Nonparametric Regression Estimation with Applications in Radial Basis Networks and Learning

Subha Ramanan

A Thesis
in
The Department
of
Mathematics and Statistics

Presented in Partial Fulfillment of the Requirement for
the Degree of Master of Science in Mathematics at
Concordia University
Montréal, Québec, Canada

September 1994

ISBN 0-315-97653-5

Abstract

Nonparametric Regression Estimation
with Applications in
Radial Basis Function Networks and Learning

Subha Ramanan

Learning algorithms are analysed from the statistical and neural network viewpoints. In the first part, the regression based approach for minimizing the mean squared error is considered. The decomposition of the mean squared error into bias and variance components and their contributions to the error are investigated. Specifically, the k-nearest neighbor (k-NN) regression estimator and the kernel regression estimator (KRE) are studied. The optimal choice of the parameters of these estimators is discussed. In the second part, the neural network approach to the learning problem is explored. Specifically, the Radial basis function (RBF) network is studied in detail. The random sampling and clustering methods of choosing the center parameter of the network are analysed and compared. Comparisons between the RBF nets and the KRE are studied. For both parts, performance of the estimators are assessed by the mean squared error and the results of simulation are presented.

Acknowledgements

I would like to take this opportunity to express my appreciation to my advisor Professor A. Krzyzak for his support, encouragement and advice in the preparation and composition of this thesis. Professor A. Krzyzak has been immensely helpful throughout the course of this work, teaching me much not only about computer science, mathematics and statistics, but also of the joys of research and the pursuit of knowledge. I feel fortunate to have had Professor A. Krzyzak as my thesis advisor.

I am grateful to Professor Y. Chaubey for his suggestions to improve the work. I am also grateful for the financial assistance provided by the Department of Mathematics and Statistics.

Special thanks to friend and mentor S. Vunnava for very useful and helpful discussions. Finally, many thanks to J. Huang, C. Nadal and N. Strathy from the CENPARMI institute for their help and encouragement.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Much of the recent research on artificial neural networks suggests a close analogy to *nonparametric statistical inference.* A branch of statistics concerned with model-free estimation, nonparametric inference has matured in the recent years. With new theoretical and practical developments that have been introduced, there is now a large literature containing themes that parallel neural modeling. A typical problem in nonparametric inference is the estimation (or learning) of arbitrary decision boundaries in a classification problem, based on a collection of pre-classified (or labeled) samples. No assumption is made on the shape of the boundaries and in particular, no parametric model is assumed, in contrast with parametric estimation.

In this thesis, we study learning algorithms from the statistical and neural network viewpoints. We consider the learning problem with a feature or input vector $\mathbf{x}$ and a response vector $\mathbf{y}$. The goal of learning is to predict $\mathbf{y}$ from $\mathbf{x}$, where the pair $(\mathbf{x}, \mathbf{y})$ obeys some unknown joint probability distribution, P. A training set denotes a collection of observed input-response pairs:$\{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_N, \mathbf{y}_N)\}$. These samples are usually independently drawn from P. The response $\mathbf{y}$ may or may not be uniquely determined by the input $\mathbf{x}$, leading to deterministic or ambiguous classifications respectively. The learning problem then, is to construct a decision rule or function $f(\mathbf{x})$ based on the training samples $\{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_N, \mathbf{y}_N)\}$, so that $f(\mathbf{x})$ approximately

generates the response vector $\mathbf{y}$.

The function $f(\mathbf{x})$ is chosen to minimize some functional, the form of which varies with the situation on hand. The optimal $f^*(\mathbf{x})$ depends on the unknown distribution of $(\mathbf{x}, \mathbf{y})$. In practice, we estimate $f^*(\mathbf{x})$ from the learning sequence and obtain an empirical decision rule. One of the standard methods of obtaining this empirical rule is to minimize the sum of squared errors

$$\sum_{i=1}^{N}[\mathbf{y}_i - f(\mathbf{x}_i)]^2 \tag{1.1}$$

where f represents a neural network. Since f is really parameterized, this minimization is over the class generated by all possible values allowed for the parameters.

The thesis can be broadly divided into two parts. In the first part, we focus our attention on the regression based approach for minimizing the mean squared error (1.1), by reasoning that amongst all functions of $\mathbf{x}$, the regression is the best predictor of $\mathbf{y}$ given $\mathbf{x}$. Formulating the learning problem as a non-linear regression problem, the error in estimation is decomposed into two factors : bias and variance. While an incorrect model leads to high bias, truly model-free estimation leads to high variance. Model free approaches are typically slow to converge – a large number of training samples being required to achieve a reasonable performance. This is the effect of high variance caused by the the large number of parameters, in fact, infinite number in true model free inference that are to be estimated. Hence massive training sets are required to significantly reduce the variance contribution to the estimation error. Indeed, the variance can be controlled by using model-based estimation. However, for complex inference problems, correct models are hard to identify and hence model-based inference suffers from high bias. This suggests that, when faced with a complex inference task, there is a trade-off involved between bias and variance. We illustrate this trade-off involved in the inference problem.

Non-parametric estimators that deal with the regression problem have been extensively studied in the modern statistics literature [23]. Different techniques have been proposed to construct these estimators. These include the k-Nearest neighbor(k-NN) estimators [2, 3], Kernel regression estimators(KRE) [7], Multiple adaptive regression

splines(MARS) [8], Boltzmann machines [12] and feed-forward neural networks [9] . We approach the regression problem with the k-NN estimator and the KRE, these being well known for their good performance.

In experimenting with the above estimators of the regression function, we consider a suitable classification curve and obtain expressions for the regression. We describe the generic features of the bias and variance of the mean squared error observed from the above non-parametric estimators. We then study the behaviour of the mean squared error as a function of the parameters involved in each method of estimation. We consider the consistency property of the estimators and study the large sample convergence of the error. We perform the experiments with two kinds of data:

- Deterministic data: Inputs are chosen in a deterministic manner.
- Ambiguous data: Inputs are perturbed by a random mechanism.

Similar studies were made by Geman [9], on the k-NN estimators and feedforward neural networks. We extend the experiments performed with the k-NN estimators to the KRE and study various choices of kernels suitable for the regression problem. We identify the optimal kernel suited to the nonparametric regression estimation problem considered.

The second part of the thesis explores the learning problem from a neural network perspective. After several years of extensive studies on the multilayer perceptron – initially the most popular model of feedforward neural networks, researchers have turned their attention to a number of other models including Multivariate Adaptive regression splines (MARS) nets [8], Wavelet function nets, sigmoidal nets [16] and Radial Basis Function (RBF) nets [28, 26].

The sigmoidal nets have been extensively studied and its properties well employed in the regression estimation [1] and classification [9] problems. Approximation results for these nets have been obtained by Barron [1]. On the other hand, although generalization and approximation abilities of the RBF net have been explored [19, 20, 15],

these estimation properties have not been exploited yet in classification and regression. We therefore approach the learning problem through RBF nets. Further, we experimentally compare the performance of the RBF network and the KRE [28].

In exploring the RBF approach, we describe the several parameters that are involved in the network. We describe the *random sampling* and *clustering* methods of selecting the center parameters of the network, the former method assigning a randomly selected subset of the training set to the center parameters and the latter selecting the cluster centers of the data for this purpose. Further, we consider the minimization of the mean squared error 1.1 with respect to the weight vectors of the net and use the solution of this linear optimization problem as the weight vectors of the network.

The above presents a difficult problem to analyse from the theoretical standpoint: we therefore study it experimentally by performing simulation modelling with two kinds of data:

- Unclustered data : We use uniformly distributed deviates.
- Clustered data : We use a mixture of different gaussian distributed deviates.

The purpose of such a choice of data was to study the RBF net when the center vectors are chosen by the random sampling and clustering methods and to study the effects of clustering. The following questions were explored in this connection:

- How does the RBF net perform when the center vectors are chosen by selecting a random subset of the training set?
- How does the behaviour of the RBF net alter when the center vectors are chosen by a clustering algorithm?
- How does the KRE behave and compare with each of the above cases?

Our plan of presenting the above is the following: Chapter 2 gives a description of the regression problem and the bias-variance trade-off, describing the k-NN and KRE

methods of estimating the regression. Chapter 3 describes the experiments performed with the k-NN and Kernel regression estimators. In Chapter 4, a detailed discussion is given on the RBF network approach to the learning problem and its connections with the KRE. Chapter 5 details the experiments performed with the RBF network and compares the results with those performed with the KRE. We conclude with an analysis of the results from the above experiments.

Chapter 2

Nonparametric Approaches to the Regression problem

In this chapter, we examine the regression problem through parametric and non-parametric approaches. The advantages and disadvantages of either approach are discussed. We consider the bias/variance decomposition of the mean-squared error and illustrate the issues that arise in balancing these components. The trade-off involved in aiming to reduce contributions of each of these components is illustrated through an example of the regression problem. We conclude the chapter with a description of the k-NN and KRE estimators and a discussion on the optimal choice of the parameters involved in each method.

2.1 Regression and Least-Squares Estimation

In Least-Squares Estimation, the goal of learning is to construct a function f based on the data $\{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_N, \mathbf{y}_N)\}$ such that f minimizes the sum of observed squared

errors :

$$\sum_{i=1}^{N}[\mathbf{y}_i - f(\mathbf{x}_i)]^2$$

This method defines one way of estimating the regression.

Definition 2.1 *The regression of* $\mathbf{y}$ *on* $\mathbf{x}$ *is* $E[\mathbf{y} \mid \mathbf{x}]$*, that is, that deterministic function of* $\mathbf{x}$ *that gives the mean value of* $\mathbf{y}$ *conditioned on* $\mathbf{x}$.

In aiming to "fit the data",we look for that function of $\mathbf{x}$ that gives the best predictor of $\mathbf{y}$[1] given $\mathbf{x}$. The regression is an excellent solution, in the mean-squared error sense, as can be seen from the following reasoning.

Denoting $\mathrm{E}[\mathbf{y} \mid \mathbf{x}] = m(\mathbf{x})$, consider the following mean squared error, for any function $f(\mathbf{x})$, and any fixed $\mathbf{x}$:

$$\begin{aligned}
E[(y - f(\mathbf{x}))^2|\mathbf{x}] &= E[((y - m(\mathbf{x})) + (m(\mathbf{x}) - f(\mathbf{x}))^2|\mathbf{x}] && (2.1)\\
&= E[(y - m(\mathbf{x}))^2|\mathbf{x}] + (m(\mathbf{x}) - f(\mathbf{x}))^2 + \\
&\quad 2E[(y - m(\mathbf{x}))|\mathbf{x}](m(\mathbf{x}) - f(\mathbf{x})) \\
&= E[(y - m(\mathbf{x}))^2|\mathbf{x}] + (m(\mathbf{x}) - f(\mathbf{x}))^2 + \\
&\quad 2(m(\mathbf{x}) - m(\mathbf{x}))(m(\mathbf{x}) - f(\mathbf{x})) \\
&= E[(y - m(\mathbf{x}))^2|\mathbf{x}] + (m(\mathbf{x}) - f(\mathbf{x}))^2 && (2.2)\\
&\geq E[(y - m(\mathbf{x}))^2|\mathbf{x}] \\
& && (2.3)
\end{aligned}$$

Hence among all functions of $\mathbf{x}$, the regression is the best predictor of y given $\mathbf{x}$, in the mean-squared error sense.

Alternative approaches to the least-squares estimator include likelihood based approaches. As opposed to decreasing the squared error, these algorithms aim at increasing the likelihood. The maximum likelihood estimator is certainly much studied

[1] For convenience, we take $\mathbf{y}$ to be one dimensional, that is, $\mathbf{y} = y$, though the discussion applies to the multidimensional case as well.

in statistics, mainly due to its optimality properties. One of the most extensively studied neural network in recent years is the back-propagation network which uses the least squares estimator in the error back-propagation algorithm [12]. This leads us to focus our attention on the least squares estimators.

2.2 Parametric vs Nonparametric Estimation

Given a set of training or design samples representative of the type of features and underlying class principles, with each labelled by its correct class, we can proceed to estimate the decision boundaries through parametric or non-parametric approaches.

Illustration

Consider the classification example with two classes : class A and its complement. Let y be 1 when $\mathbf{x}$ falls in class A, and 0 otherwise.

The regression is then

$$\begin{aligned} \mathrm{E}[y \mid \mathbf{x}] &= P[y = 1 \mid \mathbf{x}] \\ &= P[y \in \text{class A} \mid \mathbf{x}] \end{aligned}$$

which is the probability of falling in class A as a function of the input vector $\mathbf{x}$. Suppose the decision rule is to choose class A if $P[y \in \text{class A} \mid \mathbf{x}] > 1/2$. This partitions the range (H) of $\mathbf{x}$ into two regions : $H_A = \mathbf{x} : P[y \in \text{class A} \mid \mathbf{x}] > 1/2$ and its complement $H - H_A = H_{\bar{A}}$. The separation between H_A and $H_{\bar{A}}$ can be a regular surface or a highly irregular one – this decides the efficiency of the approach chosen.

Parametric Approach

This approach assumes a priori knowledge of H_A up to a finite number of parameters. This would indeed be the case if H_A and $H_{\bar{A}}$ were separated by a linear or quadratic surface. If this is the case, then fewer samples are needed for accurate estimation than if we were to estimate without parametric specifications. Hence, this approach has the advantage of efficiency. But suppose the true discrimination function substantially deviates from the assumed decision boundary, then the parametric approach would converge to an incorrect and hence suboptimal solution.

Nonparametric Approach

This approach makes no a priori commitments on the decision boundaries. Proceeding without parametric specifications, this approach is destined to converge to the correct solution, if the number of training samples increases without bound. We define the important property of consistency which most nonparametric regression estimators share. There are several versions of consistency depending on the type of convergence taken into account. Here we give the definition in terms of convergence in the mean squared error sense.

Definition 2.2 *An estimator $f_n(\mathbf{x})$ of y is said to be consistent if*

$$E(f_n(\mathbf{x}) - y)^2 \rightarrow 0 \ \ as \ n \ \rightarrow \infty$$

Consistency therefore deals with the asymptotic (large sample) convergence of an estimator to the object of estimation.

This property, indeed, furnishes a great advantage to nonparametric algorithms, but it comes with a high price : nonparametric estimators may be extremely slow to converge. An unreasonably large number of training samples may be needed to make even crude approximations of the target regression function.

Parametric algorithms face the risk of being bias-prone, when the assumed form of the

decision boundaries depart considerably from the true separation. On the other hand, non-parametric algorithms may be too dependent on the particular observations when small samples are used, and hence tend to have a high variance factor. A potential problem in using parametric approaches in practice is that we require the underlying class-conditional distributions. Unfortunately, these problems may arise:

1. We are not able to determine a specific form (eg., Gaussian) of the distributions.
2. The form chosen does not fit one of the 'estimable' formulations.

For these reasons, we resort to non-parametric estimation techniques.

In the following section, we discuss the roles played by bias and variance factors by considering the bias/variance decomposition of the mean-squared error.

2.3 Bias and Variance

Consider a training data $\mathcal{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$. The regression problem is to construct a function $f(\mathbf{x})$ based on $\mathcal{D}$ for the purpose of approximating y at future observations of $\mathbf{x}$. To denote explicitly the dependence of f on $\mathcal{D}$, we write $f(\mathbf{x}; \mathcal{D})$ instead of $f(\mathbf{x})$. Given $\mathcal{D}$ and a particular $\mathbf{x}$ independent of $\mathcal{D}$, the mean-squared error

$$E[(y - f(\mathbf{x}; \mathcal{D}))^2 | \mathbf{x}, \mathcal{D}]$$

is a natural measure of the effectiveness of f as a predictor of y.

Consider equation (2.2) in terms of the new notation:

$$E[(y - f(\mathbf{x}; \mathcal{D}))^2 | \mathbf{x}, \mathcal{D}] = E[(y - \mathrm{E}[y \mid \mathbf{x}])^2 | \mathbf{x}, \mathcal{D}] + (f(\mathbf{x}; \mathcal{D}) - \mathrm{E}[y \mid \mathbf{x}])^2$$

Note that the quantity

$$E[(y - \mathrm{E}[y \mid \mathbf{x}])^2 | \mathbf{x}, \mathcal{D}]$$

is the variance of y given $\mathbf{x}$ and therefore does not depend on the data $\mathcal{D}$ or on the estimator f. Hence, the effectiveness of f as a predictor of y is really measured by the squared distance to the regression function,

$$(f(\mathbf{x};\, \mathcal{D}) - \mathrm{E}[\mathrm{y} \mid \mathbf{x}])^2$$

We now measure the effectiveness of f as an estimator of the regression. The mean-squared error of f as an estimator of the regression $\mathrm{E}[\mathrm{y} \mid \mathbf{x}]$ is then,

$$E_D[(f(\mathbf{x};\, \mathcal{D}) - \mathrm{E}[\mathrm{y} \mid \mathbf{x}])^2] \tag{2.4}$$

where E_D stands for expectation with respect to the training set $\mathcal{D}$. For a fixed sample size, this quantity is nothing but the average over the collection of possible sets $\mathcal{D}$.

Analysis of the mean-squared error

For a particular training set $\mathcal{D}$, the estimator $f(\mathbf{x};\, \mathcal{D})$ may happen to be an excellent approximation to $\mathrm{E}[\mathrm{y} \mid \mathbf{x}]$. It may be the case that $f(\mathbf{x};\, \mathcal{D})$ differs widely for other realizations of $\mathcal{D}$, that is, there can be a substantial variation of $f(\mathbf{x};\, \mathcal{D})$ with $\mathcal{D}$. On the other hand, it may happen that the average of $f(\mathbf{x};\, \mathcal{D})$ (over all possible $\mathcal{D}$) is far from the regression $\mathrm{E}[\mathrm{y} \mid \mathbf{x}]$. In both these situations, there will be large contributions to the mean-squared error (2.4) making $f(\mathbf{x};\, \mathcal{D})$ an unreliable predictor of y.

2.3.1 Bias/Variance Decomposition

The sources of estimation error mentioned above, can be efficiently assessed through the bias/variance decomposition. We derive this in a way similar to equation (1.1).

For any $\mathbf{x}$, letting $E_D[f(\mathbf{x};\, \mathcal{D})] = r(\mathbf{x})$, consider the mean-squared error

$$
\begin{aligned}
E_D[(f(\mathbf{x};\mathcal{D}) - \mathrm{E}[y \mid \mathbf{x}])^2] &= E_D[((f(\mathbf{x};\mathcal{D}) - r(\mathbf{x})) + \\
&\quad (r(\mathbf{x}) - \mathrm{E}[y \mid \mathbf{x}]))^2] \\
&= E_D[f(\mathbf{x};\mathcal{D}) - r(\mathbf{x})]^2 + \\
&\quad E_D[r(\mathbf{x}) - \mathrm{E}[y \mid \mathbf{x}]]^2 + \\
&\quad 2E_D[(f(\mathbf{x};\mathcal{D}) - r(\mathbf{x}))(r(\mathbf{x}) - \mathrm{E}[y \mid \mathbf{x}])] \\
&= (r(\mathbf{x}) - \mathrm{E}[y \mid \mathbf{x}])^2 + \\
&\quad E_D[(f(\mathbf{x};\mathcal{D}) - r(\mathbf{x}))^2] \qquad (2.5)
\end{aligned}
$$

The first term on the left hand side of equation (2.5) measures the distance of the mean value of the estimator to the regression and denotes the *bias* of the estimator $f(\mathbf{x};\mathcal{D})$ while the second is the *variance* of the estimator.

An unbiased estimator can still have a large mean-squared error, if the variance is large, that is, even when $E_D[f(\mathbf{x};\mathcal{D})] = \mathrm{E}[y \mid \mathbf{x}]$, the estimator may be highly sensitive to the data, leading to a large variance and hence a large mean squared error. On the other hand, if on the average, $f(\mathbf{x};\mathcal{D})$ is different from $\mathrm{E}[y \mid \mathbf{x}]$, then $f(\mathbf{x};\mathcal{D})$ is said to be a biased estimator of $\mathrm{E}[y \mid \mathbf{x}]$. This in general, is dependent on P (See Chapter 1). Hence, a biased estimator would again lead to a large mean-squared error.

A good balance between the bias and variance factors is therefore needed to efficiently reduce the mean-squared error. The issue of balancing bias and variance has been well-studied in estimation theory. We illustrate this trade-off with the following one-dimensional regression problem.

2.3.2 Illustration

Here we take $\mathbf{x} = x$ and y to be related to x by the relation

$$y = g(x) + \eta$$

where g is some unknown function and η is zero-mean noise following a distribution independent of x. The regression E[y | x] would then be simply $g(x)$ and this, by the previous reasoning would be the best predictor of y in the mean-squared error sense.

To illustrate clearly the bias and variance contributions to the mean-squared error, let us suppose that we keep x deterministic and let y alone be random. Suppose we collect N observations $\{x_1, \ldots, x_N\}$. The data then consists of the corresponding N values of y, $\mathcal{D} = \{y_1, \ldots, y_N\}$. The goal is to make a guess at $g(x)$ using the noisy observations $y_i = g(x_i) + \eta_i$.

On the one hand, suppose we define $f(\mathbf{x}; \mathcal{D})$ to be entirely dependent on the training data, i.e, $f(\mathbf{x}; \mathcal{D})$ is an interpolant of the data. This would make the estimator truly unbiased at $x = x_i, 1 \leq i \leq N$, since

$$E_D[f(\mathbf{x}; \mathcal{D})] = E[g(x_i) + \eta_i] = g(x_i) = E[y|x_i]$$

Further, if g is continuous, there would be very little bias contributed from the neighborhood of the observation points $x_i, 1 \leq i \leq N$. Hence, the overall contribution to the mean-squared error from the bias factor would be small. But if the variance of the noise η is large, then this would lead to a large variance component in the mean-squared error (2.5) , since

$$E_D[(f(\mathbf{x}; \mathcal{D}) \ - E_D[f(\mathbf{x}; \mathcal{D})])^2] = E_D[(g(x_i) + \eta_i - g(x_i))^2] = E[\eta_i^2]$$

which is the variance of η_i, as the noise has zero mean. This estimator is therefore highly sensitive to the data.

On the other hand, we can suppose $f(\mathbf{x}; \mathcal{D})$ to be independent of $\mathcal{D}$, that is, we may define $f(\mathbf{x}; \mathcal{D}) = h(x)$, for some carefully chosen function $h(x)$. Since the estimator does not depend on $\mathcal{D}$, the issue of variance is solved. But this would indeed be likely to introduce a substantial bias factor, as this estimator completely ignores the data.

A wise choice therefore, would be an intermediate choice between the two extremes. For instance, we may aim to combine smoothness while retaining some consideration

for the observed data. We discuss various approaches to the regression problem which deal with this trade-off.

2.4 k-NN Approach

This simple approach has been extensively used because of its efficient performance. Given a training set $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_N, \mathbf{y}_N)\}$, the 'memory' of the machine comprises exactly of $\mathcal{D}$. For any input vector $\mathbf{x}$, a response vector is obtained from the training set by averaging the responses to those inputs that happen to lie close to $\mathbf{x}$.

A collection of algorithms indexed by an integer k is used to determine the number of neighbors of x that are considered in the above average. Let $N_k(\mathbf{x})$ denote the collection of indices of the k nearest neighbors to $\mathbf{x}$ among the input vectors in the training set D. Then the k-nearest neighbor estimator is given by

$$f(\mathbf{x}; \mathcal{D}) = \frac{1}{k} \sum_{i \in N_k(\mathbf{x})} \mathbf{y}_i$$

Optimal choice of k

If k is large, then the response $f(\mathbf{x}; \mathcal{D})$ is a relatively smooth function of $\mathbf{x}$, but this function does not give due consideration to the actual position of the $\mathbf{x}_i$'s in the training set. In fact, when $k =$ N, $f(\mathbf{x}; \mathcal{D})$ does not depend on $\mathbf{x}$ or on $\mathbf{x}_i$, $1 \leq i \leq N$; the output in this case is just the average observed output $1/N \sum_{i=1}^{N} \mathbf{y}_i$. When N itself is large, $1/N \sum_{i=1}^{N} \mathbf{y}_i$ is likely to remain unchanged from one training set to another. Thus, the contribution of the variance to the mean squared error is small. On the other hand, the bias contribution would be large since the response to a particular $\mathbf{x}$ is systematically biased toward the population response irrespective of any local variation in the neighborhood of $\mathbf{x}$.

The other extreme is $k = 1$, that is, the first nearest neighbor estimator. With this choice, we can expect the bias to be appreciably small, as the bias of the first

nearest neighbor goes to zero as N goes to infinity. On the other hand, the variance contribution to the mean squared error is typically large. This is because the response at each $\mathbf{x}$ is rather sensitive to the peculiarities of the particular training samples in $\mathcal{D}$.

It follows from this reasoning, that the optimal mean squared error is obtained from a compromise between the two extremes $k = 1$ and $k = \mathrm{N}$. By choosing an intermediate k, thereby making $f(\mathbf{x}; \mathcal{D})$ reasonably smooth, we aim to achieve a significant reduction of the variance without introducing too much bias.

Consistency properties

If we let $N \rightarrow \infty$, the k- nearest neighbor estimator can be made consistent by letting $k = k_N \uparrow \infty$ sufficiently slowly. This is because the variance is controlled by letting $k_N \uparrow \infty$, while the bias is controlled by ensuring that the $k_N th$ nearest neighbor of $\mathbf{x}$ gets closer to $\mathbf{x}$ as $N \rightarrow \infty$.

It has been shown [5] that under various noise conditions, the nearest neighbor estimate is strongly uniformly consistent. In the case when there is a large amount of data, the computational burdens of processing the data could be large. Devroye and Wise [6] have proposed a recursive method of estimation to deal with this, giving distribution-free consistency results for the recursive nearest neighbor regression estimator. Further, the rates of convergence of the k-NN regression estimate have been obtained by Györfi [10] and Stone [24], while the rates of convergence of bias and variance have been analysed by Mack [17].

We study the large-sample convergence of the k-NN estimator in our experiments conducted with the nearest neighbor estimator(See Chapter 3).

2.5 KRE Approach

In this method, again, the 'memory' of the machine is composed of the entire training set $\mathcal{D}$, but here, the estimation is done by combining kernels (or Parzen windows) which are placed around each observed input $\mathbf{x}_i, 1 \leq i \leq N$.

The kernel is usually chosen to be a non-negative function of $\mathbf{x}$ which is maximum at $\mathbf{x} = 0$ and decreasing away from $\mathbf{x} = 0$. A common choice is the gaussian kernel :

$$W(\mathbf{x}) = \left(\frac{1}{\sqrt{2\pi}}\right)^d \exp\left\{-\frac{1}{2}|\mathbf{x}|^2\right\}, \text{ for } \mathbf{x} \in R^d$$

where $|\mathbf{x}|$ stands for the vector norm of $\mathbf{x}$.

A number of alternatives to the gaussian kernel can be also be considered. In addition to the gaussian kernel, we have experimented with the exponential kernel given by:

$$W(\mathbf{x}) = \exp\{-|\mathbf{x}|\}$$

and the rectangular kernel given by:

$$W(\mathbf{x}) = \begin{cases} 1/2 & \text{for } |\mathbf{x}| \leq 1 \\ 0 & \text{otherwise} \end{cases}$$

where $|\mathbf{x}|$ denotes the distance to the zero vector.

Further, we have also studied the mean squared error of the KRE with the Fejer-de la Vallee-Poussin kernel, commonly called the *sinc* kernel, given by

$$W(\mathbf{x}) = \begin{cases} \frac{1}{2}\left(\frac{\sin(|\mathbf{x}|/2)}{|\mathbf{x}|/2}\right)^2 & \text{if } |\mathbf{x}| \neq 0 \\ 1/2\pi & \text{if } |\mathbf{x}| = 0 \end{cases}$$

The scale of the kernel is adjusted by a bandwidth σ which governs the extent to which the window is concentrated at $\mathbf{x} = 0$, or is spread out around $\mathbf{x} = 0$.

Having fixed a kernel W and a bandwidth σ, the kernel regression estimator at $\mathbf{x}$ is formed from a weighted average of the observed responses $y_i, 1 \leq i \leq N$:

$$f(\mathbf{x}; \mathcal{D}) = \frac{\sum_{i=1}^{N} y_i W[(\mathbf{x} - \mathbf{x}_i)/\sigma]}{\sum_{i=1}^{N} W[(\mathbf{x} - \mathbf{x}_i)/\sigma]} \tag{2.6}$$

Since $W[(\mathbf{x} - \mathbf{x}_i)/\sigma]$ has its maximum at $\mathbf{x} = \mathbf{x}_i$, it follows that observations with inputs closer to $\mathbf{x}$ are weighted more heavily.

Optimal choice of σ

When the bandwidth σ is small, only points that are close to $\mathbf{x}$ contribute to the response at this point. This procedure is similar to the k-nearest neighbor method with small k. On the other hand, when σ is large, many neighbors contribute significantly to the response presenting a situation analogous to the choice of large k in the k- nearest neighbor method.

Hence the bandwidth σ governs the bias and variance in a manner similar to the parameter k in the nearest neighbor procedure : Small bandwidths produce high variance and low bias while large bandwidths incur relatively high bias but low variance.

The problem of selecting the optimal bandwidth in nonparametric regression estimation has been studied by Härdle and Marron [11].

We further discuss the selection of optimal bandwidth for the problem on hand, in our description of experiments with the KRE (See Chapter 3).

Consistency properties

The KRE has been shown to be consistent, and distribution-free consistency properties of the KRE in regression estimation have been studied [7]. Further, global convergence of the recursive kernel regression estimates have been established and their rates of convergence analysed [14, 24].

Chapter 3

Experiments with the k-NN and KRE estimators

In this chapter, we describe the experiments performed with the k-NNR and KRE estimators. We display the features of the two error components: bias and variance, and the behaviour of the mean squared error as the parameters involved are varied. We also study the asymptotic(large-sample) convergence of these estimators. The experiments were performed with two kinds of data: *deterministic* and *ambiguous*. With this facility, we study the regression problem under two broad categories consisting of deterministic and ambiguous classifications.

Similar studies for the k-NN estimator were performed by Geman [9]. In addition to the nearest neighbor estimator, we perform experiments with the KRE and study the choice of kernel.

3.1 The Classification Problem

We consider a binary classification problem – where the output can be categorized in one of two classes. We represent these classes by the values ± 0.9. The input comprises of two components, $\mathbf{x} = (x_1, x_2)$, and is drawn from the rectangle $\mathcal{R} =$ [-6, 6] $\times$ [-1.5, 1.5].

3.1.1 Deterministic Data

In the deterministic case, the classification is determined by the curve

$$x_2 = \sin((\pi/2)x_1)$$

which divides the rectangle $\mathcal{R}$ into two pieces: class A $= [x_2 \geq \sin((\pi/2)x_1), y = 0.9]$ and class B $= [x_2 < \sin((\pi/2)x_1), y = -0.9]$. The regression is then the binary-valued function:

$$\mathrm{E}[\mathrm{y} \mid \mathbf{x}] = \begin{cases} 0.9 & \text{if } y \in \text{class A} \\ -0.9 & \text{if } y \in \text{class B} \end{cases}$$

In order that each class is well represented, the training set is comprised of 200 examples and is constructed to have 100 examples from each class. The 100 inputs corresponding to $y = 0.9$ are chosen from the uniform distribution above the sinusoid and the other 100 inputs are chosen from the uniform distribution below the sinusoid.

3.1.2 Ambiguous Data

Within the same basic setup described in the previous subsection , the classification can be made ambiguous by randomly perturbing the input vector before determining its class. To describe this random mechanism, let us denote by $S_1(\mathbf{x})$, the disk with unit radius centered at $\mathbf{x}$. Given an input $\mathbf{x}$, the output y is randomly classified as follows: we first perturb the input vector $\mathbf{x}$ by choosing a point $\mathbf{z} = (z_1, z_2)$ from the

uniform distribution on $S_1(\mathbf{x})$. This is equivalent to choosing z_1 from the uniform distribution on $[x_1 - 1, x_1 + 1]$ and z_2 from the uniform distribution on $[x_2 - 1, x_2 + 1]$ and restricting (z_1, z_2) to lie inside the unit disk $S_1(\mathbf{x})$ [22]. y is then assigned the value 0.9 if $z_2 \geq \sin((\pi/2)z_1)$ and -0.9 otherwise. For a given $\mathbf{x}$, and hence a randomly chosen $\mathbf{z}$, the resulting regression would then be:

$$\mathrm{E}[\mathrm{y} \mid \mathbf{x}] = 0.9P[z_2 \geq \sin((\pi/2)z_1] - 0.9P[z_2 < \sin((\pi/2)z_1]$$

Denoting A_1 to be the area above the sine curve bounded by the unit disk and A_2 to be the area below the sine curve bounded by the unit disk, $P[z_2 \geq \sin((\pi/2)z_1] = A_1/\pi$ and $P[z_2 < \sin((\pi/2)z_1] = A_2/\pi$, since $z = (z_1, z_2)$ is an uniform deviate on the unit disk. The above regression therefore becomes

$$\mathrm{E}[\mathrm{y} \mid \mathbf{x}] = \frac{0.9}{\pi}(A_1 - A_2) \tag{3.1}$$

The areas A_1 and A_2 depend on where the intersecting points of the unit disk and the sine curve lie – depending on their position with respect to the center of the disk, the calculation of the regression can be categorized into several cases. These cases and the ensuing expressions for the regression are derived in the appendix.

As proved in Chapter 2, in the problem of minimizing the mean squared error, the best response to a given $\mathbf{x}$ is the regression $\mathrm{E}[\mathrm{y} \mid \mathbf{x}]$.

As with the deterministic case, the training set for the ambiguous classification task was also built in such a way that it comprised of 100 examples from each class. This was done by repeatedly selecting pairs $(\mathbf{x}, y)$, where $\mathbf{x}$ was chosen uniformly from the rectangle $\mathcal{R}$ and y was chosen by the random procedure described above. The first 100 examples for which $y = 0.9$ and the first 100 examples for which $y = -0.9$ formed the training set.

3.2 Evaluation of Bias, Variance and MSE

In each of the experiments conducted with the nearest neighbor and the kernel regression estimators, the bias, variance and MSE were evaluated by the following procedure:

Let $f(\mathbf{x};\ \mathcal{D})$denote the regression estimator for any given training set $\mathcal{D}$. Then, from equation (2.5), the squared bias at $\mathbf{x}$ is given by

$$(E_D[f(\mathbf{x};\ \mathcal{D})] - \mathrm{E}[\mathrm{y} \mid \mathbf{x}])^2$$

and the variance is given by

$$E_D[(f(\mathbf{x};\ \mathcal{D}) - E_D[f(\mathbf{x};\ \mathcal{D})])^2]$$

These error components are assessed by independently choosing 100 training sets $\mathcal{D}^1, \mathcal{D}^2, \ldots, \mathcal{D}^{100}$, and by finding the corresponding regression estimators $f(\mathbf{x}, \mathcal{D}^1), \ldots, f(\mathbf{x}, \mathcal{D}^{100})$. The average response at $\mathbf{x}$ is then given by:

$$\bar{f}(\mathbf{x}) = 1/100 \sum_{k=1}^{100} f(\mathbf{x}; \mathcal{D}^k)$$

Squared bias and variance are then estimated by the formulas:

$$\mathrm{Bias}^2(\mathbf{x}) \approx (\bar{f}(\mathbf{x}) - \mathrm{E}[\mathrm{y} \mid \mathbf{x}])^2$$

$$\mathrm{Variance}(\mathbf{x}) \approx \frac{1}{100} \sum_{k=1}^{100} [f(\mathbf{x}; \mathcal{D}^k) - \bar{f}(\mathbf{x})]^2$$

The estimated mean squared error, MSE(x) is the sum, $\mathrm{Bias}^2(\mathbf{x})$ + Variance(x) and is given by :

$$\mathrm{MSE}(\mathbf{x}) \approx \frac{1}{100} \sum_{k=1}^{100} (f(\mathbf{x}; \mathcal{D}^k) - \mathrm{E}[\mathrm{y} \mid \mathbf{x}])^2$$

3.3 Analysis of bias, variance and MSE curves

We study the performance of the different estimators by assessing the behaviour of the error components and the mean squared error for different values of the parameters involved.

3.3.1 k-NN Estimation

In both the deterministic and ambiguous case, the bias increased and the variance decreased when the number of neighbors is increased, as expected from the discussion in Chapter 2. The least mean squared error, in the deterministic case, is achieved using a small number of neighbors, two or three. In contrast to this, in the ambiguous case, the least error is obtained by using the more biased 15 or 16 nearest neighbor estimators. Figures 3.1 and 3.2 display the results of the experiments with the nearest-neighbor procedure.

Large sample convergence

When the training size N in the above experiments is gradually increased, the mean squared error can be studied as a function of the training size. Indeed, as explained in Chapter 2, the k nearest neighbor estimator can be made consistent by choosing $k = k_N \rightarrow \infty$ sufficiently slowly, with $k_N/N \rightarrow 0$. This means that, by choosing an appropriate number of neighbors satisfying the above condition, the mean squared error of the k nearest neighbor estimator would go to zero as the training size N goes to infinity. We study this property of consistency by choosing $k = \sqrt{N}$ and analysing the behavior of the mean squared error as N is increased. The convergence of the mean squared error for the deterministic case is displayed in figure 3.3.

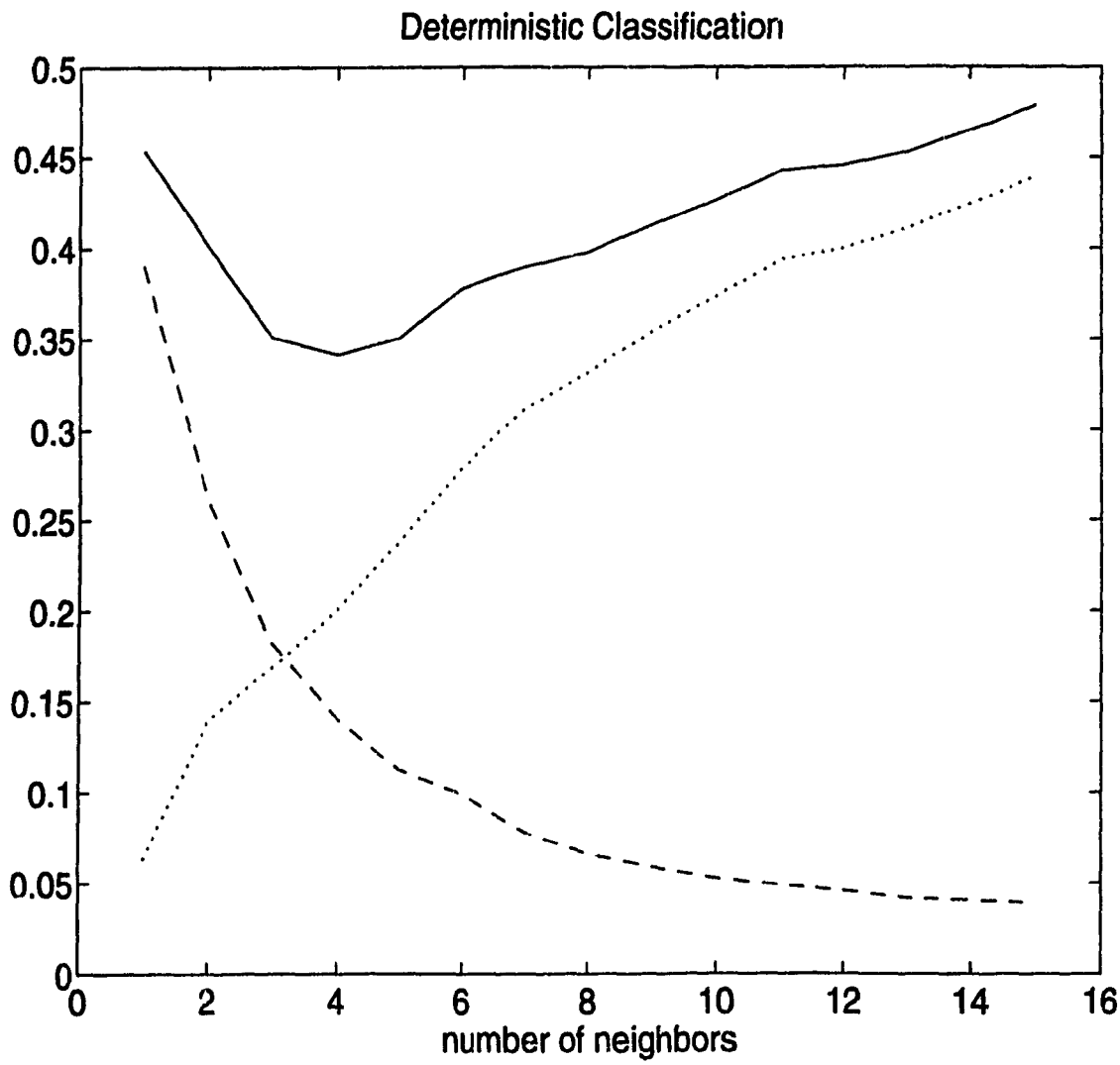


Figure 3.1: Bias(dots), variance(dashes) and mean squared error(solid) as functions of the number of neighbors in the k-NN estimation, when the classification is *deterministic.*

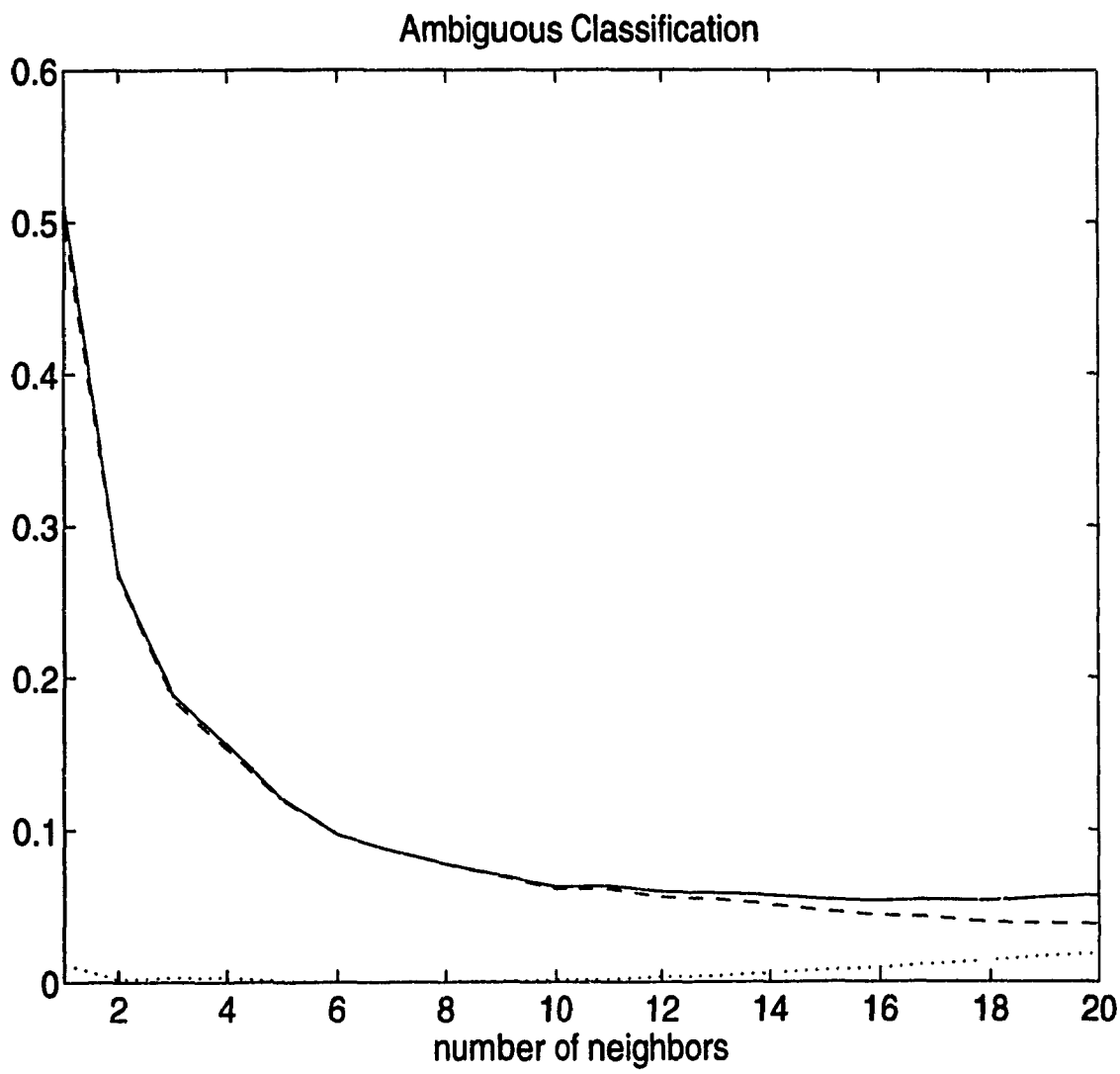


Figure 3.2: Bias(dots), variance(dashes) and mean squared error(solid) as functions of the number of neighbors in the k-NN estimation when the classification is *ambiguous*.

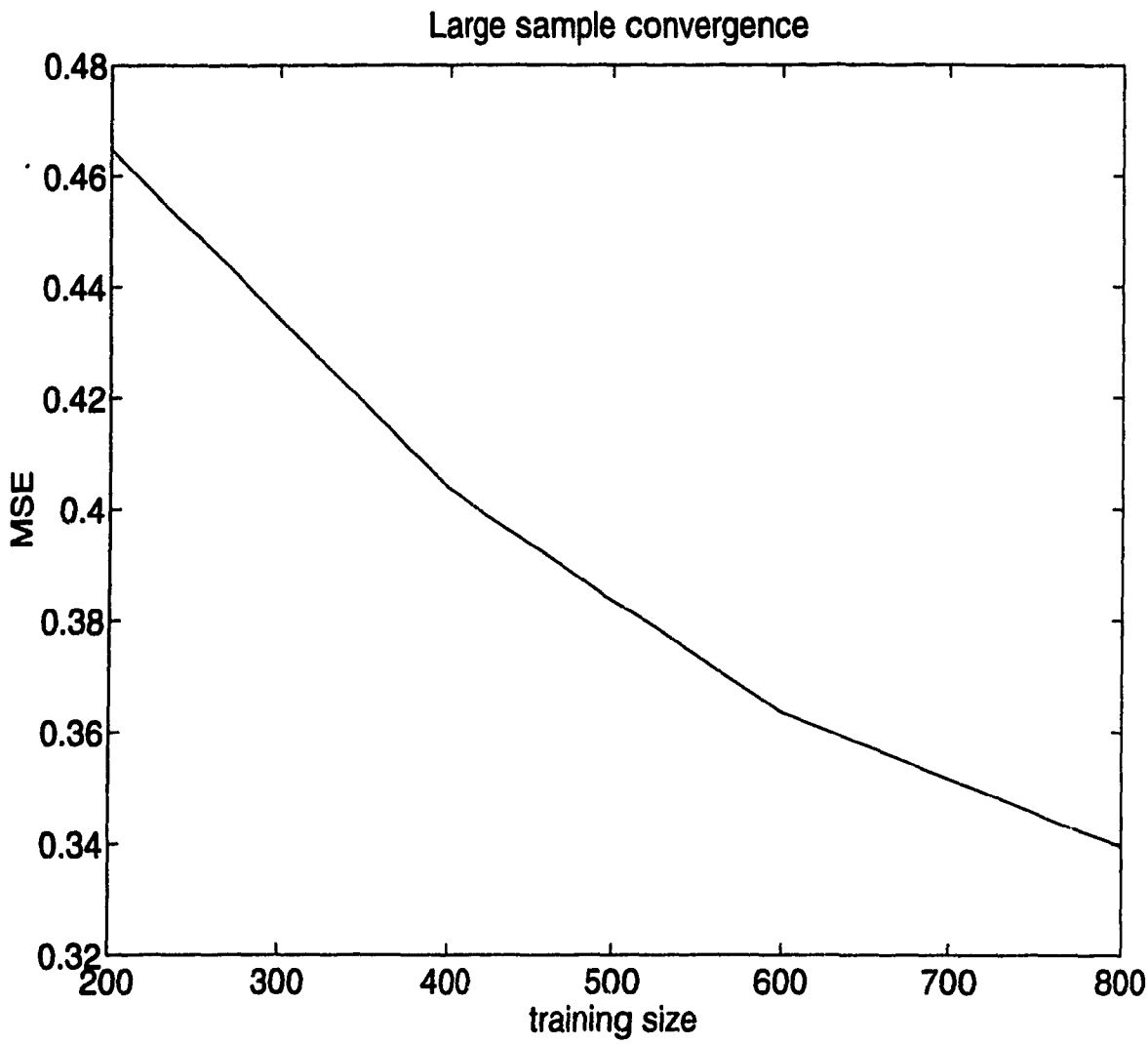


Figure 3.3: Behaviour of the mean squared error as the training size is increased in the k-NN estimation for the deterministic classification problem.

3.3.2 KRE Estimation

The bandwidth σ in kernel regression estimation is comparable to the number of neighbors in the k-NN estimation as both these parameters govern the bias and variance factors in an analogous manner (see section 3.3). A study of the bias, variance and MSE curves with the gaussian kernel displayed this analogous behavior : in both deterministic and ambiguous cases, the bias increased and the variance decreased as the bandwidth is increased. In the deterministic case, the optimal mean squared error is attained at small bandwidths, $\sigma = 0.1$ or $\sigma = 0.2$ while in the ambiguous case it is attained at larger bandwidths of 0.5 or 0.6.

Figures 3.4 and 3.5 illustrate the above results.

In the deterministic case, in addition to the gaussian kernel, we experimented with the exponential and rectangular kernels (Chapter 2) as well, to observe how the choice affects the curves and to select the kernel that best fits the problem on hand. All three kernels resulted in similar trends in the bias-variance curves, as displayed in figures 3.6 and 3.7.

We also experimented with the *sinc* kernel (Chapter 2) which has been proven by Watson and Leadbetter [25] and Davis [4] to guarantee parametric rate of mean integrated square error (MISE) convergence of the KRE. Figure 3.8 illustrates the MSE as a function of bandwidth.

Comparison of the optimal mean squared error (optimised over σ) corresponding to the three kernels revealed that the exponential kernel gave a smaller MSE and is therefore better suited to the regression problem than the gaussian and rectangular kernels. The sinc kernel gave the best optimal MSE for the regression problem.

The above comparative results are summarized in table 3.1.

kernel type	*optimal MSE*
Rectangular	0.362599
Gaussian	0.263368
Exponential	0.254209
Sinc	0.140048

Table 3.1: Comparison of MSE optimized over bandwidth, obtained using different kernels.

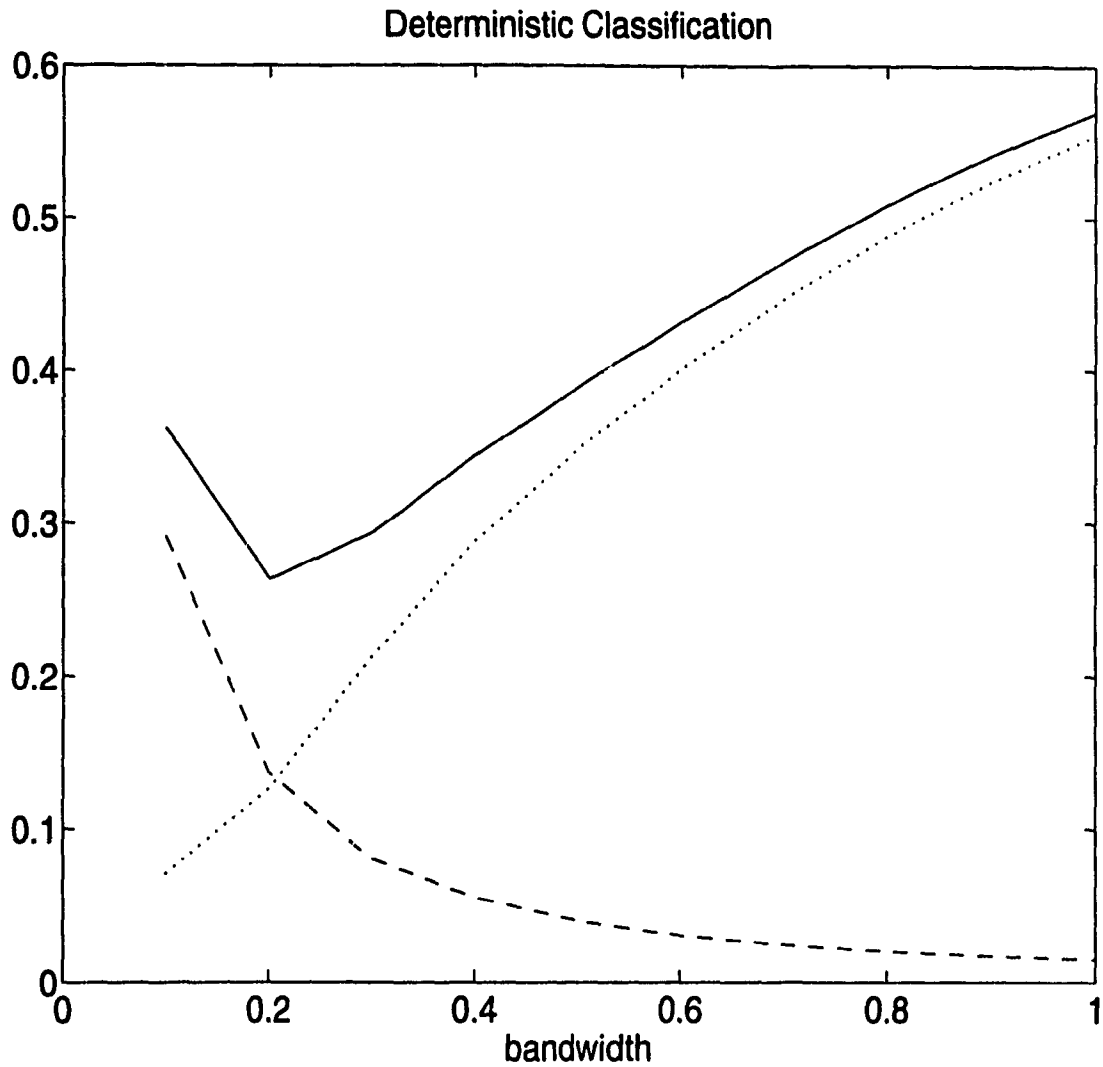


Figure 3.4: Bias(dots), variance(dashes) and mean squared error(solid) as functions of the bandwidth in the KRE with the *gaussian* kernel, when the classification is *deterministic.*

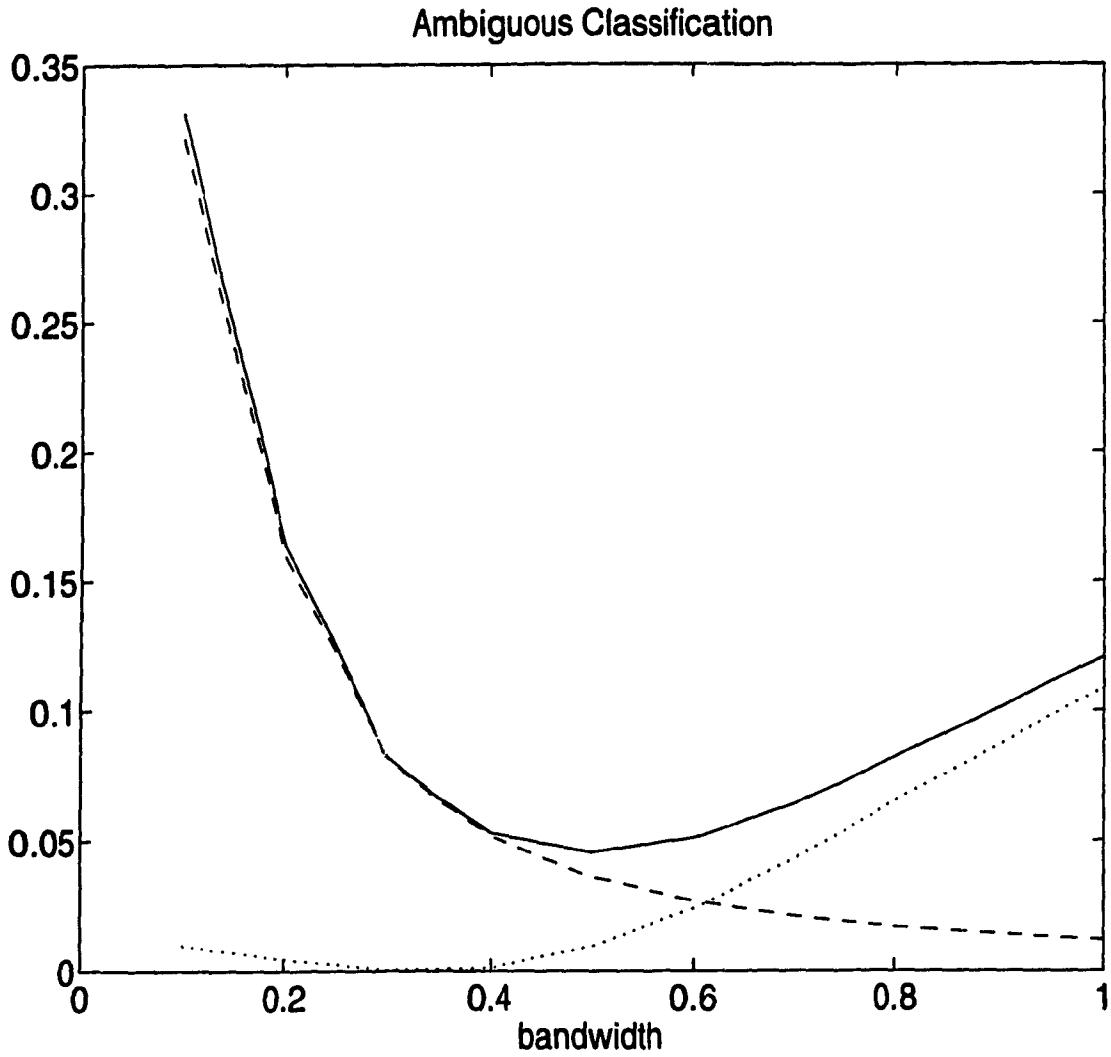


Figure 3.5: Bias(dots), variance(dashes) and mean squared error(solid) as functions of the bandwidth in the KRE with the *gaussian* kernel, when the classification is *ambiguous*.

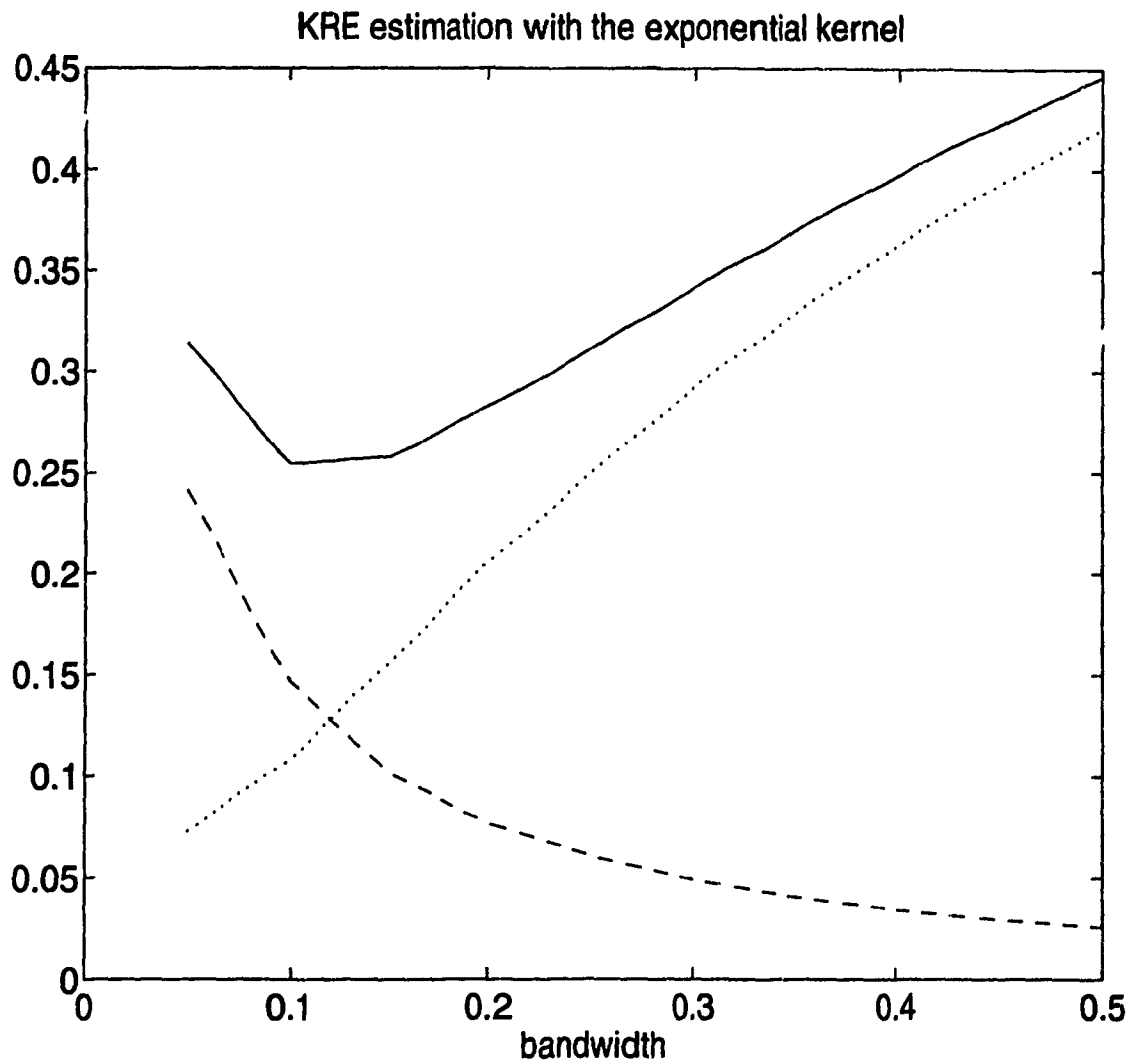


Figure 3.6: Bias(dots), variance(dashes) and mean squared error(solid) as functions of the bandwidth in the KRE with the *exponential* kernel.

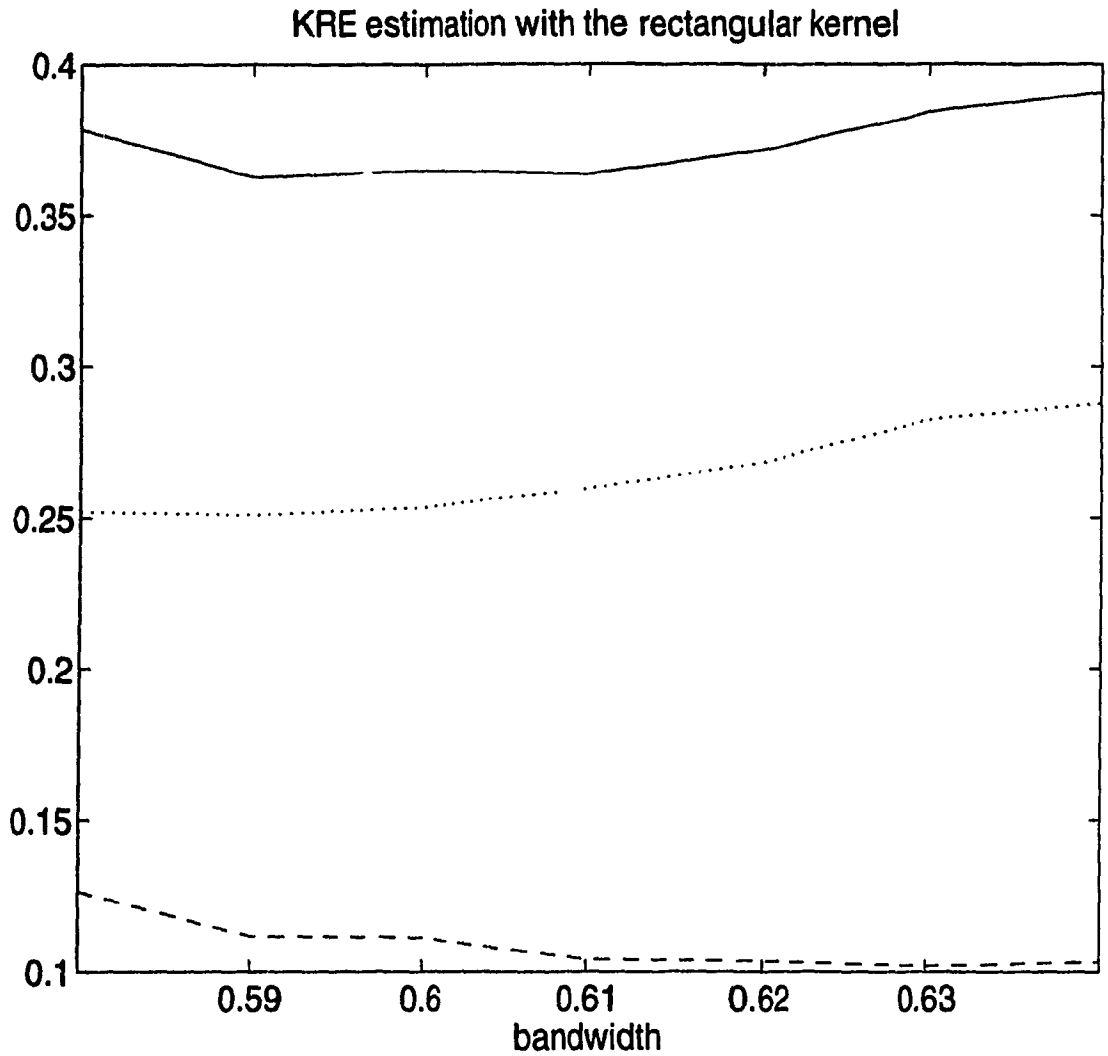


Figure 3.7: Bias(dots), variance(dashes) and mean squared error(solid) as functions of the bandwidth in the KRE with the *rectangular* kernel.

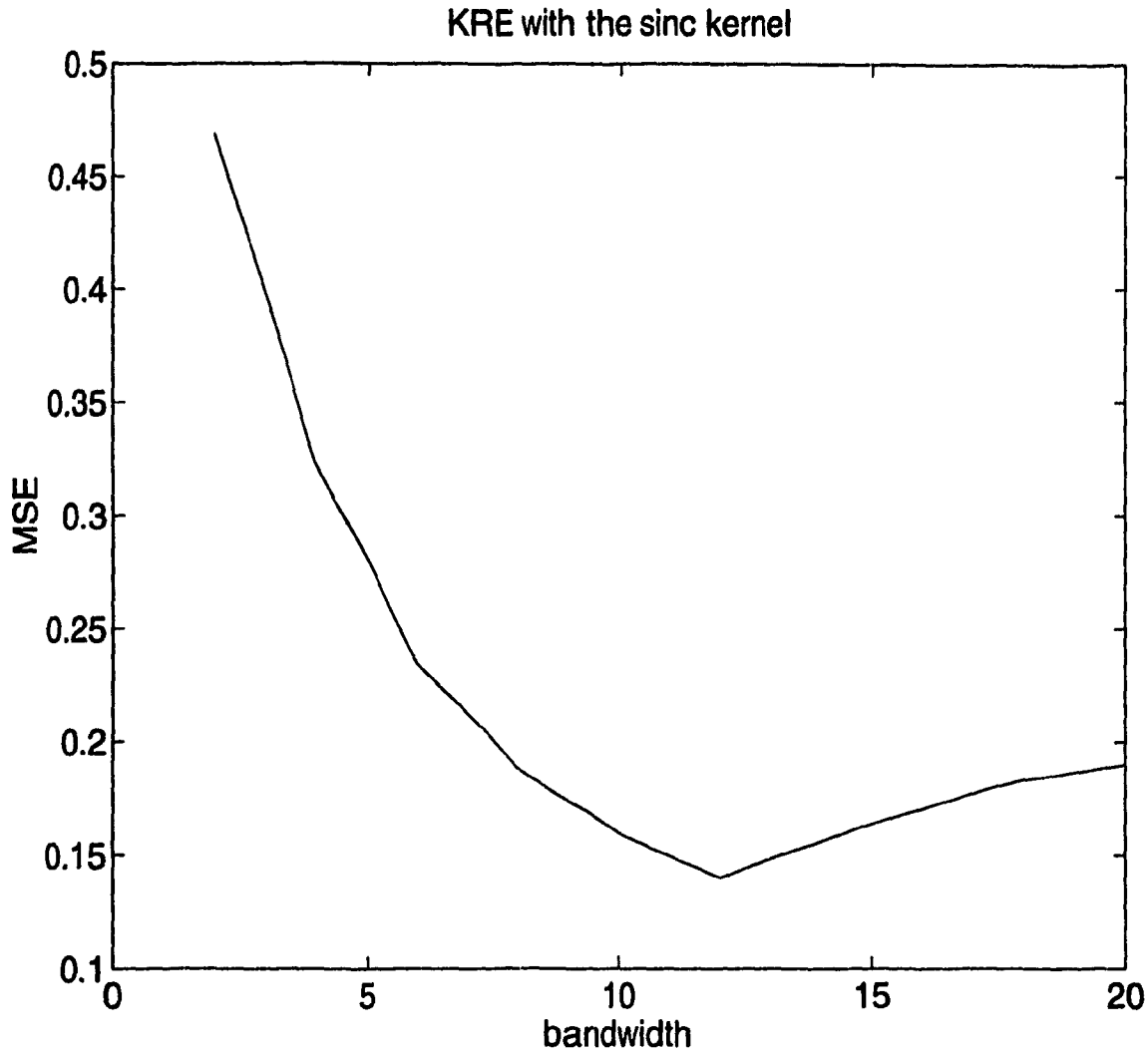


Figure 3.8: Mean squared error as a function of the bandwidth in the KRE with the *sinc* kernel.

Chapter 4

The RBF approach to the learning problem

In this chapter, we study the RBF net in detail. We describe the various models of this network existent in the literature and explain the particular model which we use for our experiments in Chapter 5. We discuss the various parameters involved in the network and describe the optimization of the mean squared error with respect to these parameters. Further, we consider various ways of selecting the center parameters of the network and conclude with a comparison of the RBF net and the KRE.

4.1 Estimation through RBF nets

Basis functions

In neural network research, after several years of extensive study on the multilayer perceptron, researchers have focussed their attention on a number of other models for feedforward neural networks. Amongst these, the biggest group consists of those models that implement function approximation or probability density estimation by

basis function expansion. These networks can be broadly represented as :

$$f_i(\mathbf{x}) = \sum_{j=1}^{n} w_{ij}\phi_j(\mathbf{x}), i = 1, \ldots, m. \tag{4.1}$$

Represented in the network terminology, the above functions correspond to a network architecture consisting of one hidden layer and n hidden neurons. Each hidden neuron is fully connected to all components of the input $\mathbf{x}$ and represents a basis function $b_j = \phi_j(\mathbf{x})$. The output layer has m neurons - each neuron being a linear summation neuron $\phi_i = \sum_{j=1}^{n} w_{ij}b_i$ with weight vector $\mathbf{w}_i = [w_{i1}, \ldots, w_{im}]^t$.

This architecture can be broadly perceived as a variant of the one-hidden layer perceptron by changing the hidden neuron's functions. However, a key difference that categorizes these networks into a group of new models is that this new architecture decides the updating of the function $\phi_j(\mathbf{x})$ based on values that are predetermined externally or specified directly by training samples. For instance, in the RBF network, as discussed in the next section, this updating is done by radially symmetric basis functions. Figures 4.1 and 4.2 illustrate the sigmoidal nets and the RBF nets respectively.

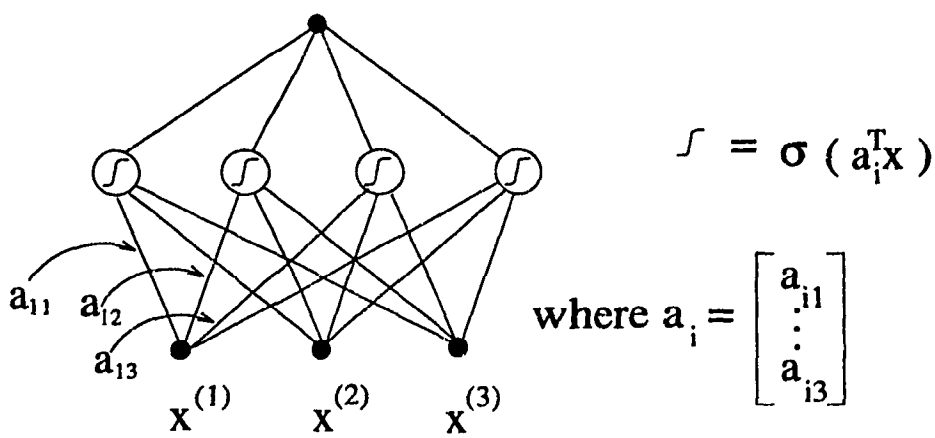


Figure 4.1: Sigmoidal network with one hidden layer and 3-dimensional input

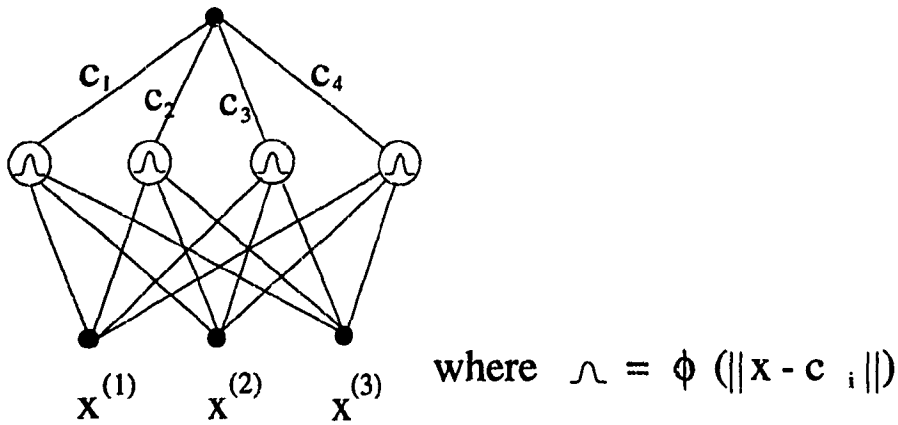


Figure 4.2: RBF network with one hidden layer and 3-dimensional input

The updating in the basis function networks has the following direct advantage: Given a training set $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_N, \mathbf{y}_N)\}$, the weights $w_{ij}, i = 1, \ldots, m, j = 1, \ldots, n$ of the output layer can be easily determined if the basis functions $b_j = \phi_j(\mathbf{x}), j = 1, \ldots, n$ are known *a priori*, since the weights $W = [w_{ij}]_{m \times n}$ can then be obtained by minimizing the least square error E:

$$\begin{aligned} E &= \sum_{k=1}^{N} \|y_k - W\mathbf{b}_k\|^2 \\ \mathbf{b}_k &= [b_1^{(k)}, \ldots, b_n^{(k)}]^t \\ b_j^{(k)} &= \phi_j(\mathbf{x}) \end{aligned} \tag{4.2}$$

Selection of appropriate basis functions $\phi_j(\mathbf{x}), j = 1, \ldots, n$ can be made in several ways, resulting in different models of basis functions. Based on the different types of basis functions currently used in the neural network literature, we can roughly divide these models into two groups [26]. The first group called *localized basis functions* consists of functions that can be expressed in the following general form

$$\phi_j(\mathbf{x}) = \phi(\mathbf{x} - \mathbf{c}_j, \Sigma_j), j = 1, \ldots, n, \tag{4.3}$$

where $\phi(.)$ is called a *mother function.* Each basis function is obtained by locating the mother function at a point given by the locating parameter vector $\mathbf{c}_j$, and may or may not be subject to some deformation caused by the symmetric matrix Σ_j. The second group called *nonlocalized basis functions* comprises of functions that are not expressible by equation (4.3).

In our study, we focus our attention on the first group of models – localized basis functions, which has been generating an ever-increasing interest in the literature. Studies of these models have been profuse in recent years. We study the RBF model in detail and explore its connections with the kernel regression estimator.

4.2 The RBF network

Modifications of the parameters in equation (4.1) give rise to several versions of the RBF net. We describe these models, reviewing the advantages and disadvantages of each, leading up to a description of the model that we consider for our experiments.

4.2.1 The Original Model

The original model of the RBF net is obtained simply by letting in equation (4.3), $\Sigma_j = I$ and $\phi(.)$ be radially symmetric around the locating center $\mathbf{c_j} = \mathbf{x_j}$, where $\mathbf{x_j}$ is a training sample. This implies that we let $\phi_j(\mathbf{x}) = \phi(\|\mathbf{x} - \mathbf{x_j}\|)$, where $\|.\|$ stands for the Euclidean norm in $\mathcal{R}^2$. Thus, equation (4.1) can be rewritten as

$$f_i(\mathbf{x}) = \sum_{j=1}^{N} w_{ij}\phi_j\left(\|\mathbf{x} - \mathbf{x}_j\|\right), i = 1, \ldots, m. \tag{4.4}$$

Note that in forming this model, all the samples in the training set have been used.

The possibilities for the selection of $\phi(.)$ are many. The most common choice is the Gaussian function $\phi(r) = exp(-r^2)$, but a number of alternatives can be considered [21]. These different basis functions may give good results on a training set but may behave differently on a testing set. Hence we need to select the basis function depending on the problem on hand, in order to achieve a good generalization ability.

It has been shown [21] that the RBF expansion given by equation (4.4) has universal approximation ability.

However, this simple RBF net has the disadvantage that it requires all the training samples, which could indeed be numerous. This has the serious drawback that the costs of computation and storage would be considerably large.

4.2.2 Modifications

Two modifications that remedy the above problem have been proposed. The first is to try and select a subset of the whole training set $\mathcal{D}$ by discarding those points that are not so important, so that the number N of training samples can be reduced to a smaller number $K \ll N$. The second is to modify equation (4.4) into

$$f_i(\mathbf{x}) = \sum_{j=1}^{n_h} w_{ij}\phi_j\left(\|\mathbf{x} - \mathbf{c}_j\|^2\right), i = 1, \ldots, m. \tag{4.5}$$

The idea here is to use a few movable location vectors instead of directly using the training samples as the loci of each basis function. The key question that arises here is : how to determine those $\mathbf{c}_j, j = 1, \ldots, n_h$ which can be regarded as good representatives of the training samples.

One answer to the above question lies in using gradient descent updating through back-propagation [21], but Moody and Darken [18] found such kind of learning to be very slow.

A faster approach to speed up learning is to use the clustering algorithms that have been suggested [18] to find the n_h cluster centers $\mathbf{c}_j, j = 1, \ldots, n_h$. However, one problem of the clustering algorithm is that the number of clusters has to be predefined externally. If this number is not appropriately chosen, the clustering results could be poor, resulting in the poor performance of the RBF net. As a guideline to this choice,

a method called *Rival Penalised Competitive Learning* has been recently proposed [27]. This algorithm automatically decides an appropriate number of clusters for training data.

An other alternative to locate the movable centers $\mathbf{c_j}$ is to randomly select a n-element subset from the training set $\mathcal{D}$ and use every selected sample directly as a center vector.

In our experiments with the RBF net, we explore the last two approaches discussed above. We use the clustering algorithm experimenting with different number of clusters as well as the random-sampling method, for the selection of the movable location vectors. A detailed description of the methods used is given in Chapter 5.

4.2.3 The General Model

The RBF network given by equation (4.5) can be further generalized into

$$f_i(\mathbf{x}) = \sum_{j=1}^{n_h} w_{ij}\phi\left([\mathbf{x}-\mathbf{c}_j]^t \Sigma_j^{-1} [\mathbf{x}-\mathbf{c}_j]\right), i = 1, \ldots, m \tag{4.6}$$

where each $\Sigma_j, j = 1, \ldots, n_h$ is a $p \times p$ semi positive matrix.

The simplest case of this model occurs when

$$\Sigma_j = \sigma_j^2 I_{p\times p}. \tag{4.7}$$

Equation (4.6) in fact reduces to equation (4.5) when $\sigma_j^2 = 1, j = 1, \ldots, n_h$.

Consistency and Approximation

Bounds for the the pointwise and L_2 convergence rates of the least squares estimator for RBF nets have been obtained [28]. For a nonnegative radial basis function $\phi(\mathbf{x}) = H(||\mathbf{x}||)$, where H is a non-increasing function with $H(t) = o(t^{-d})$ and $H \downarrow 0$, the

RBF estimator is pointwise consistent, for $\mathbf{x} \in R^d$ [28]. The approximation ability of RBF nets for L^2 integrable functions have been studied by Park and Sandberg [19, 20]. Further, the generalization ability of RBF nets for a large class of basis functions with the network parameters learned through empirical risk minimization has been proven in nonlinear function estimation [15].

4.2.4 Parameters of the Network and their Optimal Choice

Receptive field of the network

The parameters σ_j^2 in equation (4.7) affect the influential radius $\|\mathbf{x} - \mathbf{c}_j\|^2$ of every basis function located at $\mathbf{c}_j$ and hence are termed as the width of the *receptive field* of the basis function. The receptive field is defined as the support of the function $\max\left[\phi\left([\mathbf{x} - \mathbf{c}_j]^t \Sigma_j^{-1} [\mathbf{x} - \mathbf{c}_j]\right) - a_c, 0\right]$ with $a_c \geq 0$ being a constant. The receptive field, in other words, is the subset of the domain of $\mathbf{x}$ such that $\phi\left([\mathbf{x} - \mathbf{c}_j]^t \Sigma_j^{-1} [\mathbf{x} - \mathbf{c}_j]\right)$ takes on values larger than a pre-specified number a_c. Hence, the receptive field can be interpreted as the range for which an input $\mathbf{x}$ can cause a sufficiently large output.

Different basis functions $\phi(r^2)$ call for different receptive fields. Having fixed a specific basis function, say, a Gaussian $\phi(r^2) = \exp(-r^2)$, the size, shape and orientation of the receptive field are determined by the matrix Σ_j. For example, when $\Sigma_j = \Sigma = \sigma^2 I$, the shape is a hyper-spherical ball with its size given by the value of σ. One more general case for the matrix Σ_j is that it is a diagonal matrix $\Sigma_j = diag[\sigma_{j1}^2, \ldots, \sigma_{jp}^2]$. i.e, the width of each basis function is scaled differently in every dimension. In this case, the shape of the receptive field is an elliptic ball with each axis coinciding with a coordinate axis, and the length of each axis decided by $\sigma_{j1}^2, \ldots, \sigma_{jp}^2$ respectively.

The most general case for the matrix Σ_j is that it is a non-diagonal matrix $\Sigma_j = R^T D R$ with D being a diagonal matrix which determines the shape and size of the receptive field and R being a rotation matrix which determines the orientation of the receptive field.

Choice of parameters

We consider the class of radial basis function networks with one hidden layer and at most n_h nodes, and with a constant matrix Σ. This, from equation (4.6) can be represented as:

$$f_{n_h}(x) = \sum_{j=1}^{n_h} w_j \phi\left([\mathbf{x} - \mathbf{c}_j]^t \Sigma^{-1} [\mathbf{x} - \mathbf{c}_j]\right) \tag{4.8}$$

For our study, we consider the normalized version of equation (4.8) which has often been used [18]:

$$f_{n_h}(x) = \frac{\sum_{j=1}^{n_h} w_j \phi\left([\mathbf{x} - \mathbf{c}_j]^t \Sigma^{-1} [\mathbf{x} - \mathbf{c}_j]\right)}{\sum_{j=1}^{n_h} \phi\left([\mathbf{x} - \mathbf{c}_j]^t \Sigma^{-1} [\mathbf{x} - \mathbf{c}_j]\right)} \tag{4.9}$$

For a given fixed basis function $\phi(r^2)$, the above network, now involves three sets of parameters:

1. the weight vectors $w_j, j = 1, \ldots, n_h$, of the output layer the network.
2. the center vectors $\mathbf{c}_j, i = 1, \ldots, n_h$.
3. the matrix Σ.

However, as described earlier, a key characteristic of the RBF net is that the last two sets of parameters are chosen through values that are either externally specified or specified directly by the training samples. Hence the minimization of the mean squared error given by equation (4.2) becomes a problem of linear optimization with respect to the first set of parameters alone, namely the weight vectors [27]. Given a training set $\{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_N, \mathbf{y}_N)\}$, the set of linear equations given below:

$$E = \frac{1}{N} \sum_{i=1}^{N} |y_i - f_{n_h}(\mathbf{x}_i, \mathbf{w}_i)|^2 \tag{4.10}$$

can be solved by the least squares method with the solution given by:

$$W = YM^T(MM^T)^{-1} \tag{4.11}$$

where $W = [w_1, \ldots, w_{n_h}]$, $Y = [Y_1, \ldots, Y_N]$ and

$$M = [m_{ij}]_{n_h \times N}, m_{ij} = \frac{\phi_{ij}}{\sum_{i=1}^{n_h} \phi_{ij}}, \phi_{ij} = \phi([X_j - \mathbf{c}_i]^t \Sigma^{-1} [X_j - \mathbf{c}_i]) \tag{4.12}$$

We use the above procedure in our experiments with the RBF network for the determination of optimal weights. The experiments and the results are described in Chapter 5.

4.3 Connections between KRE and the RBF network

The RBF network given by equation (4.9) shares interesting properties with the KRE described in equation (2.6) due to the close connections that exist between the two [28].

Let (X, Y) be a pair of random vectors in $R^2 \times R^1$ and $f(\mathbf{x}) = E[Y|X = \mathbf{x}]$ be the corresponding regression function. Let $\{\mathbf{x}_i, y_i\}_{i=1}^n$ be a set of n independently identically distributed samples drawn from (X, Y). The kernel regression estimate of $f(\mathbf{x})$ from equation (2.6) can be written as:

$$f(\mathbf{x}; \mathcal{D}) = \frac{\sum_{i=1}^n y_i W[(\mathbf{x} - \mathbf{x}_i)/\sigma]}{\sum_{i=1}^n W[(\mathbf{x} - \mathbf{x}_i)/\sigma]} \tag{4.13}$$

Imposing the following conditions on the kernel W:

$$c_1 H(\|\mathbf{x}\|) \leq W(\mathbf{x}) \leq c_2 H(\|\mathbf{x}\|) \; and \; c I_{\|\mathbf{x}\|) \leq r} \leq W(\mathbf{x}), \tag{4.14}$$

where H is a non-increasing bounded function with $t^d H(t) \rightarrow 0$ as $t \rightarrow \infty$ and c_1, c_2, c, r are positive constants, Krzyzak (1986) shows that these conditions are

nearly as strong as assuming spherical symmetry of the kernel $W(x)$. Hence, assuming spherical symmetry of the kernel W, i.e letting $W(r^2) = \phi(r^2)$, equation (2.6) can be written as:

$$f(\mathbf{x}; \mathcal{D}) = \frac{\sum_{i=1}^{n} y_i \phi[\|(\mathbf{x} - \mathbf{x}_i)/\sigma\|]}{\sum_{i=1}^{n} \phi[\|(\mathbf{x} - \mathbf{x}_i)/\sigma\|]} \tag{4.15}$$

Now, if we let

$$\Sigma = \sigma^2 I, \ \ and \ \mathbf{w}_i = y_i, \mathbf{c}_i = \mathbf{x}_i, i = 1, \ldots, n_h, \tag{4.16}$$

in equation (4.9), then we see that equation (4.15) is identical to equation 4.9 with $n = n_h$. This indicates that the spherically symmetric kernel $W(r^2)$ is in fact just a type of basis function where the smoothing parameter σ represents the size of the basis function's receptive field and y_i acts as an approximate solution to w_i.

Further, Xu, Krzyzak and Yuille [28] pointed out that under the assumption of a hyper-spherically shaped receptive field in the RBF net, the optimal weight vectors $w_i, i = 1, \ldots, n$ of the normalized RBF net (4.9) obtained from equation (4.11) approximately equal the response vectors $y_i, i = 1, \ldots, n$. As a result of this, the RBF nets with $\Sigma = \sigma^2 I$, and $\mathbf{c}_i = \mathbf{x}_i, i = 1, \ldots, n_h$,are approximately identical with the KRE.

We illustrate these connections in our experiments in the comparison of RBF nets and KRE. The description and results of these experiments are detailed in Chapter 5.

Chapter 5

Experiments with the RBF network

In this chapter, we explore the RBF network in detail. We implement the random sampling and clustering methods for selecting the center parameters of the network discussed in Chapter 3 and study the performance of the net in each method. Further, we compare the behavior of the RBF net with that of the KRE estimator in view of the close connections between the two estimators detailed in Chapter 3.

5.1 Experimental models

5.1.1 The RBF Model

Consider the normalized model of the RBF given by equation (4.9):

$$f_{n_h}(x) = \frac{\sum_{j=1}^{n_h} w_j \phi\left([\mathbf{x} - \mathbf{c}_j]^t \Sigma^{-1} [\mathbf{x} - \mathbf{c}_j]\right)}{\sum_{j=1}^{n_h} \phi\left([\mathbf{x} - \mathbf{c}_j]^t \Sigma^{-1} [\mathbf{x} - \mathbf{c}_j]\right)}$$

As discussed earlier, the above network involves three sets of parameters namely the weight vectors $w_j, j = 1, \ldots, n_h$, the center vectors $\mathbf{c_j}, i = 1, \ldots, n_h$, and the matrix Σ. For convenience, we denote Θ to be the vector consisting of all these parameters. Each specific value of Θ yields a specified RBF net. In the learning problem, we aim to determine a specific value $\hat{\Theta}$ for Θ. Given a training set $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, we decide a value of $\hat{\Theta}$ based on the minimization of the mean squared error (4.2) which can be written as

$$E = \frac{1}{N} \sum_{i=1}^{N} \left| y_i - f_{n_h}(\mathbf{x}_i, \hat{\Theta}) \right|^2$$

However, the center vectors $\mathbf{c}_j, i = 1, \ldots, n_h$ and the matrix Σ are usually specified externally or chosen directly from the training samples (see Chapter 3). Hence the above minimization is performed with respect to the weight vectors $\mathbf{w}_j, j = 1, \ldots, n_h$. The mean squared error that we consider to analyse the performances of the RBF net and the KRE estimator is therefore given by

$$E = \frac{1}{N} \sum_{i=1}^{N} |y_i - f_{n_h}(\mathbf{x}_i, \mathbf{w}_i)|^2. \tag{5.1}$$

Recalling the solution of this set of linear equations from equation (4.11), the optimal weights that we use in the experiments with the RBF net are given by

$$W = YM^T(MM^T)^{-1} \tag{5.2}$$

where $\quad W = [w_1, \ldots, w_{n_h}], \quad Y = [Y_1, \ldots, Y_N] \quad$ and

$$M = [m_{ij}]_{n_h \times N}, m_{ij} = \frac{\phi_{ij}}{\sum_{i=1}^{n_h} \phi_{ij}}, \phi_{ij} = \phi([X_j - \mathbf{c}_i]^t \Sigma^{-1} [X_j - \mathbf{c}_i]) \tag{5.3}$$

We choose Σ to be a hyper-spherically shaped receptive field given by $\Sigma = \sigma^2 I$. The kernel ϕ is chosen to be the gaussian kernel $\phi(r^2) = e^{-r^2}$.

With the above specifications, the normalized RBF model (4.9) can be expressed as

$$f_{n_h}(x) = \frac{\sum_{j=1}^{n_h} w_j \phi\left(\frac{\|\mathbf{x}-\mathbf{c}_j\|}{\sigma^2}\right)}{\sum_{j=1}^{n_h} \phi\left(\frac{\|\mathbf{x}-\mathbf{c}_j\|}{\sigma^2}\right)} \tag{5.4}$$

where $\|.\|$ stands for the Euclidean norm in $\mathcal{R}^2$.

Choosing the kernel to be the gaussian function $\phi(r^2) = e^{-r^2}$, we obtain the following RBF net:

$$f_{n_h}(x) = \frac{\sum_{j=1}^{n_h} w_j \exp\left(\frac{-\|\mathbf{x}-\mathbf{c}_j\|}{\sigma^2}\right)}{\sum_{j=1}^{n_h} \exp\left(\frac{-\|\mathbf{x}-\mathbf{c}_j\|}{\sigma^2}\right)} \tag{5.5}$$

In the experiments that follow, we consider this model of the RBF net with the center vectors chosen by clustering and random sampling methods described in Chapter 3 and the weight vectors given by the optimal solution (4.11). We perform the experiments with two kinds of data: (1) *clustered data* and (2) *unclustered data.* The purpose of such a choice was to weigh the performance of the RBF net when the center vectors are chosen by *random sampling* and *clustering* methods and to study the dependence of this choice on the form of the data.

5.1.2 The KRE Model

The kernel regression estimate of the regression function with the gaussian kernel can be written from equation (2.6) as

$$f_n(x) = \frac{\sum_{j=1}^{n} y_j \exp\left(\frac{-\|\mathbf{x}-\mathbf{x}_j\|}{\sigma^2}\right)}{\sum_{j=1}^{n} \exp\left(\frac{-\|\mathbf{x}-\mathbf{x}_j\|}{\sigma^2}\right)} \tag{5.6}$$

which is analogous to the RBF net (5.5).

Parallel to the mean squared error of the RBF net given by equation (5.1), the mean squared error E' of the KRE estimator can be written as

$$E' = \frac{1}{N}\sum_{i=1}^{N} |y_i - f_N(\mathbf{x}_i)|^2. \tag{5.7}$$

In the following sections, we describe experiments performed with the RBF net (5.5) and the Kernel regression estimator (5.6), the performance of these estimators being assessed by the mean squared errors (5.1) and (5.7). The training set $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$ is composed in such a way that the $\mathbf{x}_i$ are chosen from the distributions described below and the responses y_i are designed by the deterministic classification procedure described in Chapter 4.

5.2 Generation of Clustered Data

In generating data which is clustered, we use an approach which is related to a popular approach to clustering based on the notion of a *mixture density.*

Each pattern or sample is assumed to be drawn from one of K underlying populations, or clusters. Here, we assume knowledge about the form and the number of underlying population densities, i.e, the samples $(\mathbf{x}_1, \ldots, \mathbf{x}_N)$ are drawn from a population with a known number of clusters. The samples may or may not be labeled by population. We denote the underlying probability density function for cluster ω_i by $p(\mathbf{x}|\omega_i)$. Then, if $P(\omega_i)$ is the *a priori* probability of class ω_i, or the chance that a sample comes from ω_i, the mixture density can be written as

$$p(\mathbf{x}) = \sum_{i=1}^{K} p(\mathbf{x}|\omega_i) P(\omega_i) \tag{5.8}$$

The class-conditional densities $p(\mathbf{x}|\omega_i)$ are usually called the *component densities*, and the *a priori* probabilities $P(\omega_i)$ are termed as the *mixing parameters.*

Note that

$$\sum_{i=1}^{K} P(w_i) = 1 \tag{5.9}$$

Using the above procedure we generate data which is a mixture of several gaussians, by considering the underlying population densities $p(\mathbf{x}|\omega_i)$ to be gaussian and by choosing the *a priori* probabilities $P(\omega_i)$ in such a way that (5.9) is satisfied.

We first generate the data as a mixture of two gaussian densities $\mathcal{N}_1$ and $\mathcal{N}_2$. This generation was done based on the following algorithm:

Step 1 Choose a number α between 0 and 1.

Step 2 Select a random number r from the uniform distribution on (0,1).

Step 3 If $r \leq \alpha$ then draw the required mixed variate sample x from $\mathcal{N}_1$. Else, select x from $\mathcal{N}_2$.

Since Step 3 implies that the *a priori* probabilities of the two populations of $\mathcal{N}_1$ and $\mathcal{N}_2$ are respectively α and $1-\alpha$, it follows that the required sample $x \in \alpha\mathcal{N}_1 + (1-\alpha)\mathcal{N}_2$ Note that the mixed parameters add up to one and hence satisfy condition (5.9).

We chose the underlying population densities to be $\mathcal{N}(0,1)$ and $\mathcal{N}(1,1)$ and used the above procedure to generate data with two clusters.

We also experiment with data containing three clusters, by generating data as a mixture of three gaussians $\mathcal{N}_1$, $\mathcal{N}_2$, and $\mathcal{N}_3$. The above algorithm can be tailored for this purpose, by the following simple modification:

Step 1 Choose two numbers α_1 and α_2 between 0 and 1, with $\alpha_1 \leq \alpha_2$.

Step 2 Select a random number r from the uniform distribution on (0,1).

Step 3 If $r \leq \alpha_1$, then draw the required mixed variate sample x from $\mathcal{N}_1$. If $\alpha_1 < r \leq \alpha_2$, then select x from $\mathcal{N}_2$. Else, select x from $\mathcal{N}_3$.

The required sample $x \in \alpha_1\mathcal{N}_1 + (\alpha_1 - \alpha_2)\mathcal{N}_2 + (1-\alpha_3)\mathcal{N}_3$. Again note that condition 5.9 is satisfied by the mixing parameters.

We chose the population densities $\mathcal{N}(0,1)$, $\mathcal{N}(2,1)$, and $\mathcal{N}(4,1)$ and generated data containing three clusters.

The samples in the clustered data could be labeled or unlabeled by population. We experiment with both these cases and compare the results.

5.3 The Clustering Method

In the clustering method of choosing center vectors for the RBF net, the center vectors are chosen to be the mean vectors of the clusters in the data. The clustering techniques that can be used to find the cluster centers are multifold : these include the *hierarchical* and *partitional* clustering algorithms [13]. Hierarchical clustering techniques organize the data into a nested sequence of groups while partitional techniques generate a single partition of the data in an attempt to recover natural groups present in the data. For engineering applications where single partitions are important, partitional techniques are appropriate. We therefore follow the partitional clustering in finding the cluster centers of the data.

Principle

The underlying principle of partitional clustering techniques can be stated as follows: Given n samples in a d-dimensional space, a partition of the samples into K groups or clusters is determined in such a way that samples in a cluster are more similar to each other than to samples in different clusters.

The general algorithm to implement the iterative partitional clustering method is outlined below:

Step 1 Select an initial partition with K clusters. Repeat steps 2 and 3 until the cluster membership stabilizes.

Step 2 Generate a new partition by assigning each sample to its closest cluster center.

Step 3 Compute new cluster centers as the centroids of the clusters.

A crucial factor in the algorithm is the selection of the initial partition. As explained in Section 5.2, we generate the clustered data from one of K underlying populations, the form and number of which are assumed to be known. When the samples are unlabeled by population, we use the knowledge about the actual number of clusters in the data to predefine the number of clusters K in the clustering algorithm. We study the performance of the net when K is chosen to be the actual number of clusters in the data, and observe the performance as K is increased beyond the actual number of clusters. On the other hand, when the samples are labeled by population, then the center vectors of the net would simply be given by the sample means of the populations. The results of these experiments are analysed in Section 5.5.

5.4 The Random Sampling Method

In this method, we randomly select K samples from the N samples $(\mathbf{x}_1, \ldots, \mathbf{x}_N)$ and assign these to the center vectors of the net. This means that any one of the $C_N^K = \frac{N!}{K!(N-K)!}$ K-sample subsets is chosen at random. However, without loss of generality, we can assume that this subset simply consists of the first K samples of $\mathcal{D}^x = (\mathbf{x}_1, \ldots, \mathbf{x}_N)$ since if this is not the case, we can re-order the indices in $\mathcal{D}^x$ to let it be true as these indices are originally specified arbitrarily. Hence, in studying the network with center vectors chosen by random sampling, we simply choose $\mathbf{c}_j = \mathbf{x}_j, j = 1, \ldots, K$.

The RBF estimator (5.5) is now given by :

$$f_{n_h}(x) = \frac{\sum_{j=1}^{n_h} w_j \exp\left(\frac{-\|\mathbf{x}-\mathbf{x}_j\|}{\sigma^2}\right)}{\sum_{j=1}^{n_h} \exp\left(\frac{-\|\mathbf{x}-\mathbf{x}_j\|}{\sigma^2}\right)} \tag{5.10}$$

Employing the above estimator for clustered data, we assess the performance of the RBF net when its center vectors are chosen randomly and compare it with the selection through clustering. Further, we also test the RBF estimator (5.10) on unclustered data and study the efficiency of the random sampling method when the data is devoid of clusters. We generate the unclustered data by selecting the samples from

the uniform distribution on (0, 1). The results of these experiments are analysed in Section 5.5 below.

5.5 Analysis of Results

We study the results of the experiments described in the previous sections in two parts – one dealing with the with the random sampling and clustering methods and the second with the comparisons between the RBF and Kernel regression estimators. The figures illustrating these results are placed at the end of the section. In each experiment, we observe the mean squared error as a function of the bandwidth σ and compare the optimal mean squared error (optimized over σ) arising from the different techniques.

5.5.1 Clustering and Random Sampling Methods

In the case when the data is clustered, we first consider the case when the samples are labeled by population. The center vectors of the RBF net are then given simply by the sample means of the population. Figures 5.1 and 5.4 display the results of this experiment when the number of clusters in the data is two and three respectively.

We then consider the case when the samples are unlabeled by population. The center vectors of the net in this case are chosen by the iterative clustering and random sampling methods. In the iterative clustering method, K is initially chosen to be the actual number of clusters in the data and then gradually increased. For each such choice of K in the clustering method, the random sampling method is studied with K points chosen randomly.

Figure 5.2 gives the results for two-clustered data when K is chosen to be two and then increased to three and four, in the partitional algorithm. As seen in the figure, we see that the mean squared error tends to improve when the number of clusters

is increased beyond the actual number in the data. The same trend can be seen in Figure 5.5 also, which deals with three-clustered data. Here K is first chosen to be three and then increased to two, three and four.

Further, comparison of the cases where the samples are labeled and unlabeled by population reveals that though the latter starts by having a higher mean squared error than the former the error in the unlabeled case soon improves and becomes lower than that in the labeled case as the number K is increased. This can be observed by comparing figures 5.1 and 5.2 for two-clustered data and figures 5.4 and 5.5 in the case of three-clustered data.

Finally, for each choice of K in the clustering method, we study the random sampling method with K randomly selected points. Figure 5.3 displays the results when the data has two clusters, with the number of randomly selected points initially matching the number of clusters and then increased to three and four. We see that as we select more points, the mean squared error gradually decreases. This follows from observing that selection of a larger subset from the sample set would capture more information and hence reduce the squared error of the estimator. Analogous results for the three-clustered data are displayed in Figure 5.6 .

Comparison of random sampling with iterative clustering for each K shows that the mean squared error of the estimator is lower in the clustering technique. i.e, the RBF net performs better with its center vectors chosen by clustering rather than random sampling. This is because the random sampling method disregards any natural clusters occurring in the data and randomly selects K samples, while the clustering method incorporates information about the form of the data and attempts to find K *centered* points in the data. By the same token, the error of the estimator decreases with increase in K, much faster in the clustering method than in the random sampling method. As a result, a very large number of samples would have to be randomly chosen from the sample set to appreciably reduce the error in the random sampling technique. Hence, with a reasonably good initial partition, the iterative algorithm is more appropriate when the data is clustered. The clustering algorithm is particularly efficient when there is some knowledge about the number of clusters in the data.

These comparative results are summarized in tables 5.1 and 5.2.

The above reasoning suggests that the performance of the RBF net under the random sampling method would improve if the data were naturally unclustered. Figures 5.7 and 5.8 display the results when the data is generated from the uniform distribution on (0, 1). Comparison of these figures with figures 5.3 and 5.6 shows that this is indeed the case : the RBF net performs appreciably better when random sampling is done on unclustered data.

5.5.2 Comparison between the RBF net and KRE

In this section, we display the results obtained for the mean squared error of the KRE and compare these with the those obtained in the previous section on the RBF net, in view of the close relation these estimators share (See Chapter 3).

The mean squared error of the KRE is displayed in figures 5.9 and 5.10 respectively for the two clustered and three clustered data. We see that the error decreases with the increase in the number n, analogous to the results of the RBF net. Further, comparing these figures with those obtained in the previous section for the RBF net shows that the corresponding mean squared error of the RBF net under any of the methods studied, is lower than those of the KRE. This is because, under the conditions explained in Chapter 3, the KRE can be considered as a special case of the RBF net, the weight vectors in the KRE being chosen as the responses y_i themselves rather than the optimal manner of selection in the RBF net.

Illustrations of all the above experiments with the RBF net and the KRE are listed in the following pages.

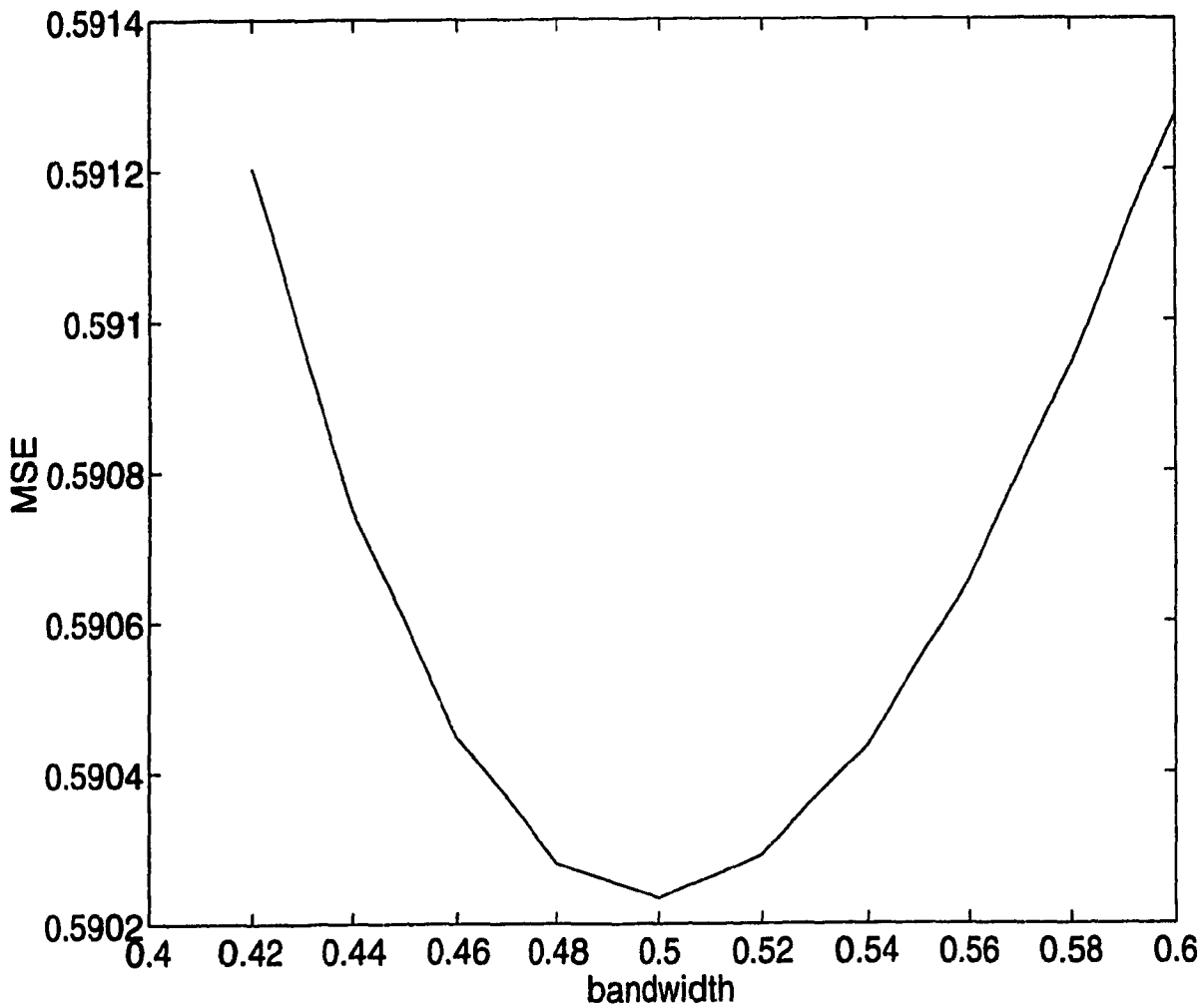


Figure 5.1: Mean squared error as a function of bandwidth when the data has *two* clusters and the samples are labeled by population.

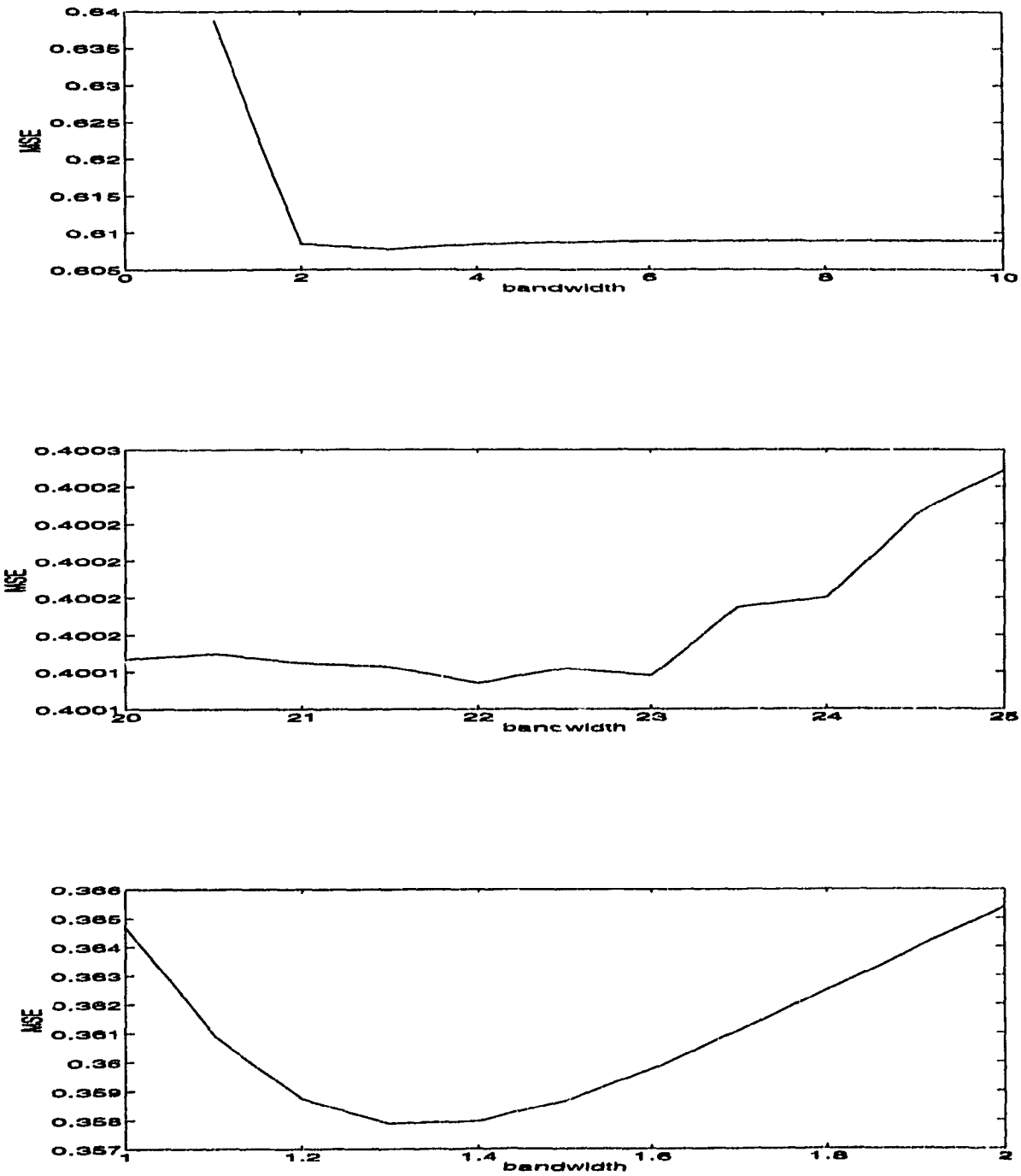


Figure 5.2: Mean squared error as a function of bandwidth when the data has *two* clusters and the clustering algorithm is used with $K = 2$, 3, and 4 respectively.

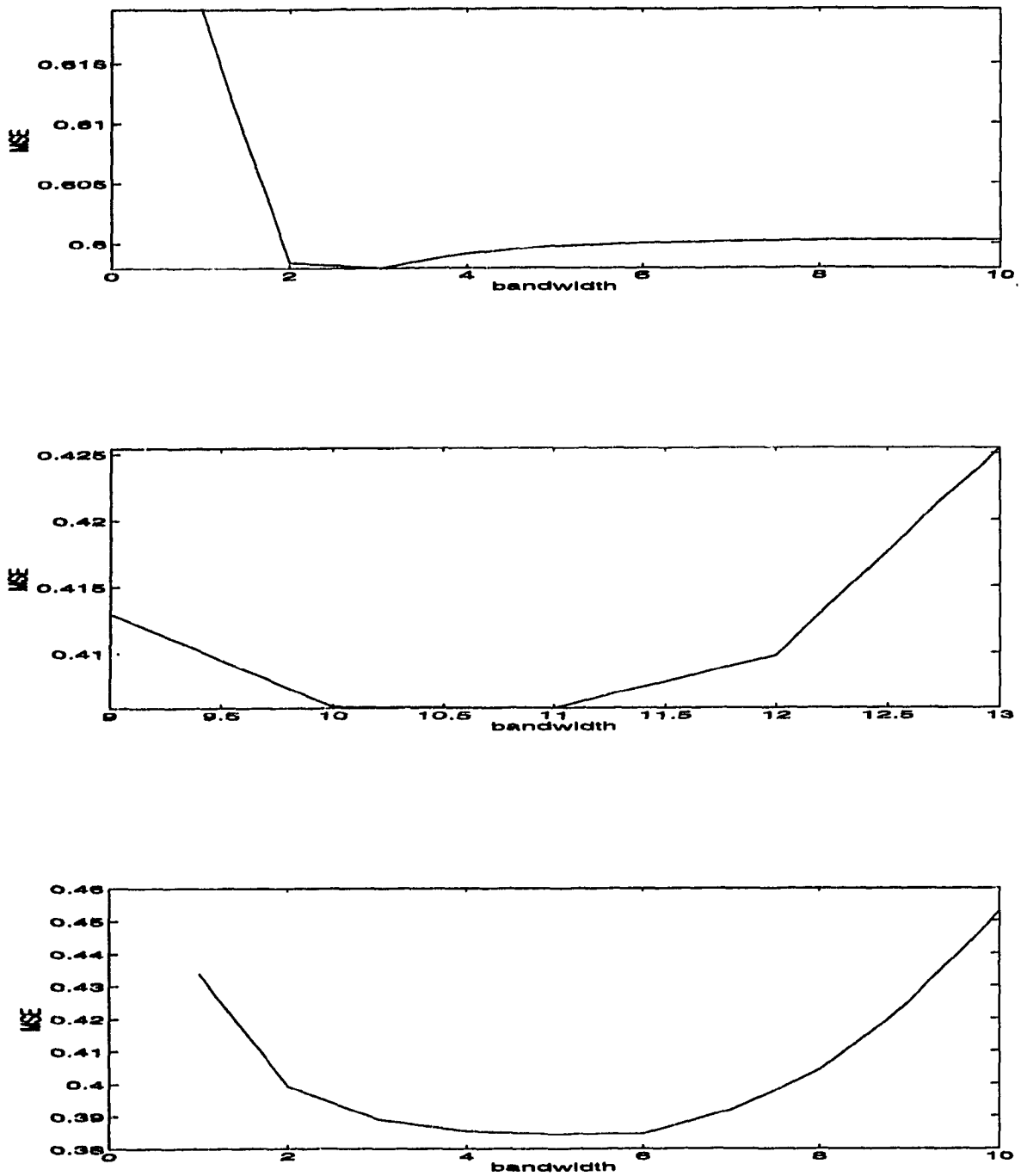


Figure 5.3: Mean squared error as a function of bandwidth when the data has *two* clusters and random sampling is used with $K = 2$, 3, and 4 respectively.

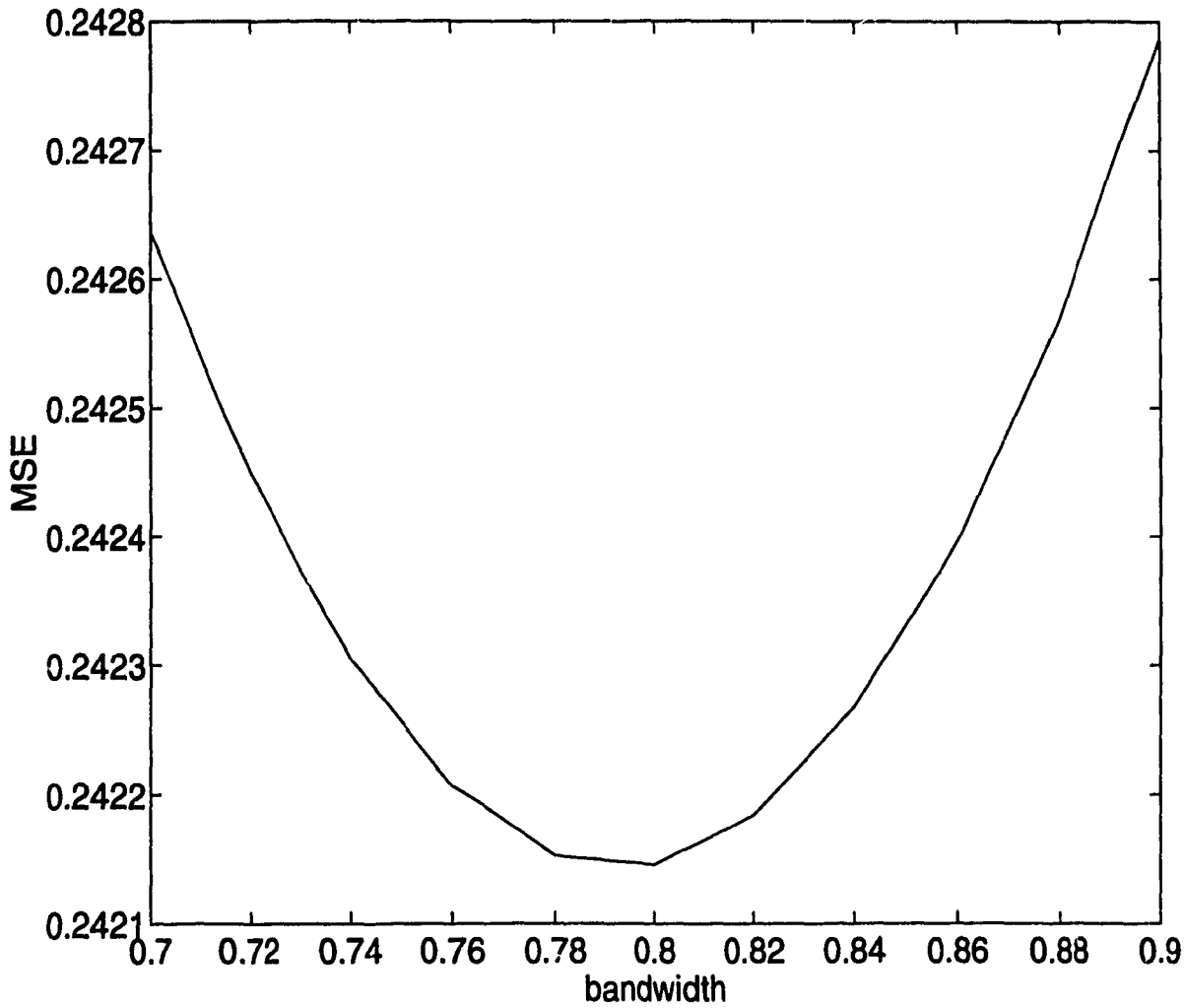


Figure 5.4: Mean squared error as a function of bandwidth when the data has *three* clusters and the samples are labeled by population.

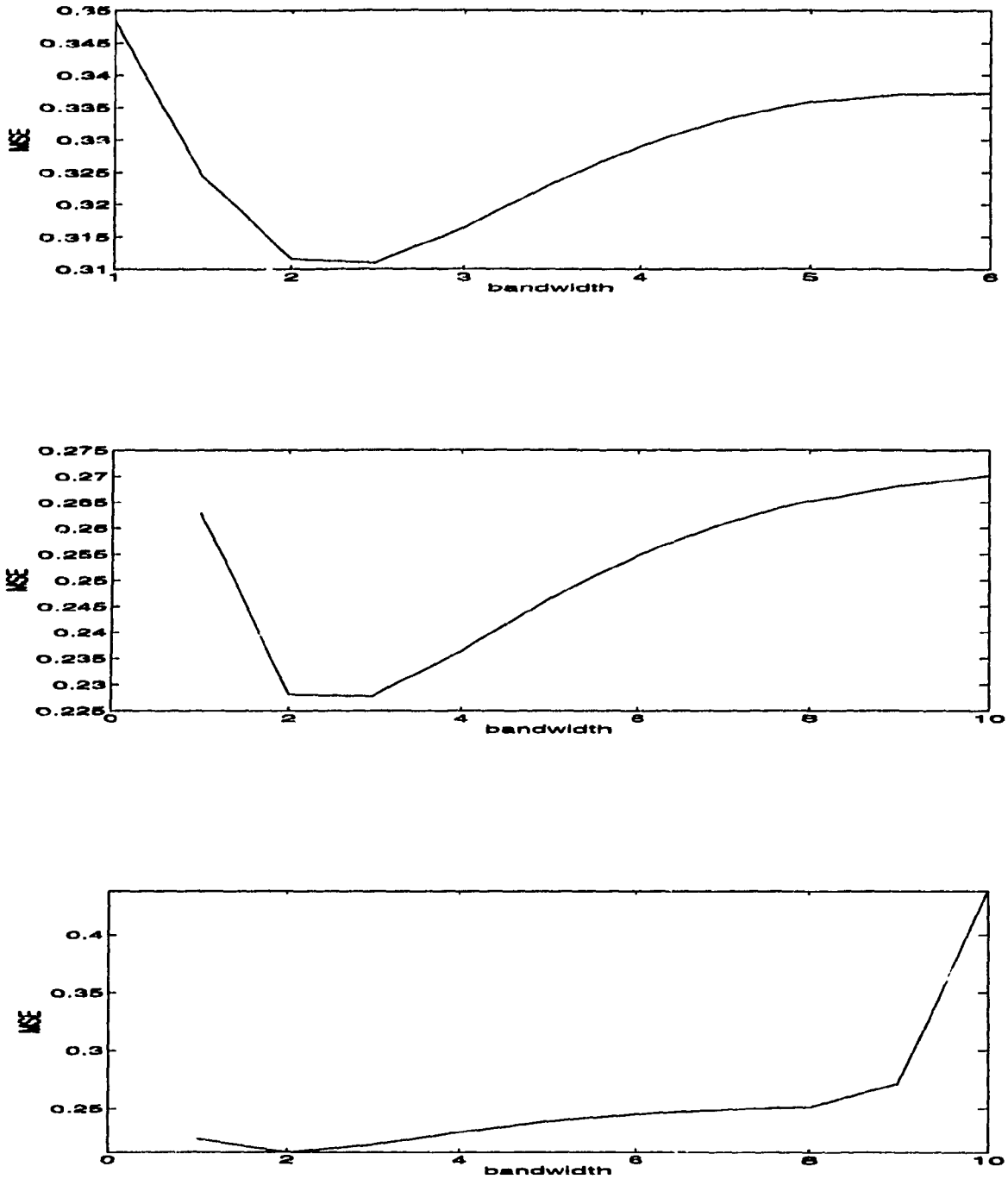


Figure 5.5: Mean squared error as a function of bandwidth when the data has *three* clusters and the clustering algorithm is used with $K = 3$, 4, and 5 respectively.

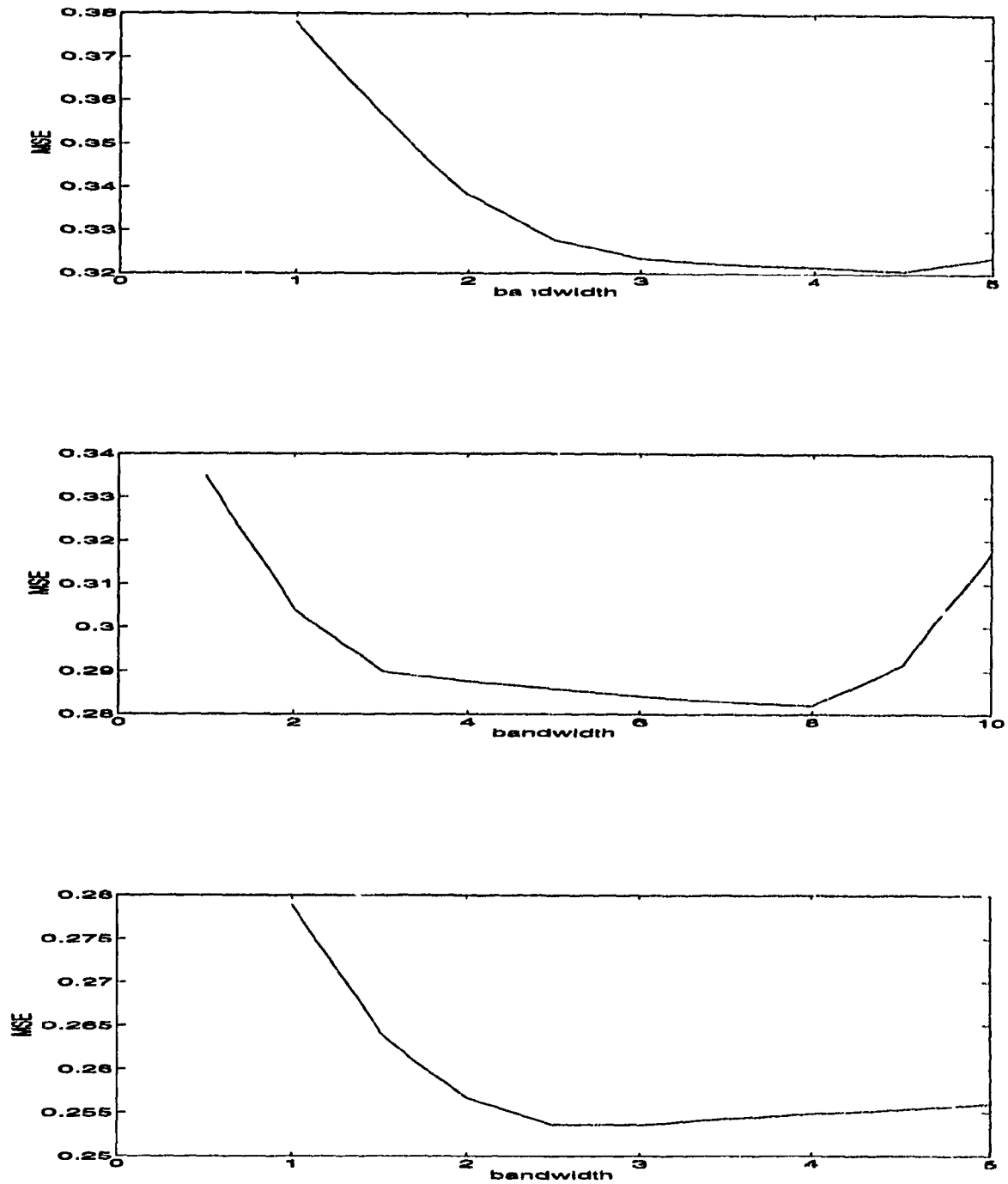
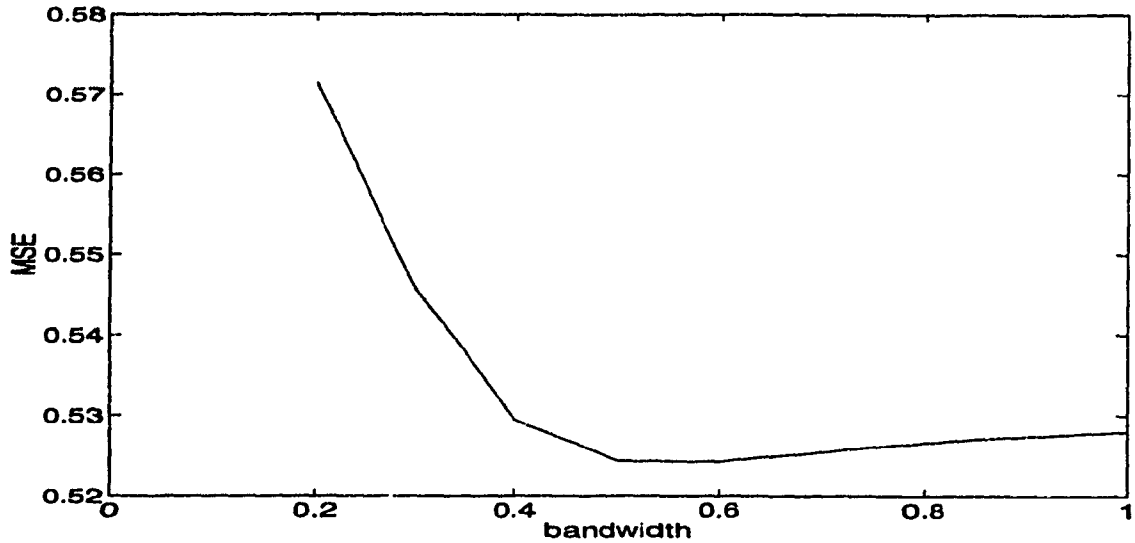


Figure 5.6: Mean squared error as a function of bandwidth when the data has *three* clusters and random sampling is used with $K = 3$, 4, and 5 respectively.



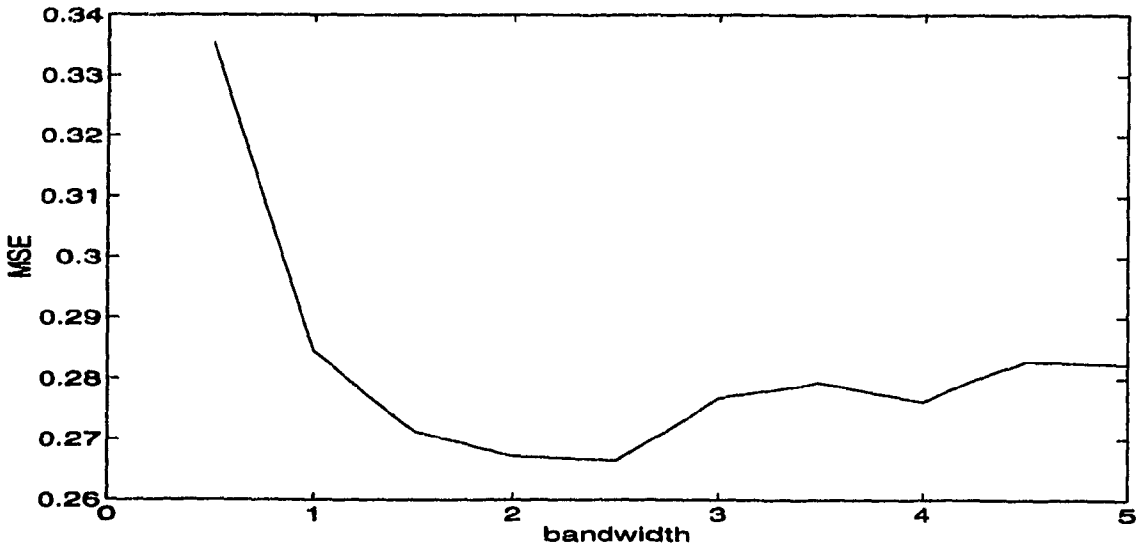


Figure 5.7: Mean squared error as a function of bandwidth when the data is uniformly distributed and random sampling is used with $K = 2$ and 3 respectively.

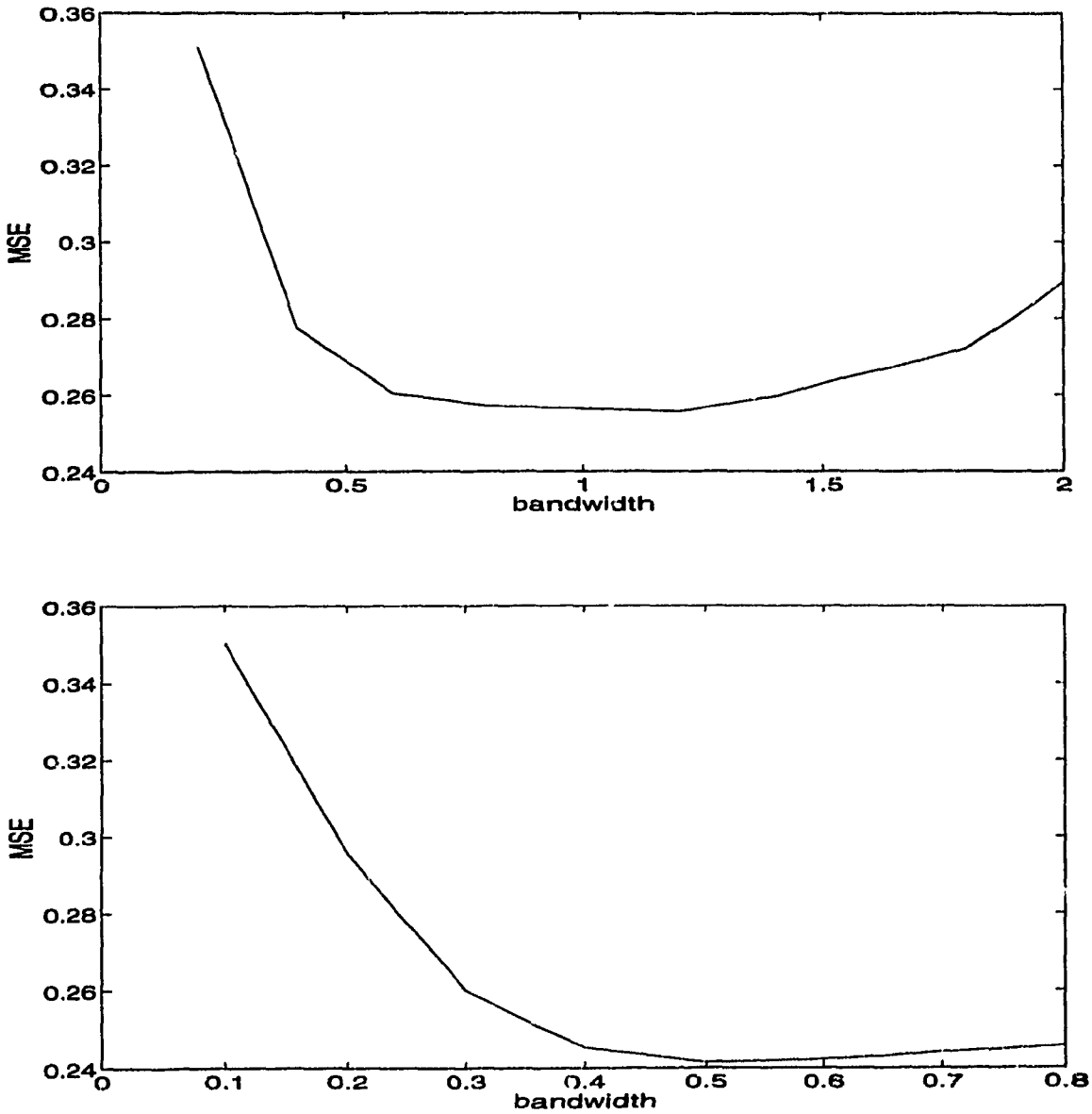


Figure 5.8: Mean squared error as a function of bandwidth when the data is uniformly distributed and random sampling is used with $K = 4$ and 5 respectively.

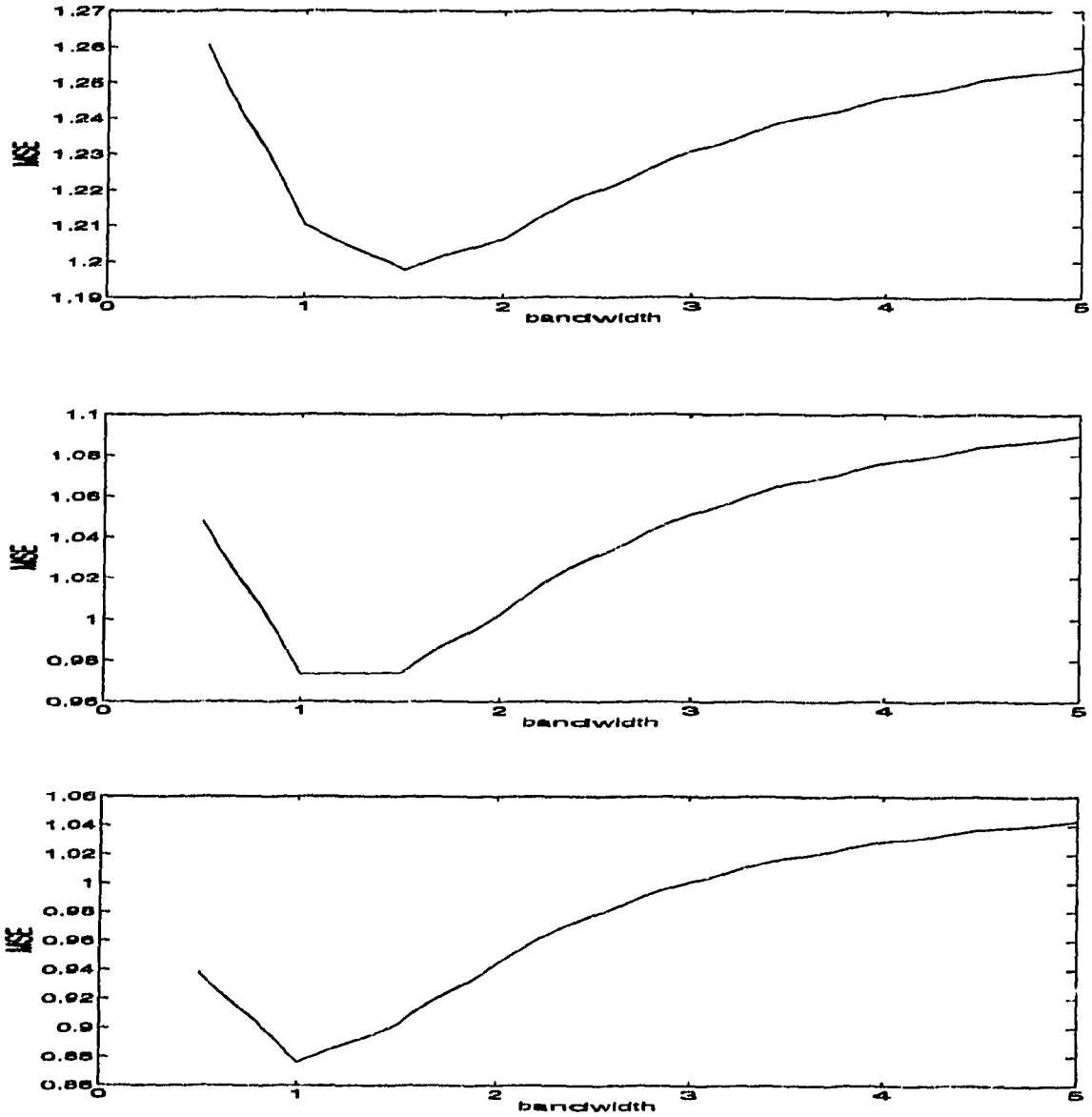


Figure 5.9: Mean squared error as a function of bandwidth when the data has *two* clusters and the KRE is used with $n = 2$, 3, and 4 respectively.

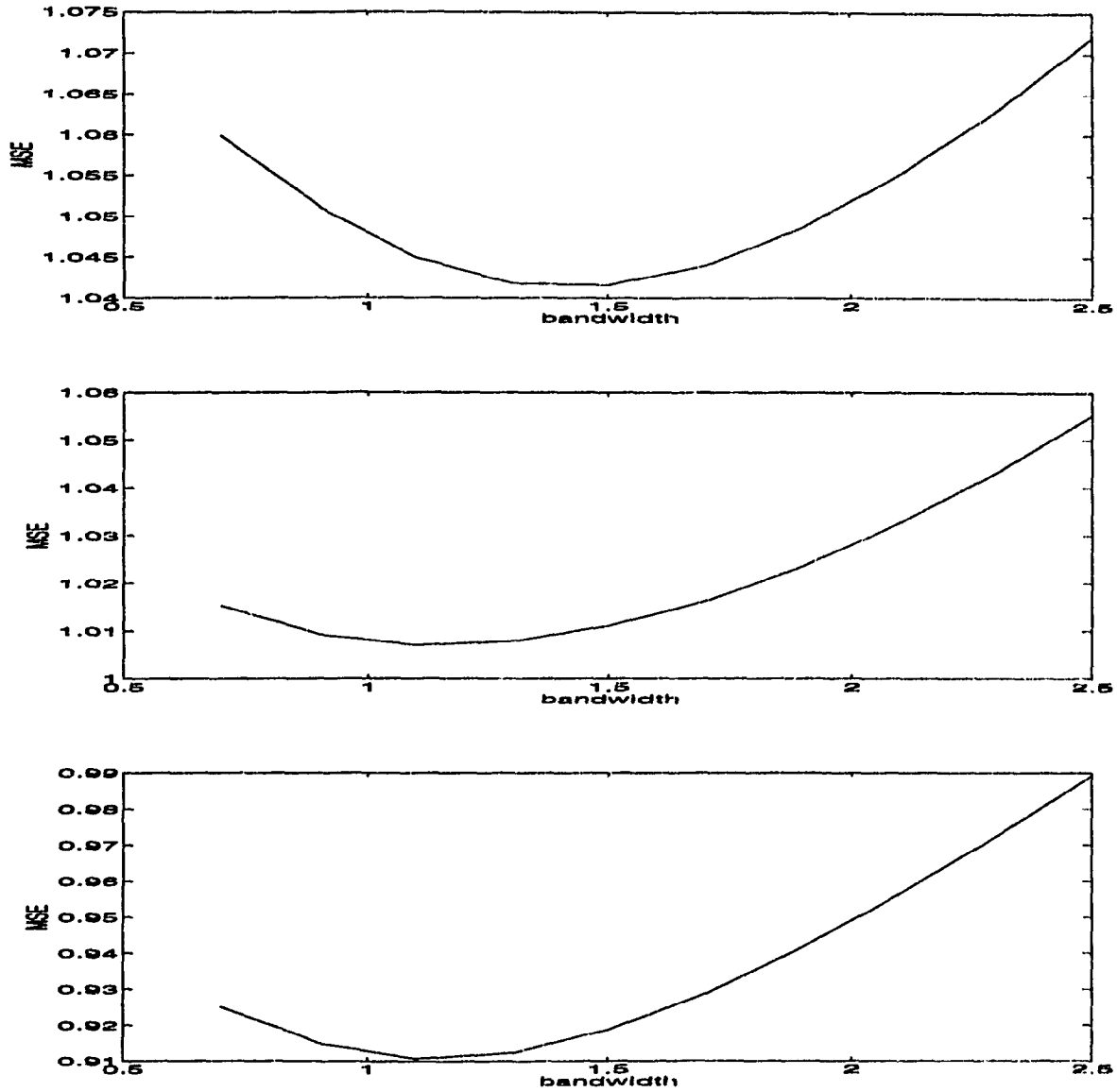


Figure 5.10: Mean squared error as a function of bandwidth when the data has *three* clusters and the KRE is used with $n = 3$, 4, and 5 respectively.

Estimator	*Method of finding centers*	K in method	*optimal MSE*
RBF	From sample means	2	0.590234
	Iterative clustering algorithm	2	0.607768
		3	0.400134
		4	0.357901
	Random sampling	2	0.597921
		3	0.405880
		4	0.384457
KRE	-	2	1.197736
		3	0.973242
		4	0.875832

Table 5.1: Comparison of MSE optimized over bandwidth, obtained using different methods on *two-clustered* data.

Estimator	*Method of finding centers*	K in method	*optimal MSE*
RBF	From sample means	3	0.242146
	Iterative clustering algorithm	3	0.310982
		4	0.227814
		5	0.212280
	Random sampling	3	0.320681
		4	0.282012
		5	0.253618
KRE	-	3	1.041611
		4	1.007027
		5	0.910731

Table 5.2: Comparison of MSE optimized over bandwidth, obtained using different methods on *three-clustered* data.

Conclusions

Our study explored the learning problem from two broad perspectives consisting of the statistical regression estimation and the RBF network formulation. Through the regression formulation of the learning problem, we illustrated the trade-off involved between the bias and variance contributions to the estimation error: while the bias increased, the variance decreased with increase of the parameter concerned in both k-NN and Kernel regression estimation. We analysed the choice of parameters that give an optimal value of the mean squared error in both these estimators. We showed that in a deterministic classification problem, the optimal mean squared error is achieved using a small number of neighbors or small bandwidth as the case may be while in an ambiguous classification problem, the trend of the error is reversed. Through the RBF network formulation of the learning problem, we studied the choice of the parameters of the network that minimize the estimation error by treating this as a linear optimization problem with respect to the weight parameters of the network and choosing all other parameters either externally or directly based on the training samples. We showed that out of the two major methods available for the selection of the center vectors of the network, the clustering method appreciably improves the performance of the net, particularly when the data is clustered, provided the iterative method is started with a reasonably good choice for the number of clusters. The random sampling method, on the other hand improved when the data is unclustered and hence is more appropriate for unclustered data, rather than clustering, because of the simplicity of the former method. Hence any little information available on the form of the data has to be employed to decide on this choice. Finally, we compared the RBF net with the KRE in view of their close relation. Our study revealed that the RBF net performs better than the KRE, irrespective of the choice of the method used in the former to determine the center vectors. This is because of the weights chosen in an optimal manner in the former as opposed to approximating them as the responses, in the latter.

Appendix A

Calculation of Regression in the Ambiguous Classification problem

Consider the random mechanism used for classification, described in section 4.1.2. Given an input $\mathbf{x} = (c_1, c_2)$, the unit disk $S_1(\mathbf{x})$, centered at $\mathbf{x}$, is given by

$$(x_1 - c_1)^2 + (x_2 - c_2)^2 \leq 1$$

where x_1 and x_2 denote the axes of reference.

Having chosen a random point $\mathbf{z} = (z_1, z_2)$ from the uniform distribution on the unit disk, we used the curve

$$x_2 = \sin((\pi/2)x_1)$$

to classify the input vector. Denoting A_1 and A_2 to be the areas above and below the sine curve respectively, bounded by the unit disk, the regression (3.1) was seen to be

$$\mathrm{E}[\mathrm{y} \mid \mathrm{x}] = \frac{0.9}{\pi}(A_1 - A_2) \tag{A.1}$$

Hence computation of the above regression is straightforward if the areas A_1 and A_2 are known.

The computation of these areas depends on where the intersecting points of the sine curve and the unit circle $C : (x_1 - c_1)^2 + (x_2 - c_2)^2 = 1$ lie. For each test vector (c_1, c_2), the points of intersection were obtained numerically from MATLAB. The required areas were then found from the following procedure.

First, noting that

$$A_1 + A_2 = \pi \tag{A.2}$$

we observe that it is sufficient to know one of the areas A_1 and A_2 so that the other can be found from the above relation. The choice of the area computed depends on the case considered. In each case, we derive the simpler area and find the other from the above expression. Knowing the two areas, the regression for each case is given by substituting these into equation A.1. Based on the location of the two intersecting points with respect to the four quadrants, four cases arise: we derive the regression in each case below. We have denoted the points of intersection by (a, b) and (c, d).

Case 1: When both points of intersection lie above the line joining the points $(c_1 - 1, c_2)$ and $(c_1 + 1, c_2)$, i.e, in the first and second quadrants of C (see figure A.1).

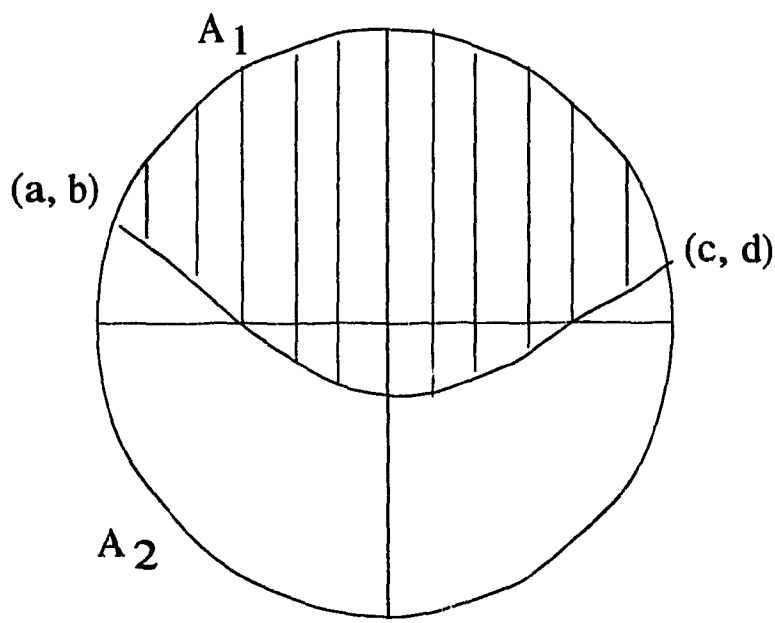


Figure A.1: Case 1

In this case, A_1 is easier to compute than A_2 as the latter requires to be split into

several regions in order to be computed. Hence we compute A_1, which in this case is given by the area between the positive half

$$x_2 = \sqrt{1-(x_1-c_1)^2} + c_2$$

of the circle C, and the sine curve

$$x_2 = \sin((\pi/2)x_1)$$

Hence

$$A_1 = \int_a^c \left[\left(\sqrt{1-(x_1-c_1)^2} + c_2\right) - (\sin((\pi/2)x_1))\right] dx_1$$

where we have assumed (a, b) lies to the left of (c, d).

Integrating the above expression gives

$$\begin{aligned} A_1 &= \frac{1}{2}[\arcsin(c-c_1) - \arcsin(a-c_1)] + \\ &\quad \frac{1}{2}\left[(c-c_1)\sqrt{1-(c-c_1)^2} - (a-c_1)\sqrt{1-(a-c_1)^2}\right] + \\ &\quad c_2(c-a) + \frac{2}{\pi}[\cos((\pi/2)c) - \cos((\pi/2)a)] \end{aligned} \tag{A.3}$$

Substituting equation A.2 into equation A.1, we get the regression entirely in terms of A_1:

$$\mathrm{E}[\mathrm{y} \mid \mathbf{x}] = \frac{0.9}{\pi}(2A_1 - \pi)$$

Thus, we can substitute A_1 calculated from equation A.3 and obtain the regression for this case as

$$\begin{aligned} \mathrm{E}[\mathrm{y} \mid \mathbf{x}] &= \frac{0.9}{\pi}[\arcsin(c-c_1) - \arcsin(a-c_1)] + \\ &\quad \frac{0.9}{\pi}\left[(c-c_1)\sqrt{1-(c-c_1)^2} - (a-c_1)\sqrt{1-(a-c_1)^2}\right] + \\ &\quad \frac{1.8}{\pi}c_2(c-a) + \frac{3.6}{\pi^2}[\cos((\pi/2)c) - \cos((\pi/2)a)] - 0.9 \end{aligned} \tag{A.4}$$

Case 2: When both the points of intersection lie below the line joining the points (c_1-1, c_2) and (c_1+1, c_2), i.e in the third and fourth quadrants of C (see figure A.2).

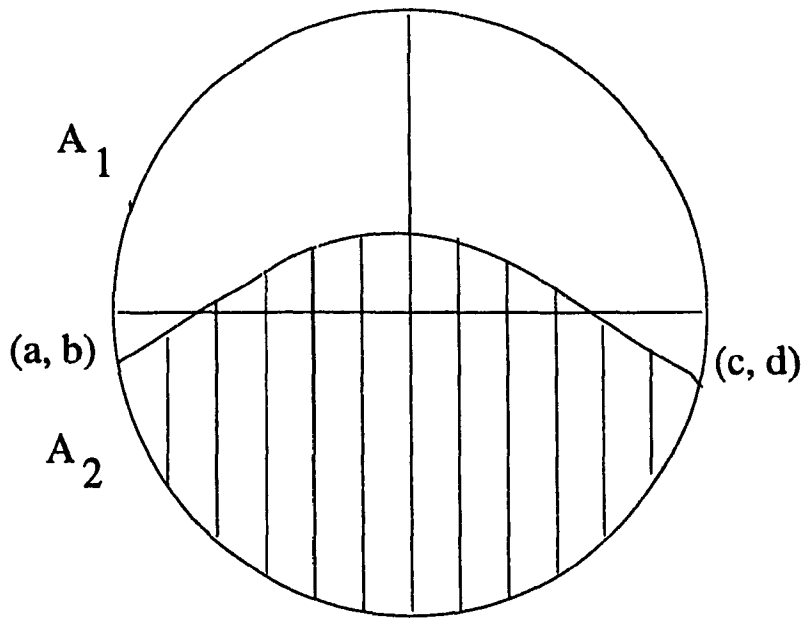


Figure A.2: Case 2

In this case, A_2 is easier to compute than A_1 as the latter requires to be split into several regions in order to be computed. Hence we compute A_2, which in this case is given by the area between the negative half

$$x_2 = -\sqrt{1-(x_1-c_1)^2} + c_2$$

of the circle C, and the sine curve

$$x_2 = \sin((\pi/2)x_1)$$

Hence

$$A_2 = \int_a^c \left[(\sin((\pi/2)x_1)) - \left(-\sqrt{1-(x_1-c_1)^2} + c_2\right)\right] dx_1$$

where we have assumed (a, b) lies to the left of (c, d).

Integrating the above expression gives

$$\begin{aligned} A_1 &= \frac{1}{2}[\arcsin(c-c_1) - \arcsin(a-c_1)] + \\ &\quad \frac{1}{2}\left[(c-c_1)\sqrt{1-(c-c_1)^2} - (a-c_1)\sqrt{1-(a-c_1)^2}\right] - \\ &\quad c_2(c-a) - \frac{2}{\pi}[\cos((\pi/2)c) - \cos((\pi/2)a)] \end{aligned} \tag{A.5}$$

Substituting equation A.2 into equation A.1, we get the regression entirely in terms of A_2:

$$E[y \mid \mathbf{x}] = \frac{0.9}{\pi}(\pi - 2A_2)$$

Thus, we can substitute A_2 calculated from equation A.5 and obtain the regression for this case as

$$\begin{aligned} E[y \mid \mathbf{x}] &= 0.9 - \frac{0.9}{\pi}[\arcsin(c - c_1) - \arcsin(a - c_1)] - \\ &\quad \frac{0.9}{\pi}\left[(c - c_1)\sqrt{1 - (c - c_1)^2} - (a - c_1)\sqrt{1 - (a - c_1)^2}\right] + \\ &\quad \frac{1.8}{\pi}c_2(c - a) + \frac{3.6}{\pi^2}[\cos((\pi/2)c) - \cos((\pi/2)a)] \end{aligned} \tag{A.6}$$

Case 3: When the points of intersection lie on opposite sides of the line joining the points $(c_1 - 1, c_2)$ and $(c_1 + 1, c_2)$: in particular we fix one point say, (a, b) in the third quadrant, and let the other point be in the first or the second quadrant (see figure A.3).

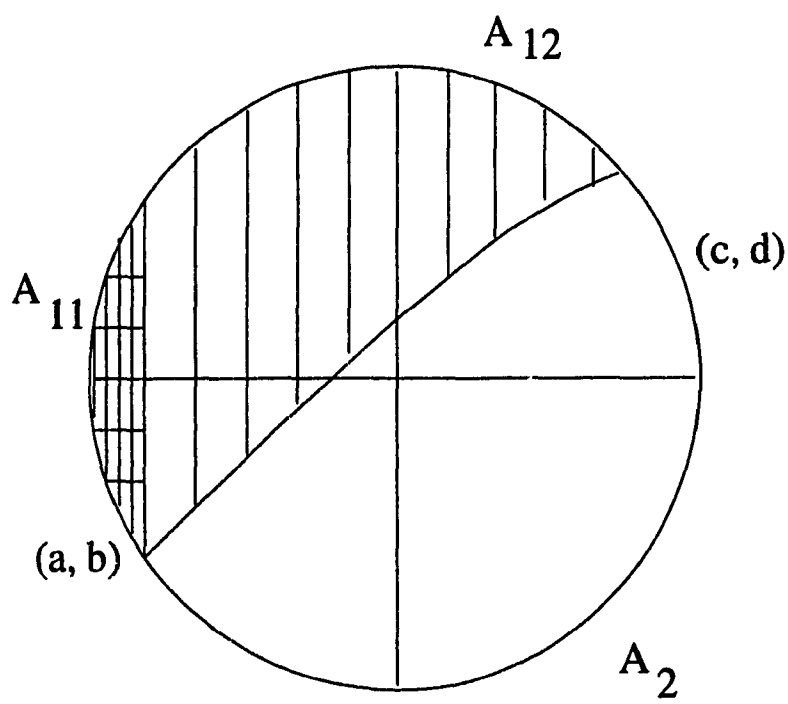


Figure A.3: Case 3

In this case, both areas A_1 and A_2 require to be split into two regions in order to be

computed. We choose to compute A_1 which we express as

$$A_1 = A_{11} + A_{12} \tag{A.7}$$

where A_{11} is given by the area bounded by the line $x_1 = a$ and the left half

$$x_1 = -\sqrt{1-(x_2-c_2)^2} + c_1$$

of the circle $\mathcal{C}$, between the points (a, b) and $(a, \sqrt{1-(a-c_1)^2}+c_2)$ and A_{12} is given by the area bounded by the regions:

$$\begin{aligned} x_1 &= a; \\ x_2 &= \sqrt{1-(x_1-c_1)^2} + c_2 \\ x_2 &= \sin((\pi/2)x_1) \end{aligned} \tag{A.8}$$

Hence

$$A_{11} = \int_b^{\sqrt{1-(a-c_1)^2}+c_2} \left[(a) - (\sqrt{1-(x_2-c_2)^2} + c_1)\right] dx_2$$

Integrating the above expression gives

$$\begin{aligned} A_{11} &= (a-c_1)\left[\sqrt{1-(a-c_1)^2} + c_2 - b\right] + \\ &\quad \frac{1}{2}\left[\arcsin\left(\sqrt{1-(a-c_1)^2}\right) - \arcsin(b-c_2)\right] + \\ &\quad \frac{1}{2}\left[(a-c_1)\sqrt{1-(a-c_1)^2} - (b-c_2)\sqrt{1-(b-c_2)^2}\right] \end{aligned} \tag{A.9}$$

Further, we have

$$A_{12} = \int_a^c \left[\left(\sqrt{1-(x_1-c_1)^2} + c_2\right) - (\sin((\pi/2)x_1))\right] dx_1$$

Integrating the above expression gives

$$\begin{aligned} A_{12} &= \frac{1}{2}[\arcsin(c-c_1) - \arcsin(a-c_1)] + \\ &\quad \frac{1}{2}\left[(c-c_1)\sqrt{1-(c-c_1)^2} - (a-c_1)\sqrt{1-(a-c_1)^2}\right] + \\ &\quad c_2(c-a) + \frac{2}{\pi}[\cos((\pi/2)c) - \cos((\pi/2)a)] \end{aligned} \tag{A.10}$$

Substituting equations A.9 and A.10 into equation A.7, we can get A_1. Writing the regression entirely in term of A_1 from equations A.1 and A.2, we get

$$E[y \mid \mathbf{x}] = \frac{0.9}{\pi}(2A_1 - \pi)$$

Substituting A_1 from the above calculations into the above expression gives the regression for this case.

Case 4: When both the points of intersection lie on opposite sides of the line joining the points $(c_1 - 1, c_2)$ and $(c_1 + 1, c_2)$: in particular we fix one point say, (c, d) in the fourth quadrant, and let the other point be in the first or the second quadrant (see figure A.4).

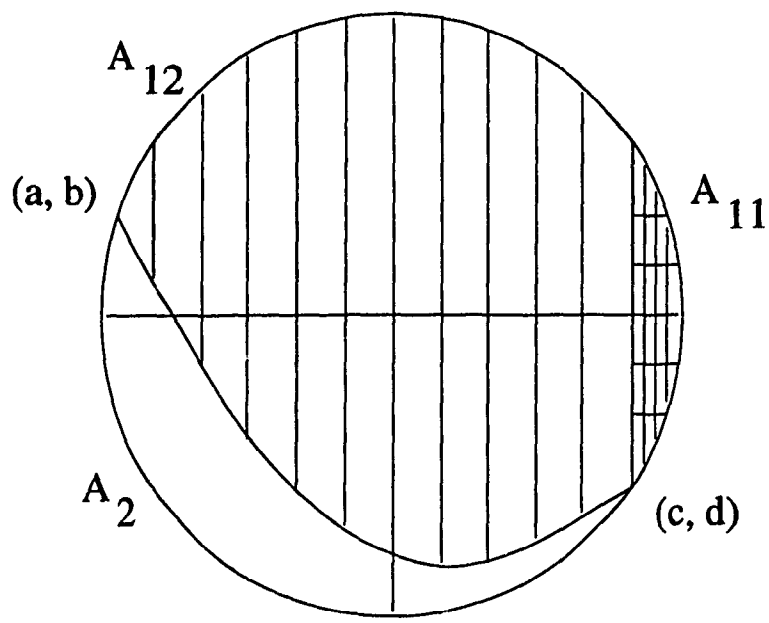


Figure A.4: Case 4

In this case again, both areas A_1 and A_2 require to be split into two regions in order to be computed. We choose to compute A_1 which we express as

$$A_1 = A_{11} + A_{12} \tag{A.11}$$

where A_{11} is given by the area bounded by the line $x_1 = c$ and the right half

$$x_1 = \sqrt{1 - (x_2 - c_2)^2} + c_1$$

of the circle C, between the points (c, d) and $(c, \sqrt{1-(c-c_1)^2}+c_2)$ and A_{12} is given by the area bounded by the regions:

$$\begin{aligned} x_1 &= c; \\ x_2 &= \sqrt{1-(x_1-c_1)^2}+c_2 \\ x_2 &= \sin((\pi/2)x_1) \end{aligned} \tag{A.12}$$

Hence

$$A_{11} = \int_d^{\sqrt{1-(c-c_1)^2}+c_2} \left[(\sqrt{1-(x_2-c_2)^2}+c_1) - (c) \right] dx_2$$

Integrating this expression gives

$$\begin{aligned} A_{11} &= (c_1-c)\left[\sqrt{1-(c-c_1)^2}+c_2-d\right] + \\ &\quad \frac{1}{2}\left[\arcsin\left(\sqrt{1-(c-c_1)^2}\right) - \arcsin(d-c_2)\right] + \\ &\quad \frac{1}{2}\left[(c-c_1)\sqrt{1-(c-c_1)^2} - (d-c_2)\sqrt{1-(d-c_2)^2}\right] \end{aligned} \tag{A.13}$$

Further, we have

$$A_{12} = \int_c^a \left[(\sin((\pi/2)x_1)) - \left(\sqrt{1-(x_1-c_1)^2}+c_2\right) \right] dx_1$$

Integrating, we obtain

$$\begin{aligned} A_{12} &= -\frac{1}{2}[\arcsin(a-c_1) - \arcsin(c-c_1)] - \\ &\quad \frac{1}{2}\left[(a-c_1)\sqrt{1-(a-c_1)^2} - (c-c_1)\sqrt{1-(c-c_1)^2}\right] - \\ &\quad c_2(a-c) - \frac{2}{\pi}[\cos((\pi/2)a) - \cos((\pi/2)c)] \end{aligned} \tag{A.14}$$

As in the previous case, substituting equations A.13 and A.14 into equation A.11, we can get A_1. Writing the regression entirely in term of A_1 from equations A.1 and A.2, we get

$$\mathrm{E}[\mathrm{y} \mid \mathbf{x}] = \frac{0.9}{\pi}(2A_1 - \pi)$$

Substituting A_1 from the above calculations, into the above expression gives the regression for this case.

Bibliography

[1] A. R. Barron, "Approximation and Estimation bounds for Artificial Neural Networks", *Machine Learning*, 14, 115-133 (1994).

[2] T. M. Cover, "Estimation by the Nearest Neighbor Rule", *IEEE Transactions on Information Theory*, Vol. IT-14, No. 1, 50-55 (1968).

[3] T. M. Cover and P. E. Hart, "Nearest Neighbor Pattern Classification", *IEEE Transactions on Information Theory*, Vol. IT-13, No. 1, 21-27, (1967).

[4] K. B. Davis, "Mean Square Error Properties of Density Estimates", *The Annals of Statistics*, Vol. 3, No. 4, 1025-1030 (1975).

[5] L. P. Devroye, "The Uniform Convergence of Nearest Neighbor Regression Function Estimators and their Application in Optimization", *IEEE Transactions on Information theory*, Vol. IT-24, No. 2, 142-151 (1978).

[6] L. Devorye and G. L. Wise, "Consistency of a Recursive Function Estimate", *Journal of Multivariate Analysis*, Vol. 10, No. 4, 539-550 (1980).

[7] L. P. Devroye and T. J. Wagner, "Distribution- free consistency results in nonparametric discrimination and regression estimation", *The Annals of Statistics* Vol. 8, No. 2, 231-239 (1980).

[8] J. H. Friedman, "Multivariate Adaptive Regression Splines", *The Annals of Statistics*, Vol. 19, No. 1, 1-67 (1991)

[9] S. Geman, E. Bienenstock and R. Doursat, "Neural Networks and the Bias/Variance Dilemma", *Neural Computation*, 4, 1-58, (1992).

[10] L. Györfi, "The Rate of Convergence of k_n -NN Regression Estimates and Classification Rules", *IEEE Transactions on Information Theory*, Vol. IT-27, No. 3, 362-364, (1981).

[11] W. Härdle and J. S. Marron, "Optimal bandwidth Selection in Nonparametric Regression Function Estimation", Vol. 13, No. 4, 1465-1481 (1985).

[12] J. Hertz, A. Krogh and R. Palmer, "Introduction to the theory of Neural Computation", 1991, Addison-Wesley.

[13] A. K. Jain, R. C. Dubes, "Clustering Methods and Algorithms", Prentice Hall, 1988.

[14] A. Krzyzak, "Global Convergence of the Recursive Kernel Regression Estimates with Applications in Classification and Nonlinear System Estimation", *IEEE Transactions on Information Theory*, Vol. 38, No. 4, 1323-1337, (1992).

[15] A. Krzyzak, T. Linder and G. Lugosi, "Nonparametric Estimation using Radial Basis Function Nets and Empirical Risk Minimization", submitted to *IEEE Transactions on Neural Networks*, (1994).

[16] R. P. Lippmann, "An Introduction to Computing with Neural Nets", *IEEE ASSP Magazine*, 4-22, April 1987.

[17] Y. Mack, "Local properties of k-NN Regression Estimates", *Society for Industrial and Applied Mathematics*, Vol. 2, No. 3, 311-323 (1981).

[18] J. Moody and J. Darken, "Fast learning in networks of locally-tuned processing units", *Neural Computation* I 281-294 (1989).

[19] J. Park and I. W Sandberg, "Universal Approximation Using Radial Basis Function Networks", *Neural Computation*, 3, 246-257 (1991).

[20] J. Park and I. W. Sandberg, "Approximation and Radial Basis Function Networks", *Neural Computation*, 5, 305-316, (1993).

[21] T. Poggio and F. Girosi, " Networks for Approximation and Learning", *Proceedings of the IEEE*, Vol. 78, No. 9, (1990).

[22] R. Y. Rubinstein, "Simulation and the Monte Carlo Method", *Wiley Series in Probability and Mathematical Statistics*, John Wiley and Sons, 1981.

[23] C. J. Stone, " Consistent Nonparametric regression", *The Annals of Statistics*, Vol. 5, No. 4, 595-645 (1977).

[24] C. Stone, "Optimal global rates of convergence for nonparametric regression estimation", *Annals of Statistics*, Vol. 10, No. 4, 1040-1053 (1982).

[25] G. S. Watson and M. R. Leadbetter, "On the Estimation of the Probablility Density", *The Annals of Mathematical Statistics*, Vol. 34, 480-491 (1963).

[26] L.Xu, S. Klasa and A. Yuille, "Recent advances on techniques of static feedforward networks and supervised learning", *International Journal of Neural Systems*, Vol. 3, No. 3, 253-290, (1992).

[27] L. Xu, A. Krzyzak and E. Oja, "Rival penalized competitive learning for clustering analysis, RBF net and curve detection", *IEEE Transactions on Neural Networks*, Vol. 4, No. 4, 636-649 (1993).

[28] L. Xu, A. Krzyzak and Y. Yuille, "On Radial Basis Function Nets and Kernel Regression: Statistical Consistency, Convergence Rates and Receptive Field Size.", *Neural Networks*, Vol. 7, No. 4, 609-628, (1994).