both their past inputs as well as their present outputs, were reported to be capable of significantly increasing the scope of neural network for model identification problems. Three different case studies including a pH reactor dynamics, a batch reactor and an industrial fluidised bed reactor were considered. The simulation results were compared with networks using standard neurones. Zamankhan *et al* [1997] applied back propagation, feed-forward neural networks to estimate mass-transfer parameters in fast fluidised beds of fine solids. They reported that neural networks provide more accurate predictions of mass transfer coefficients compared to traditional heuristic models. Krothapally and Palanky [1997] developed a neural network approach for calculating the optimal operation trajectory based on initial loading conditions and process parameters of batch reactors. The
performance of the neural network strategy was compared with classical and Model Predictive Control approaches for batch polymerisation processes of styrene and methyl methacrylate. The reported results revealed that the MPC algorithm takes two hours to correct for the change in initial conditions while the neural network is able to correct for the same error almost instantaneously. Yang *et al* [1997] used material balance related equations as the regularisation term of a feed-forward neural network to filter out the measurement error. They reported that feed-forward neural networks when coupled with material balances perform well for data reconciliation of a steady state floatation process. Wilson and Zorzetto [1997] used a hybrid model which combines mechanistic elements with artificial neural networks for on line state estimation of the chemical processes. The mechanistic components of the hybrid model were constructed from the balance equations across the process whilst the artificial neural network components were used to estimate the rate relationships.

Nascimento and Guidici [1998] applied another optimisation technique based on neural network approach to an industrial nylon-6, 6-polymerization process. They reported that the new approach provided more comprehensive information for engineering analysis than the conventional non-linear programming procedures and was more reliable for industrial applications. Martinez and Wilson [1998] developed a hybrid neural network for optimisation of a semi-batch auto-catalytic reactor. The proposed hybrid methodology is reported to be able to take advantage of *a priori* information and batch to batch data, to provide a powerful framework for optimisation of complex batch processes. Meghloui *et al* [1998] used a feed-forward neural network to predict the cell
resistance and the cell dynamic trend indicator of an aluminium electrolysis cell, 15 minutes in advance. Elakamel [1998] used a multilayer Perceptron neural network to predict the performance of an enhanced oil recovery processes against an empirical approach employing nine dimensionless groups, a back-propagation algorithm based on Levenberg-Marquardt optimisation technique was used for training the network.

The brief literature survey above shows that there has been an increasing interest for the application of neural networks to chemical engineering applications. However, in the majority of studies to date the main emphasise has been on the application of conventional or standard neural networks as a black box tool to chemical engineering problems-in process modelling, simulation, control, diagnosis, optimisation and physical property estimation.
References

Aldrich, C., Deventer, J.S.J., 1995 “Modelling of induced aeration in turbine aerators by use of radial basis function neural networks” Canadian Journal of Chemical Engineering, Vol. 73, No. 6, pp.808-816


Bulsari, A.B. (Editor), 1995 “Neural Networks for Chemical Engineers”, Elsevier Science

251


Chester, M. 1993 “Neural Network (A Tutorial)”, Prentice-Hall


Dorney, 1975 “A vector space approach to models and optimisation” *Wiley* (interscience), New York

Duong D. Do, 1998 “Adsorption Analysis: Equilibria and Kinetics” *Imperial College Press*


Gill, P.E., Murray, W., and M.H., wright, 1997 “Practical Optimisation” *Academic Press*, San Diego, CA, USA, Eleventh printing

Girosi, F., and N.T., Chan, 1995 “Prior knowledge and the relation of Virtual examples for RBF networks” *IEEE*, pp. 201-210


Karayiannis, N.B., 1997 “Gradient descent learning of radial basis neural networks” IEEE, pp. 1815-1820


Maffezzoni, P., and P., Gubian, 1995a “A new unsupervised algorithm for the placement of centres in a radial basis function neural network (RBFNN)”

Maffezzoni, P., and P., Gubian, 1995b “Approximate radial basis function neural networks (RBFNN) to learn smooth relations from noisy data” *IEEE*, p.553-556


Sohn, I., and N., Ansari, 1997 “A novel algorithm to configure RBF networks” IEEE, pp. 1809-1814
Tao, K.M., 1993 “A closer look at the radial basis function (RBF) networks” IEEE,pp. 401-405
Thorpe, P., 1996 “Making advances in distillation” Process Engineering (London), Vol. 77, No. 9, pp.3-4
Tikhonov, A.N., and V.Y., Arsenin, 1977 “solution of Ill-posed problems” Winston, Washington DC, USA

262
Tikhonov, A.N., and Goncharskey, A.V. (eds.) 1987 “Ill-posed problems in natural sciences”, MIR, Moscow
Tsai, J.R., Chung, P.C., and C.I., Chang, 1996a “Resisting the influence of outliers in radial basis function neural networks” pp. 42-51
Twomey, S., 1977 “ Introduction to the mathematics of inversion in remote sensing and indirect measurements” Elsevier, Amsterdam

263


Wold, S., 1974 “Spline Functions in Data Analysis” Technometrics, Vol. 16, No. 1, pp. 11


264