11-1-1995

BAYES ERROR ESTIMATION USING LOCAL METRICS

Keinosuke Fukunaga  
Purdue University School of Electrical and Computer Engineering

Sze-po R. Hung  
Purdue University School of Electrical and Computer Engineering

Follow this and additional works at: http://docs.lib.purdue.edu/ecetr

http://docs.lib.purdue.edu/ecetr/147

This document has been made available through Purdue e-Pubs, a service of the Purdue University Libraries. Please contact epubs@purdue.edu for additional information.
BAYES ERROR ESTIMATION USING LOCAL METRICS

KEINOSUKE FUKUNAGA
SZE-PO R. HUNG

TR-ECE 95-26
NOVEMBER 1995

SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING
PURDUE UNIVERSITY
WEST LAFAYETTE, INDIANA 47907-1285
BAYES ERROR ESTIMATION
USING
LOCAL METRICS

Keinosuke Fukunaga
Sze-po R. Hung

School of Electrical Engineering
1285 Electrical Engineering Building
Purdue University
West Lafayette, IN 47907-1285

This work was in part supported by Sandia National Laboratories under Contract AE-8286.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>LIST OF TABLES</th>
<th>vi</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIST OF FIGURES</td>
<td>viii</td>
</tr>
<tr>
<td>ABSTRACT</td>
<td>xiii</td>
</tr>
</tbody>
</table>

1. INTRODUCTION                       | 1   |
| 1.1 Problem Statement and Motivation | 1   |
| 1.2 Thesis Organization              | 4   |

2. BAYES ERROR ESTIMATION USING LOCAL METRICS | 7   |

2.1 Introduction                      | 7   |

2.2 Parzen Classifier                 | 8   |
| 2.2.1 Parzen approach                | 8   |
| 2.2.2 Global metric                  | 10  |
| 2.2.3 Local metric - theoretical Hessian | 10  |
| 2.2.4 Local metric - estimated Hessian | 16  |
| 2.2.5 Modification of $\tilde{K}$      | 21  |
| 2.2.6 Common metric                   | 26  |

2.3 $K$-NN Classifier                 | 32  |
| 2.3.1 $K$-NN approach                | 32  |
| 2.3.2 Global metric                  | 35  |
| 2.3.3 Local metric - theoretical Hessian | 35  |
| 2.3.4 Local metric - estimated Hessian | 37  |
2.3.5 Common metric ................................. .40
2.4 Kernel-type \textbf{k-NN} and \textbf{Parzen} Classifiers ............... .40
2.5 Conclusion ....................................... .46

3. REFINEMENT OF THE LOCAL METRIC ESTIMATION ........ .49
3.1 Introduction ..................................... .49
3.2 Selection of $B$ ................................... .50
3.3 Number of Local Samples - $l_s$ and $l_p$ .................. .57
   3.3.1 Intrinsic dimensionality and number of local samples .. .57
   3.3.2 Sample size and number of local samples .............. .69
3.4 Sample Size Effect .................................. .76
3.5 Kernel Shape Effect .................................. .85
3.6 Conclusion ........................................... .87

4. PARAMETRIC ANALYSIS OF TWO-DIMENSIONAL RADAR IMAGES . .89
4.1 Introduction ........................................ .89
4.2 \textbf{Window-Shifting Distance Measurement Algorithm} ........ .91
4.3 Bhattacharyya Distance by Window-Shifting .................. .98
4.4 Outlier's Effect on The Bhattacharyya Distance - A Side Issue .105
4.5 Sample Correlation and Quadratic Classifiers ................. 107
4.6 Quadratic Classifier with Window-shifting .................... 120
4.7 Conclusion ........................................... 131

5. BAYES ERROR ESTIMATION OF THE ISAR DATA ............... 135
5.1. Introduction ........................................ 135
5.2 Estimation of Intrinsic Dimensionality ......................... 136
   5.2.1. Background study ............................... 136
   5.2.2. Intrinsic dimensionality estimation by window-shifting method .140
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.3</td>
<td>Sample Correlation and Local Metric B</td>
<td>149</td>
</tr>
<tr>
<td>5.4</td>
<td>Fast Computation Method of The Bayes Error Estimation Algorithm</td>
<td>154</td>
</tr>
<tr>
<td>5.5</td>
<td>Bayes Error Estimation of The ISAR Data</td>
<td>159</td>
</tr>
<tr>
<td>5.6</td>
<td>Conclusion</td>
<td>161</td>
</tr>
<tr>
<td>6.</td>
<td>SUMMARY AND FUTURE WORK</td>
<td>163</td>
</tr>
<tr>
<td>6.1</td>
<td>Summary of Contribution</td>
<td>163</td>
</tr>
<tr>
<td>6.2</td>
<td>Future Work</td>
<td>164</td>
</tr>
<tr>
<td></td>
<td>LIST OF REFERENCES</td>
<td>166</td>
</tr>
<tr>
<td>APPENDICES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Appendix A</td>
<td>The Data Sets Used in This Thesis</td>
<td>170</td>
</tr>
<tr>
<td>Appendix B</td>
<td>Derivation of The Local Estimation of $\frac{\nabla p(X)}{p(X)}$ and $\frac{\nabla^2 p(X)}{p(X)}$.</td>
<td>176</td>
</tr>
<tr>
<td>Appendix C</td>
<td>Derivation of The Trace Equalization Algorithm.</td>
<td>181</td>
</tr>
<tr>
<td>Appendix D</td>
<td>The Proof of Theorem 2.1</td>
<td>184</td>
</tr>
<tr>
<td>Appendix E</td>
<td>The Proof of Theorem 4.1</td>
<td>187</td>
</tr>
</tbody>
</table>
LIST OF TABLES

Table                  Page
3.1. Look-up table for Fig. 3.9. ............... .63
3.2. Look-up table for Fig. 3.10. ............... .65
3.3. Look-up table for Fig. 3.11. ............... .66
3.4. Look-up table for Fig. 3.12. ............... .67
3.5. Look-up table for Fig. 3.13. ............... .69
3.6. Look-up table for Fig. 3.14. ............... .72
3.7. Look-up table for Fig. 3.15. ............... .74
3.8. Look-up table for Fig. 3.16. ............... .75
4.1. The computational complexity of the window-shifting method for ISAR data. \( n=20 \times 32 \) .................... .97
4.2. The computational complexity of the window-shifting method for SAR data, \( n=64 \times 64 \) .................... .97
4.3. Outlier's effect on the Bhattacharyya distance for 8-dimensional I-I data. \( N=50 \) .................... .106
4.4. Outlier's effect on the Bhattacharyya distance for 8-dimensional I-41 data. \( N=50 \) .................... .107
4.5. Outlier's effect on the Bhattacharyya distance for 128-dimensional I-I data, \( N=200 \) .................... .108
4.6. The R and H errors of the correlated samples \( Y \)'s with dependent design and test sets. \( N=200 \) .................... .112
4.7. The R and H errors of the uncorrelated samples \( Z \)'s, \( N=200 \) .................... .113
4.8. Comparison of R and H errors with correlated and uncorrelated samples for various dimensions of ISAR data. .................... .120
<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.9. Quadratic classification errors for the Radar-32 data with window-shifting, N=400.</td>
<td>123</td>
</tr>
<tr>
<td>4.10. Quadratic classification errors for the Radar-64 data with window-shifting, N=400.</td>
<td>123</td>
</tr>
<tr>
<td>4.11. Quadratic classification errors for the ISAR data, n=5×8, N=246.</td>
<td>124</td>
</tr>
<tr>
<td>4.12. Quadratic classification errors for the ISAR data, n=10×16, N=246.</td>
<td>124</td>
</tr>
<tr>
<td>4.13. Quadratic classification errors for the ISAR data, n=20×32, N=246.</td>
<td>124</td>
</tr>
<tr>
<td>5.1. Metric effect on the intrinsic dimension of the Radar-64 data, N=400.</td>
<td>138</td>
</tr>
<tr>
<td>5.2. The summary of experimental results of case 1: 1&lt;N/n&lt;2.</td>
<td>149</td>
</tr>
<tr>
<td>5.3. The summary of experimental results of case 2: 0&lt;N/n&lt;1.</td>
<td>149</td>
</tr>
<tr>
<td>5.4. Bayes error estimation of I-I data, ε=10%, with correlated and uncorrelated samples for metric B, N=200.</td>
<td>152</td>
</tr>
<tr>
<td>5.5. Bayes error estimation of I-41 data, ε=9%, with correlated and uncorrelated samples for metric B, N=200.</td>
<td>152</td>
</tr>
<tr>
<td>5.6. Bayes error estimation of I-A data, ε=1.9%, with correlated and uncorrelated samples for metric B, N=200.</td>
<td>153</td>
</tr>
<tr>
<td>5.7. Bayes error estimation of Radar-32 data with correlated and uncorrelated samples for metric B, N=200, ε≥9%.</td>
<td>153</td>
</tr>
<tr>
<td>5.8. Bayes error estimation of Radar-64 data with correlated and uncorrelated samples for metric B, N=200, ε≥18%.</td>
<td>153</td>
</tr>
<tr>
<td>5.9. Performance comparison of local metric, global metric and quadratic classification for ISAR data, N=246.</td>
<td>160</td>
</tr>
<tr>
<td>A.1. The parameter list of the Gaussian pulse data set I.</td>
<td>172</td>
</tr>
<tr>
<td>A.2. The parameter list of the Gaussian pulse data set II.</td>
<td>173</td>
</tr>
</tbody>
</table>
**LIST OF FIGURES**

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Global metric $A_i = \Sigma_i$ applied to Parzen classifier using (a) $I$ data; (b) $I-41$ data; (c) $I-A$ data.</td>
<td>11</td>
</tr>
<tr>
<td>2.2</td>
<td>Trace equalization using theoretical Hessian (a) without metric normalization; (b) with metric normalization.</td>
<td>14</td>
</tr>
</tbody>
</table>
| 2.3    | Parzen classifier with complete expression for $K_i$ and $B_i=I$.
(a) without metric normalization; (b) with metric normalization. | 19   |
| 2.4    | Parzen classifier with $B_i=\Sigma_i$, $|A_i|=1$ and complete expression for $K_i$. | 20   |
| 2.5    | Parzen classifier with $\beta_i =$ median of eigenvalues, $|A_i|=1$ and $B_i=\Sigma_i$. | 23   |
| 2.6    | Parzen classifier with $\beta_i$ being "6p2n" and $|A_i|=1$. | 24   |
| 2.7    | Parzen classifier with $\beta_i=0$ ("8p0n") and $|A_i|=1$. | 25   |
| 2.8    | Parzen classifier with $\beta_i=-0.5\cdot$ (average of eigenvalues) and $|A_i|=1$. | 27   |
| 2.9    | Parzen classifier with $\beta_i=-1\cdot$ (average of eigenvalues) and $|A_i|=1$. | 28   |
| 2.10   | Parzen classifier with common metric and theoretical Hessian. | 30   |
| 2.11   | Parzen classifier with common metric and $\beta_i=0$, (a) $a_i = \frac{n+4}{(n+2)p_i^4}$
(b) $\alpha_i = \frac{1}{\mu_i^2}$. | 31   |
| 2.12   | Parzen classifier with common metric and $\beta_i=0$, (a) $a_i = \frac{1}{\mu_i}$
(b) $\alpha_1 = \alpha_2 = 1$. | 33   |
<p>| 2.13   | k-NN classifier with covariance as global metric. | 36   |
| 2.14   | k-NN classifier with theoretical Hessian. | 38   |</p>
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.15</td>
<td>k-NN classifier with estimated local metric, (a) $\beta_1 = 0$; (b) $\beta_1 = -0.5 \cdot \text{average}$ of eigenvalues.</td>
<td>.39</td>
</tr>
<tr>
<td>2.16</td>
<td>k-NN classifier with common metric and theoretical Hessian.</td>
<td>.41</td>
</tr>
<tr>
<td>2.17</td>
<td>k-NN classifier with common metric and estimated Hessian. (a) $\alpha_i = \frac{1}{\mu}, \beta_i = 0$; (b) $\alpha_1 = \alpha_2 = 1, \beta_i = 0$.</td>
<td>.42</td>
</tr>
<tr>
<td>2.18</td>
<td>Kernel-type k-NN classifier with Gaussian kernel ($m = 1$) and $\beta_i = 0$.</td>
<td>.45</td>
</tr>
<tr>
<td>2.19</td>
<td>Kernel-type k-NN with $\beta_i = 0$ and (a) $m = 2.5$. (b) $m = 25$.</td>
<td>.47</td>
</tr>
<tr>
<td>3.1</td>
<td>The distribution of the bimodal Gaussian data.</td>
<td>.51</td>
</tr>
<tr>
<td>3.2</td>
<td>The global covariance of the bimodal Gaussian data.</td>
<td>.51</td>
</tr>
<tr>
<td>3.3</td>
<td>Illustration of how to choose local samples to form $B$ for the Radar-32 data.</td>
<td>.53</td>
</tr>
<tr>
<td>3.4</td>
<td>Performance comparison of using (a). local covariance; (b) global covariance; as $B$ for building up local region $L(X)$. Radar-32 data are tested.</td>
<td>.54</td>
</tr>
<tr>
<td>3.5</td>
<td>Performance comparison of using (a). window covariance; (b). global covariance as $B$ for building up local region $L(X)$. Radar-64 data are tested.</td>
<td>.55</td>
</tr>
<tr>
<td>3.6</td>
<td>Selection of neighbors.</td>
<td>.58</td>
</tr>
<tr>
<td>3.7</td>
<td>Typical examples of incorrect selection of the number of local samples.</td>
<td>.61</td>
</tr>
<tr>
<td>3.8</td>
<td>Selection of the number of local samples.</td>
<td>.62</td>
</tr>
<tr>
<td>3.9</td>
<td>Study of $I_s$ and $I_p$ on the Radar-32 data with $N=200$ (first pass).</td>
<td>.63</td>
</tr>
<tr>
<td>3.10</td>
<td>Study of $I_s$ and $I_p$ on the Radar-32 data with $N=200$ (second pass).</td>
<td>.65</td>
</tr>
<tr>
<td>3.11</td>
<td>Study of $I_s$ and $I_p$ on the Radar-64 data with $N=200$ (first pass).</td>
<td>.66</td>
</tr>
<tr>
<td>3.12</td>
<td>Study of $I_s$ and $I_p$ on the Radar-64 data with $N=200$ (second pass).</td>
<td>.67</td>
</tr>
<tr>
<td>3.13</td>
<td>Study of $I_s$ and $I_p$ on the Gaussian pulse data with $N=100$ and $n=12$.</td>
<td>.68</td>
</tr>
<tr>
<td>3.14</td>
<td>Study of $I_s$ and $I_p$ on the Radar-32 data with $N=1600$.</td>
<td>.72</td>
</tr>
<tr>
<td>3.15</td>
<td>Study of $I_s$ and $I_p$ on the Radar-64 data with $N=1600$.</td>
<td>.73</td>
</tr>
<tr>
<td>3.16</td>
<td>Study of $I_s$ and $I_p$ on the Gaussian pulse data with $n=12$ and $N=800$.</td>
<td>.75</td>
</tr>
<tr>
<td>3.17</td>
<td>Sample size effect for the parametric classifier and the nonparametric classifier with global metrics.</td>
<td>.77</td>
</tr>
<tr>
<td>3.18</td>
<td>Sample size effect for the nonparametric classifier with local metrics.</td>
<td>.79</td>
</tr>
<tr>
<td>3.19</td>
<td>Sample size effect for the Linear data set.</td>
<td>.80</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>3.20.</td>
<td>Sample size effect for the Gaussian data set I with ( n=12 ).</td>
<td>.80</td>
</tr>
<tr>
<td>3.21.</td>
<td>Sample size effect for the Gaussian data set II with ( n=12 ).</td>
<td>.81</td>
</tr>
<tr>
<td>3.22.</td>
<td>Sample size effect for the Radar-32 data.</td>
<td>.82</td>
</tr>
<tr>
<td>3.23.</td>
<td>Sample size effect for the Radar-64 data.</td>
<td>.83</td>
</tr>
<tr>
<td>3.24.</td>
<td>An example of 2-D distribution of a data set without sample correlation.</td>
<td>.83</td>
</tr>
<tr>
<td>3.25.</td>
<td>An example of 2-D distribution of a data set with correlated samples.</td>
<td>.84</td>
</tr>
<tr>
<td>3.27.</td>
<td>Kernel shape effect of the Radar-64 data.</td>
<td>.86</td>
</tr>
<tr>
<td>4.1.</td>
<td>Window-shifting over 2-dimensional images.</td>
<td>.93</td>
</tr>
<tr>
<td>4.2.</td>
<td>The number of times each pixel of an 8x8 images involved in the distance computation with ( w=3 ).</td>
<td>.95</td>
</tr>
<tr>
<td>4.3.</td>
<td>Estimated Bhattacharyya distance vs. ( 1/N ) for (a) Radar-32 data; (b) Radar-64 data.</td>
<td>.99</td>
</tr>
<tr>
<td>4.4.</td>
<td>Effect of window-shifting on the Bhattacharyya distance for (a) Radar-32 data; (b) Radar-64 data.</td>
<td>101</td>
</tr>
<tr>
<td>4.5.</td>
<td>Estimation of the Bhattacharyya distance by window-shifting and extrapolation for (a) Radar-32 data; (b) Radar-64 data.</td>
<td>102</td>
</tr>
<tr>
<td>4.6.</td>
<td>Estimation of the Bhattacharyya distance by window-shifting for ISAR data.</td>
<td>104</td>
</tr>
<tr>
<td>4.7.</td>
<td>The distribution of ( X, Y ) and ( Z ) with various ( c ).</td>
<td>110</td>
</tr>
<tr>
<td>4.8.</td>
<td>Distribution of the design and test sets for correlated samples.</td>
<td>111</td>
</tr>
<tr>
<td>4.9.</td>
<td>Quadratic errors of the correlated I-I data with independent design and test sets.</td>
<td>114</td>
</tr>
<tr>
<td>4.10.</td>
<td>Quadratic errors of the correlated I-I data with independent design and test sets.</td>
<td>114</td>
</tr>
<tr>
<td>4.11.</td>
<td>Quadratic errors of the correlated I-A data with independent design and test sets.</td>
<td>115</td>
</tr>
<tr>
<td>4.12.</td>
<td>Sample size vs. correlation index for correlated I-I data.</td>
<td>115</td>
</tr>
<tr>
<td>4.13.</td>
<td>Sample size vs. correlation index for correlated I-4I data.</td>
<td>116</td>
</tr>
<tr>
<td>4.14.</td>
<td>Sample size vs. correlation index for correlated I-A data.</td>
<td>116</td>
</tr>
<tr>
<td>4.15.</td>
<td>Quadratic classifier errors for (a) Radar-32 data; (b) Radar-64 data, using correlated and uncorrelated samples.</td>
<td>119</td>
</tr>
</tbody>
</table>
Figure

<table>
<thead>
<tr>
<th>Number</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.16</td>
<td>(a)-(d). The d-display of the 50-dimensional I-I data with various sample size.</td>
<td>128</td>
</tr>
<tr>
<td>4.16</td>
<td>(e)-(g). The d-display of the 50-dimensional I-I data with various sample size.</td>
<td>129</td>
</tr>
<tr>
<td>4.17</td>
<td>The d-display of the ISAR data with $n=20 \times 32$. (a) R method; (b) H method.</td>
<td>131</td>
</tr>
<tr>
<td>5.1</td>
<td>Sample size effect on the estimated intrinsic dimension of the Radar-32 data.</td>
<td>140</td>
</tr>
<tr>
<td>5.2</td>
<td>Sample size effect on the estimated intrinsic dimension of the Radar-64 data.</td>
<td>142</td>
</tr>
<tr>
<td>5.3</td>
<td>Sample size effect on the estimated intrinsic dimension of the ISAR data, $n=5 \times 8$.</td>
<td>142</td>
</tr>
<tr>
<td>5.4</td>
<td>Estimation of intrinsic dimension of the Radar-32 data by window-shifting method.</td>
<td>144</td>
</tr>
<tr>
<td>5.5</td>
<td>Estimation of intrinsic dimension of the Radar-64 data by window-shifting method.</td>
<td>145</td>
</tr>
<tr>
<td>5.6</td>
<td>Estimation of intrinsic dimension of the ISAR data, $n=5 \times 8$, by window-shifting method.</td>
<td>146</td>
</tr>
<tr>
<td>5.7</td>
<td>Estimation of intrinsic dimension of the ISAR data, $n=10 \times 16$, by window-shifting method with $N/n=1.54$.</td>
<td>147</td>
</tr>
<tr>
<td>5.8</td>
<td>Estimation of intrinsic dimension of the ISAR data, $n=20 \times 32$, by window-shifting method with $N/n=0.38$.</td>
<td>148</td>
</tr>
<tr>
<td>5.9</td>
<td>How to obtain $B$ from correlated and uncorrelated samples.</td>
<td>150</td>
</tr>
</tbody>
</table>
ABSTRACT

Bayes error estimation has long been an interesting topic in the pattern recognition area. For data with small dimensionality and large sample size, it is easy to estimate the upper and lower bounds of the Bayes error using global metrics in nonparametric classifiers. However, for the SAR (Synthetic Aperture Radar) data, the data dimensionality is very large ($n=64 \times 64 = 4,096$) and the sample size is very small ($N=1,000$). No conventional methods are able to estimate the Bayes error of the SAR data. Our solution to this problem is to fully use the property that the intrinsic dimensionality of the SAR data is much smaller than its data dimensionality. Another important issue related to the SAR data is the design of parametric classifiers. For such large $n$ and small $N$, computing the inverse of the covariance matrix is very difficult and time-consuming. Many classifier designers have to approximate the covariance matrix by a diagonal one. This motivates us to figure out a better way of designing a quadratic classifier which can give a much better performance and run in real time.

In this thesis, we have derived an algorithm to estimate local metrics. This algorithm has been tested on many artificial and real data sets and has been proven to work very well. We have also developed a new way, called "window-shifting", for measuring distances between samples. This method can boost the performance of quadratic classifiers over the current ones, and it is very suitable for real-time applications.
1.1 Problem Statement And Motivation

The Bayes error estimation is an important subject in pattern recognition. The term "Bayes error" corresponds to the "minimal probability of classification error" for any two given distributions. The purpose of estimating the Bayes error is to set a performance measure for classifier designs. For example, if the Bayes error between two objects is 10% and the classifier designed for these two objects generates an error of 20%, we know that the classification error is too high compared with the potentially achievable error and something can still be done to improve the performance of the classifier. On the other hand, if the classifier already achieves a classification error of 11%, we know that this error is very close to its potential and it is not worthwhile to try to improve its performance.

The probability distribution of a class is represented by a certain number of samples. If the sample size $N$ is large enough with respect to the data dimension $n$, the probability distribution of the two objects may be well characterized by these samples. If $n$ is very small (say $n<10$) and $N$ is large (say $N=1000$), the Bayes error can be estimated using well developed nonparametric techniques. However, when the dimensionality $n$ is very large, more sophisticated algorithms are required to estimate the Bayes error of these high dimensional data. A large number of samples may be required in order to characterize such a distribution, and this becomes impossible in practice. Thus, no conventional tools are complicated enough to handle them. Our goal is to develop a powerful and robust tool to solve this problem.
The intrinsic dimensionality $n_e$ is usually much lower than the data dimension $n$ in real-world applications. This means that data samples are contained in the $n_e$-dimensional subspace, and all analysis can be performed on this lower-dimensional subspace. For example, the SAR (Synthetic Aperture Radar) data with 64x64 pixels have an extremely high dimension, $n = 64 \times 64 = 4096$. However, the intrinsic dimensionality may be as low as several hundred, say 200, which is supposed to be determined by the number of scattering centers of the object. It is reasonable to model the data as the result of a nonlinear mapping from the 200-dimensional "intrinsic space" onto the 4096-dimensional "sample space". Each sample has its own intrinsic space. If we can find out the intrinsic space in which each sample is located, we can do all the computation on that space. The advantage of working on the intrinsic space is obvious: instead of manipulating matrices as large as 4096x4096, we can perform all matrix operations of size 200x200. Not only will the computation time and memory requirement be significantly reduced, but also the required sample size $N$ is reduced and estimation accuracy can be greatly enhanced. For example, computing the inverse of a matrix of size 4096x4096 is a very difficult issue particularly when the matrix is highly singular, and the computation result is also unreliable. These difficulties can be removed when the matrix size is reduced to 200x200.

As mentioned above, if the data dimension is high, the sample size required in order to characterize the probability distribution is an astronomical number. In practice, the number of available samples is very limited. For the SAR data, the number of samples available for each target is about 1,000. These samples contain 360 degrees of viewing angles and three different depression angles. Although it may not be impossible, it is, at least, extremely difficult to accurately characterize the probability distribution of a given object with merely 1,000 samples in the 4096-dimensional space. Therefore, it is even more difficult to determine the Bayes error with these constraints. However, the degree of difficulty can be greatly relieved if the dimensionality can be reduced to 200.

Our idea for solving the problems mentioned above is to go into the intrinsic space. This means that we have to develop an algorithm to identify the local samples around each test sample. It is essential to use a properly selected local metric to measure distances
between neighboring samples. By mapping the test samples and their neighbors from the sample space down to the intrinsic space, we can measure the distances among them and thus compute the probability density to obtain the discriminant function. This procedure is similar to the local feature extraction. The conventional global feature extraction does not work because it maps the samples from the sample space down to the common feature space. But in fact, such a "common feature space" is grossly different from the optimal local space for each test sample.

Local metric estimation is a brand new area of research. So far we have not seen any references concentrating on solving this problem. Our first step is to develop an algorithm for estimating local metrics. Applying it to the data sets with known Bayes errors, we can learn more about the properties of local metric estimation and make necessary modifications to the algorithm to make it work better. The intermediate steps are to apply this algorithm to the real data that have been carefully studied, and to refine the whole procedure. The final step is the determination of the Bayes error of the SAR and ISAR (Inverse Synthetic Aperture Radar) data, depending on the availability of these data sets. Since the data dimension is so high and the number of samples available is so limited, there have been many other unexpected difficulties waiting to be solved.

Another important issue often encountered in real applications is the estimation of the quadratic error between two classes. For large dimensional data, as we have mentioned before, it is extremely difficult to compute the inverse of a covariance matrix. Since the distance between the sample and class mean has to be normalized by the covariance matrix, we face great computational difficulty if the full covariance matrix has to be used. In many real-time applications, quadratic classifiers are often used to classify a detected object. Therefore, the computation speed is an important concern, too. It would be too time-consuming to compute distances using full covariance matrices. Because of the insufficient number of samples, the quadratic classifier designed based on those samples performs very poorly. We have to develop a new way of designing quadratic classifiers in order to solve these problems.

As mentioned above, the SAR data have the covariance matrix of size $4096 \times 4096$. 
Taking the inverse of such a matrix is also very difficult. To alleviate this difficulty, it is common to approximate the covariance matrix by setting all the off-diagonal terms to zero. Although taking the inverse is easy with such an approximation, a lot of important classification information is lost and the performance is far from good. We would like to develop a new way of measuring distance, which involves the off-diagonal terms to boost the performance of the quadratic classifier for the SAR data. We would like to develop an algorithm which can perform better than the existing one and can avoid the problems of using the full covariance. Furthermore, this method must be fast enough to be used in real-time applications. This problem is closely related to the problem of estimating the Bayes error; therefore, we would like to investigate this problem thoroughly in this thesis to make our research complete.

In this thesis, our goal has been set to solve the problems of the SAR data. However, the SAR data are not available at the press time of this thesis. Therefore, we have to test our algorithms on the ISAR data, which are provided by Lincoln Laboratory, Massachusetts Institute of Technology. The ISAR data are collected using the motion of objects, while the SAR data depend on the motion of the radar. Basically, the SAR and ISAR data are identical. We believe that by solving the smaller dimensional ISAR data with $n=20\times32$ we can acquire the necessary knowledge to solve the higher dimensional SAR data.

1.2. Thesis Organization

Chapter 1 is an introduction to the problems that we encounter and how we are going to solve them. In Chapter 2, we derive the expressions for estimating the second order derivative of a probability density and the local metric. These methods are developed and tested on the 8-dimensional artificial data sets to get insight into the solution to solve our problem. Chapter 3 is devoted to the refinement of the algorithm we derived in Chapter 2. We also discuss some issues related to the behavior of local metrics.
The window-shifting method is introduced in Chapter 4. This method is very useful in the quadratic classifier design, Bhattacharyya distance estimation and intrinsic dimensionality estimation. The Bayes error of the ISAR data is estimated in Chapter 5. In Chapter 6, we summarize the work we have done in this thesis and propose some work that deserves more study in the future.
CHAPTER 2
BAYES ERROR ESTIMATION USING LOCAL METRICS

2.1. Introduction

In many applications it is often encountered that the dimension of data is so high that it may be very difficult to estimate the Bayes error with global metrics such as covariances. However, if the intrinsic dimensionality is much lower than the data dimensionality, it could be advantageous to use local metrics for estimating the Bayes error. For example, the 2-dimensional Synthetic Aperture Radar (SAR) signal has extremely high dimensionality, \( n = 64 \times 64 = 4096 \); whereas, the intrinsic dimensionality \( n_c \) is only about 200, which is supposed to be determined by the number of scattering centers of the target. It is very difficult to compute the inverse of a large matrix such as 4096x4096. The obvious advantage of employing local metrics is that the data are handled in a much lower-dimensional space. We believe that employing local metrics in Bayes error estimation is a very important step toward solving the problem mentioned above. In this part of the thesis, we propose a procedure for estimating local metrics for each test sample using only a small number of neighboring samples.

The kernel-type density estimate was introduced by Rosenblatt [1] and studied and extended to multidimensional case by Parzen [2] and Cacoullos [3]. Whittle [41] also generalized and developed the Parzen kernel class of estimators. The k nearest neighbor (k-NN) density estimate was introduced by Fix and Hodges [4],[5] and was demonstrated for error estimation by Cover and Hart [6]. Significant amount of work has been done to improve the k-NN procedure [7]-[11]. These two classifiers have become the major
nonparametric classifiers since then. In this chapter, we use these two procedures to perform nonparametric classification and to demonstrate the feasibility of using local metrics for Bayes error estimation. The Bayes error is estimated by the resubstitution (R) error as a lower bound and the leave-one-out (L) error as an upper bound. Fukunaga and Hummels [12] pointed out that by threshold adjustment the estimation bias can be reduced. Our experiments thus adopt their option 3 [12] to compute the R and L errors. In this procedure, the optimal threshold is found by minimizing the R error, and that threshold is used for computing the L error.

Section 2.2 of this paper shows how locally estimated metrics perform using the Parzen classifier. These results are compared with those of using global metrics and theoretical local metrics for data with Gaussian distribution. For performance comparison, we apply the same metrics to the k-NN classifier and the results are given in Section 2.3. In Section 2.4, we mention the similarity between the k-NN and Parzen classifiers as we attempt to modify the conventional k-NN procedure to boost performance by imposing a kernel function. Comparisons are made among several different types of kernel functions. Conclusion and discussion are stated in Section 2.5. Appendix A gives a detailed description of the data sets we used for experiment. In Appendix B, we demonstrate how local probability Hessians are estimated and approximated. The algorithm for trace equalization is given in Appendix C.

### 2.2. Parzen Classifier

#### 2.2.1. Parzen approach

In this section, we demonstrate how to apply locally estimated metrics to the Parzen classifier and perform Bayes error estimation. Let $p_i(X)$ be the probability density of the sample $X$ in class $i$, $i=1,2$. The Parzen density estimate with a symmetric general kernel can be defined as
\( p_i(X) = \frac{c_1}{N_i r^n |A_i|^{1/2}} \sum_{j=1}^{N_i} \exp \left\{ - \left[ c_2 \left( X_j^{(i)} - X \right)^T \left( r^2 A_i \right)^{-1} \left( X_j^{(i)} - X \right) \right]^m \right\} \) \hspace{1cm} (2.1)

where \( X_j^{(i)} \) is the \( j \)-th sample in class \( i \), \( i = 1, 2 \) and \( N_i \) is the number of samples of class \( i \).

The parameter \( m \) determines the rate at which the kernel function drops off. For \( m = 1 \), (2.1) reduces to a simple Gaussian kernel. As \( m \) gets large, the edge of the kernel becomes steeper, finally approaching a uniform (hyperelliptical) kernel. The \( n \) is the data dimensionality. It can be verified that for any value of \( m \), the covariance of (2.1) is always \( r^2 A_i \), where \( r \) is the metric size and \( A_i \) is the metric for class \( i \). The scaling constants \( c_1 \) and \( c_2 \) in (2.1) are

\[
c_1 = \frac{m \Gamma \left( \frac{n}{2} \right) \Gamma \left( \frac{n+2}{2m} \right)}{(n\pi)^{n/2} \Gamma(n/2+1)} \hspace{1cm} (2.2)
\]

and

\[
c_2 = \frac{\Gamma \left( \frac{n+2}{2m} \right)}{n \Gamma \left( \frac{n}{2m} \right)} \hspace{1cm} (2.3)
\]

where \( \Gamma(\cdot) \) is the gamma function. An extensive discussion of the statistical properties of the probability density estimate in (2.1) may be found in [13]. An intensive study of the behavior of kernel functions can be found in [39].

If local metrics are used, \( N_i \) is replaced by \( l_p \) where \( l_p \) is the number of local nearest neighbors used to estimate the probability density. The discriminant function is then defined as

\[
h(X) = -\ln \frac{\tilde{p}_1(X)}{\tilde{p}_2(X)}
\]

\[
= \frac{1}{2} \ln \frac{|A_1|}{|A_2|} - \ln \frac{1}{l_p} \sum_{j=1}^{l_p} \exp \left\{ - \left[ c_2 \left( X_j^{(1)} - X \right)^T \left( r^2 A_1 \right)^{-1} \left( X_j^{(1)} - X \right) \right]^m \right\}
\]

\[
- \ln \frac{l_p}{l_p} \sum_{j=1}^{l_p} \exp \left\{ - \left[ c_2 \left( X_j^{(2)} - X \right)^T \left( r^2 A_2 \right)^{-1} \left( X_j^{(2)} - X \right) \right]^m \right\} \hspace{1cm} (2.4)
\]
Throughout this thesis, we use Gaussian kernels unless otherwise mentioned.

2.2.2. Global metric

Before discussing local metrics, the estimated Bayes errors are shown here by using the global metric. Although the optimal global metric is not known, here we use the covariance matrix. The data tested here and throughout this chapter are described in Appendix A. For standard data sets (I-I, 1-41 and I-A), the true (theoretical) covariance is used. The I-I data set represents a case where two classes are separated by the mean vector, while the 1-41 data set is a case for covariance separation. The I-A data set is a combination of both cases. All three of these data sets are computer generated Gaussian random variables with \( n = 8 \). The theoretical Bayes errors are known. Using \( m = 1 \) and \( A_i = C_i \), \( i = 1, 2 \), where \( \Sigma_i \) is the global covariance of class \( i \), and plugging it into (2.4), we obtain the estimated errors vs. the kernel size \( r \) in Fig. 2.1. The horizontal lines in Fig. 2.1 indicate the Bayes error, which is known theoretically. The Bayes error is properly bounded by the R and L error curves from the lower and upper sides respectively.

2.2.3. Local metric - theoretical Hessian

The bias of the Parzen approach can be shown to have the following form [14]

\[
\text{Bias} = E\{\hat{\varepsilon}\} - \varepsilon^* \approx f_1 r^2 + f_2 r^4 + \frac{f_3}{r^a N}
\]

(2.5)

where \( f_1, f_2 \) and \( f_3 \) are positive constants determined by the underlying distributions; \( E\{\hat{\varepsilon}\} \) is the expected value of the estimated error, and \( \varepsilon^* \) is the Bayes error, and \( N = N_1 = N_2 \).

The first two terms of (2.5) come from the bias of the density estimate, while the third is from the variance. The variance term becomes extremely large for \( r < 1 \) and a
Fig. 2.1. Global metric $A_j = \Sigma_j$ applied to Parzen classifier using (a) I-I data; (b) I-41 data; (c) I-A data.
large n because of $1/r^n$. For example $1/r^n \approx 2 \times 10^{19}$ for $r = 0.5$ and $n = 64$. This can hardly be compensated by a small $f_3$ (as $10^4$) and a large N (as $10^4$). However, as $r$ increases beyond 1, this term disappears quickly and the first two terms dominate. Therefore it is futile to minimize $f_3$ and operate in the region $r < 1$. Instead, a sensible way to reduce the bias is to select $r > 1$ and minimize $f_1$ and $f_2$. In order to minimize $f_1$ and $f_2$, the following equality has to hold [14]:

$$\text{tr}(K_1 A_1) = \text{tr}(K_2 A_2)$$  \hspace{1cm} (2.6)

where

$$K_i = \frac{\nabla^2 p_i(X)}{p_i(X)} i=1,2.$$  \hspace{1cm} (2.7)

Now our task is to estimate $K_i$ and to find $A_i$, the metric for class $i$, to satisfy (2.6). This operation is called "trace equalization" in this thesis. If no restrictions are imposed on $A_i$, it is easy to satisfy (2.6) simply by setting $A_i = K_i^{-1}$. However, $A_i$ is a metric and thus must be positive definite. On the other hand, $K_i$ is not necessarily positive definite, and therefore $A_i$ cannot be $K_i^{-1}$. Let us investigate the Gaussian cases first to obtain a guideline of how to attack this problem. For a Gaussian $p_i(X)$,

$$K_i = \Sigma_i^{-1}(X - M_i)(X - M_i)^\top \Sigma_i^{-1} - \Sigma_i^{-1} i=1,2$$  \hspace{1cm} (2.8)

where $M_i$ and $\Sigma_i$ are the mean and the covariance of class $i$, respectively.

For example, $X = M_i$, then $K_i$ is equal to $-\Sigma_i^{-1}$ and thus all the eigenvalues of $K_i$ are negative. In general, all eigenvalues of (2.8), regardless of $X$, are not necessarily positive. In order to alleviate the difficulty of solving a matrix equation such as (2.6), let us decompose $K_i$ as

$$K_i = \sum_{j=1}^{q} \lambda_{ij} \phi_{ij} \phi_{ij}^\top i=1,2$$  \hspace{1cm} (2.9)

where $\lambda_{ij}$ and $\phi_{ij}$ are the eigenvalues and eigenvectors of $K_i$, respectively. Note that $K_i$ in (2.8) is symmetric; thus $\phi_{ij}$ are orthonormal. Furthermore, applying a whitening linear transformation,

$$Y = \Psi^\top (X - M_i)$$  \hspace{1cm} (2.10)
we can make the covariance matrix of $Y$ the identity matrix as

$$\Psi^T \Sigma \Psi = I$$  \hspace{1cm} (2.11)

Thus, (2.8) is converted to

$$Q_i = YY^T - I$$  \hspace{1cm} (2.12)

where $Q_i$ is the Hessian in the $Y$-space.

It is easy to obtain the eigenvalues of (2.12). One is $\lambda_{i1} = Y^T Y$, satisfying $(YY^T - I)Y = (Y^T Y - 1)Y$. The others are $\lambda_{ij} = -1$, $j=2, ..., n$, because $(YY^T - I)\bar{Y} = -\bar{Y}$, where $\bar{Y}$ is any vector perpendicular to $Y$, i.e., $\bar{Y}^T Y = 0$. Thus for n-dimensional Gaussian distributions, we have $n-1$ negative eigenvalues (-1) and one eigenvalue $Y^T Y - 1$, which could be positive or negative, depending on whether $Y^T Y > 1$ or $Y^T Y < 1$. That is, $\lambda_{i1} < 0$ for $X$ close to $M_i$, and $\lambda_{i1} > 0$ otherwise. One more point should be remembered here. In order to reduce the bias of the Bayes error estimate, we would like to satisfy the trace equalization of (2.6), but only on the decision boundary and the region surrounding it. In the area far from the decision boundary, the samples are classified properly and the selection of metric is not crucial. For example, on the Bayes boundaries of the I-I case, $y_1 \geq 1.28$ ($y_1$; the first component of $Y$), while in the 1-41 case, $Y^T Y \geq 13.2$. That is, in both cases, $\lambda_{i1} > 0$ for $i=1, 2$ on the Bayes boundaries. Therefore, there are 1 positive and (n-1) negative eigenvalues for $Q_i$, and thus for $K_i$. If we let $A_i$ have the form

$$A_i = \sum_{j=1}^{n} \frac{1}{|\lambda_{ij}|} \phi_j \phi_j^T$$  \hspace{1cm} (2.13)

then trace equalization is achieved and both sides of (2.6) become -(n-2). Note that $A_i$ is guaranteed to be positive definite in (2.13). The experimental results are shown in Fig. 2.2(a). In Fig. 2.2(a), it is observed that the biases cannot be removed for large $r$ just by trace equalization and threshold adjustment, particularly for the I-I data. This is somewhat expected because with large $r$ all $l_p$ neighbors almost equally contribute to make the classification decision, thus generating more bias.

During the experiments, it has been observed that, for large $r$, the metric sizes
Fig. 2.2. Trace equalization using theoretical Hessian (a) without metric normalization; (b) with metric normalization.
measured as $|r^2 A_i|$ become significantly different between class 1 ($\omega_1$) and class 2 ($\omega_2$). This results in extra samples being misclassified. Although there is not a theory to guide how the sizes of the metrics should be balanced, it is felt intuitively that the sizes of the metrics for $\omega_1$ and $\omega_2$ must be the same to measure the distances of a test sample to $\omega_1$ and $\omega_2$ neighboring samples. Therefore, we normalize the size of $A_i$ by making $|A_i| = 1$. We call this operation "metric normalization" throughout this thesis. The experimental results are shown in Fig. 2.2(b), which are better than the ones in Fig. 2.2(a).

In order to normalize $A_i$, the form of (2.13) is kept as it is, and a constant, $a_i$, is multiplied as

$$A_i = a_i \sum_{j=1}^{n} \frac{1}{|\lambda_j|} \phi_y \phi_j^T$$

such that

$$|A_i| = \frac{a_i^n}{\prod_{j=1}^{n} |\lambda_j|} = 1$$

(2.15)

This constant $a_i$ is determined by (2.15)

The improvement of Fig. 2.2(b) over Fig. 2.2(a) is not significant. However, when the Hessians, $K_n$, are estimated by a small number of local samples as will be studied later, the effect of metric normalization will become more evident. Here, it must be pointed out that the normalized $A_i$ no longer satisfies the trace equalization of (2.6). In order to satisfy both the trace equalization and the metric normalization, we need to introduce more constants, $a$'s, to modify each term or each group of terms of (2.14). We will discuss this when the estimated $K_n$ is introduced. We would like to emphasize at this point the importance of the mathematical form of (2.13) for $A_i$ and the metric normalization. Also, we would like to point out that the $r$'s used in Fig. 2.2(a) and (b) are different. That is, the size of the metric for a given $r$ is $|r^2 A_i| = r^{2n} |A_i|$ for Fig. 2.2(a), while $|r^2 A_i| = r^{2n}$ for Fig.2.2(b). Therefore, the direct comparison of these two sets of curves could be misleading. However, for the purpose of estimating the Bayes error, it is acceptable as long as the Bayes error is bounded properly regardless of $r$. 
2.2.4. Local metric - estimated Hessian

So far we have discussed how to determine a local metric for a Gaussian distribution. In practice, we cannot assume that a given data set is Gaussianly distributed. Therefore, we have to estimate the Hessian $K_i$. The Hessian involves the second order derivative, and it is very difficult to get an accurate estimate of the derivative for an unknown distribution. In addition, the estimation must be performed by a relatively small number of local samples, $I_S$. Therefore, our concern is not how accurate the estimate is, but how effective the estimate is on the Bayes error estimate.

In order to estimate the $\omega_i$ Hessian at $X$, the $I_S$ local neighboring samples around $X$ are selected by using a metric $B_i$, forming the local region $L_i(X)$. That is,

$$L_i(X) = \{ Y: (Y - X)^\top B_i^{-1} (Y - X) \leq \rho_i^2 (n+2) \}$$  \hspace{1cm} (2.16)

where $\rho_i$ is the normalized radius of $L_i$ and is measured after $I_S$ local samples are assembled. Then the estimate of $K_i$, $\hat{K}_i$, is obtained as

$$\hat{K}_i = \alpha_i (B_i^{-1} \hat{S}_i B_i^{-1} - \beta_i B_i^{-1})$$  \hspace{1cm} (2.17)

where

$$\hat{S}_i = \frac{1}{I_S} \sum_{k=1}^{I_S} (X_{k\in N}^{(i)} - X)(X_{k\in N}^{(i)} - X)^\top$$  \hspace{1cm} (2.18)

is the sample correlation matrix around the sample $X$, and

$$\alpha_i = \frac{n+4}{\rho_i^4 (n+2)}$$

and

$$\beta_i = \frac{\text{tr}(\hat{S}_i B_i^{-1}) + 2 \rho_i^2}{n+2}$$  \hspace{1cm} (2.19)

are scaling constants. In (2.18), $I_S$ is the number of nearest neighbors for building up the local sample autocorrelation matrix $\hat{S}_i$. The derivation of (2.17) is given in Appendix B.

Next, an algorithm must be established to determine $A$ from $\hat{K}_i$. As was discussed, the theoretical $K_i$ for a Gaussian distribution had either one or zero positive
eigenvalue, and all others were negative. That made the determination of $A_i$ simpler. However, for the more general case of non-Gaussian and the estimated $\hat{K}_i$, the number of positive eigenvalues could be any number between 0 and $n$, and the number for $\omega_1$ could be significantly different from the one for $\omega_2$. Thus we need a more complex algorithm to derive $A_i$ from $\hat{K}_i$.

Again let $\lambda_{ij}$ and $\phi_{ij}$ be the eigenvalues and eigenvectors of $\hat{K}_i$, and express $\hat{K}_i$ as

$$\hat{K}_i = \sum_{j=1}^{n} \lambda_{ij} \phi_{ij} \phi_{ij}^T$$  \hspace{1cm} i=1,2. \quad (2.20)$$

Then, set $A_i$ to be

$$A_i = \sum_{j=1}^{n} \frac{a_{ij}}{\lambda_{ij}} \phi_{ij} \phi_{ij}^T$$  \hspace{1cm} i=1,2, \quad (2.21)$$

where $a_{ij}$ is a positive constant and is determined to satisfy the trace equalization. With $a_{ij}/\lambda_{ij}$ as its j-th eigenvalue, $A_i$ is positive definite and forms a metric. Then the trace equalization, $\text{tr}(\hat{K}_1 A_1 - \hat{K}_2 A_2) = 0$, becomes

$$\text{tr}(\hat{K}_1 A_1 - \hat{K}_2 A_2) = \sum_{\lambda_{ij} > 0} a_{ij} - \sum_{\lambda_{ij} < 0} a_{ij} - \sum_{\lambda_{ij} > 0} a_{2ij} + \sum_{\lambda_{ij} < 0} a_{2ij} = 0 \quad (2.22)$$

There are many combinations of $a_{ij}$'s to satisfy (2.22). However, various experiments have hinted that it is not good to deviate from the form of $\sum_{\lambda_{ij}} \frac{1}{\lambda_{ij}} \phi_{ij} \phi_{ij}^T$. So we have adopted a procedure to minimize the deviation of $a_{ij}$ from 1 under the condition that the trace equalization is satisfied. That is, a criterion

$$J = \sum_{j=1}^{n} (a_{1j} - 1)^2 + \sum_{j=1}^{n} (a_{2j} - 1)^2 - \mu \left[ \sum_{\lambda_{ij} > 0} a_{1j} - \sum_{\lambda_{ij} < 0} a_{1j} - \sum_{\lambda_{ij} > 0} a_{2j} + \sum_{\lambda_{ij} < 0} a_{2j} \right] \quad (2.23)$$

is minimized, where $\mu$ is the Lagrangian Multiplier. Taking the derivative of $J$ with respect to $\mu$ and $a_{ij}$ and equating them to zero, we obtain

$$a_{1j} = 1 + \frac{\mu}{2} \quad \text{for} \quad \lambda_{1j} > 0$$
\begin{align*}
a_{ij} &= 1 - \frac{\mu}{2} \quad \text{for } \lambda_{ij} < 0 \\
a_{2j} &= 1 - \frac{\mu}{2} \quad \text{for } \lambda_{2j} > 0 \\
a_{ij} &= 1 + \frac{\mu}{2} \quad \text{for } \lambda_{ij} < 0
\end{align*}

and

\[
\frac{\mu}{2} = \frac{(n_{1p} - n_{1n} + n_{2p} - n_{2n})}{(n_{1p} + n_{1n} + n_{2p} + n_{2n})}
\]

where \(n_{ip}\) and \(n_{in}\) stand for the number of positive and negative eigenvalues of \(\hat{K}_1\), respectively. The derivation of (2.24) and (2.25) is given in Appendix C. The constants \(a_i\) can take on only one of two numbers, either \(1 + \mu/2\) or \(1 - \mu/2\), depending on \(i\) and the sign of \(\lambda_{ij}\). Also, note that \(\mu = 0\) when \(n_{1p} = n_{2p}\) (and subsequently \(n_{1n} = n_{2n}\)).

At this point, two more problems must be considered. One is metric normalization, and the other is how to determine \(B_i\), the metric used to select \(l_s\) local samples for estimating \(K_i\). In order to see the effects of these parameters, the experiments were run and the results are shown in Fig.'s 2.3 and 2.4. Note that the term "complete expression" in the caption indicates that we use (2.17)-(2.19) to estimate the local metric without any approximation or alteration.

These figures show that the procedure is not working well. This means that \(\hat{K}_i\) of (2.17) is not a good estimate of \(K_i\), since the same procedure with the theoretical \(K_i\) gives acceptable bounds of the Bayes error as in Fig. 2.2. In the next section, we will discuss various modifications of \(K_i\), which will lead to better results. Also, Fig.'s 2.3-2.4 demonstrate the importance of metric normalization as well as selection of \(B_i\). That is, the performances of Fig.2.3(b) with the metric normalization are better than the ones of Fig.2.3(a) without the metric normalization. Comparison of Fig.'s 2.3(b) and 2.4 suggests
Fig. 2.3. Parzen classifier with complete expression for $\hat{K}$ and $B_{i,j}$. (a) without metric normalization; (b) with metric normalization.
Fig. 2.4. Parzen classifier with $B_i = \sum_i |A_i| = 1$ and complete expression for $K_i$. 
that $B_i = \sum_i$ is a better metric than $B_i = I$, particularly for the 1-41 data. Although we still do not know the best $B$, we are going to use $B_i = \Sigma_i$ with metric normalization throughout this chapter unless otherwise mentioned.

2.2.5. Modification of $K_i$

Since it is very difficult to figure out how to improve $K_i$ by modifying the matrix parts in (2.17), our attempt has been concentrated to experimentally optimize the scalar $\beta_i$. Another scalar, $a_i$, is automatically determined when metric normalization, $|A_i| = 1$, is applied. For simplicity, let us select $B_i = I$. As seen in (B.37) and (B.40) of Appendix B, $\beta_i$ can be approximated by the average of the eigenvalues of $S_i$. Let $\beta_i = \mu_i = \frac{1}{n} \sum_{j=1}^{n} \mu_{ij}$, where $\mu_{ij}$ is the $j$-th eigenvalue of $S_i$. Thus,

$$\hat{K}_i = \alpha_i \sum_{j=1}^{n} (\mu_{ij} - \bar{\mu}_i) \xi_{ij} \xi_{ij}^T$$

(2.26)

where $\xi_{ij}$ are the eigenvectors of $S_i$. As seen in (2.26), the eigenvalues of $\hat{K}_i$, $\alpha_i (\mu_{ij} - \bar{\mu}_i)$, are positive or negative, depending on whether $\mu_{ij}$ is larger or smaller than average. This leads us to believe that controlling $\beta_i (= \bar{\mu}_i)$ may be interpreted as controlling the number of positive and negative eigenvalues of $\hat{K}_i$. This may be a better way to handle $\beta_i$ than the use of the original $\beta_i$ of (2.19).

For the general case where $B \neq I$, $B^{-1}(SB^{-1} - PI)$ and $(SB^{-1} - PI)$ have the same number of positive and negative eigenvalues. For notational simplicity, let $Q = B^{-1}$, and $R = (SB^{-1} - PI)$, then we have the following theorem.
Theorem 2.1:

Let $Q$ and $R$ both be symmetric $n \times n$ matrices, with $Q$ being positive definite. Then $QR$ and $R$ have the same number of positive and negative eigenvalues.

(Proof). See Appendix D.

In the previous section, $\rho_i$ was measured from $I_p$ local samples and was often observed as highly volatile. Besides, it is also observed in (2.24) that in our trace equalization algorithm the eigenvalues of the same sign are suppressed or amplified relative to those of the opposite sign because 2 a’s are used for scaling. In other words, an eigenvalue that bears important classification information may be reduced in magnitude relative to the others, leading to distance distortion, which causes an increase of the estimation bias. A feasible way to solve the above problems is to select $\beta_i$ to be the median of the eigenvalues of $S_i^A$, so that half of the eigenvalues of $K_i^A$ are positive and the other half are negative. The advantage of this selection of $\beta_i$ is that we do not have to use two parameters to scale the positive and negative eigenvalues. Instead, we can simply set $A_i$ to have the form of (2.13) and the trace equalization is automatically achieved because both sides are zero. Fig. 2.5 shows the results of selecting $\beta_i$ to be the median of the eigenvalues of $S_i^A B_i^{-1}$ with metric normalization. Also, $B_i = \Sigma_i$ is used for this experiment. Comparison of Fig.’s 2.4 and 2.5 shows that the performance is greatly improved.

A close observation of the eigenvalues of $S_i^A B_i^{-1}$ indicates that the median value is smaller than the mean value almost all the time, because the eigenvalues of $S_i^A B_i^{-1}$ are dominated by at most two or three larger ones. That is, lowering the value of $\beta_i$, the performance is improved from Fig. 2.4 (mean) to Fig. 2.5 (median). This suggests that we can lower the value of $\beta_i$ further. So, we started to change the number of positive eigenvalues, $n_p$, and the number of negative eigenvalues, $n_n$, and to maintain these numbers equal for both $\omega_1$ and $\omega_2$. We call the combination " $n_p$, $n_n$ " Fig.’s 2.6 and 2.7 show the performances for $6p2n$ and $8p0n$ for the Gaussian data set with $n=8$. 
Fig. 2.5. Parzen classifier with $\beta_i = \text{median of eigenvalues}$, $|A_i| = 1$ and $B_i = \Sigma_i$. 
Fig. 2.6. Parzen classifier with $\beta_i$ being "6p2n" and $|A_i| = 1$. 
Fig. 2.7. Parzen classifier with $\beta_i=0$ ("8p0n") and $|A_i|=1$. 
Comparing Fig.'s. 2.5-2.7, it may be concluded for $\beta_i$ that the smaller the better, although the difference is small between Fig.'s. 2.6 and 2.7.

Encouraged by the results, we even tried negative $\beta_i$'s. Fig.'s. 2.8 and 2.9 show the results for $\beta_i = -0.5 \tilde{\mu}_i$ and $\beta_i = -\tilde{\mu}_i$, respectively. These results are very similar to the ones of Fig. 2.7 with $\beta_i = 0$.

Based on these experiments, we have concluded that, although the choice of $\beta_i$ is likely to be data dependent, the optimum range for $\beta_i$ seems to be somewhere between 0 and -0.5$\tilde{\mu}_i$, and the performance is not much affected by $\beta_i$ around that range. If this is the case, $\beta_i = 0$ gives us a significant advantage in computational simplicity and efficiency.

With $\beta_i = 0$, we do not need to compute the eigenvalues of $S, B_i^{-1}$ and adjust the threshold, as seen in (2.26). $\overset{\wedge}{K}_i$ is positive definite, and may be directly used as $A_i^{-1}$ after the metric normalization of $|\overset{\wedge}{K}_i| = 1$ by adjusting $\alpha_i$.

### 2.2.6. Common metric

For the samples around the decision boundary, the distances to both $\phi_1$ and $\phi_2$ neighbors are almost the same in that area, it seems possible to use a common metric $A (=A_1 = A_2)$ which simplifies the estimation procedure. The samples far away from the decision boundary are less likely to be misclassified, even though the common metric $A$ may not carry the best classification features. Also, the common metric approach tends to preferred theoretically because there is no metric normalization involved. We thus want to investigate the feasibility of using the common metric.

If we let $A_1 = A_2 = A$, (2.6) becomes

$$\text{tr}(K_1 A) = \text{tr}(K_2 A)$$

or

$$\text{tr}((K_1 - K_2) A) = 0 \tag{2.27}$$
Fig. 2.8. Parzen classifier with $\beta = -0.5 \cdot$ (average of eigenvalues) and $|A_r| = 1$
Fig. 2.9. Parzen classifier with $\beta_i = -1 \bullet$ (average of eigenvalues) and $|A_i| = 1$. 
The trace equalization procedure for (2.27) is very similar to the one for different metrics in Section 2.2.4. By setting \( A = A_1 = A_2 \) (\( a_i = a_j = a_2 \)) in (2.23), we can solve for \( \mu \) and \( a_i \)’s. The detailed derivation is given in Appendix C.

First, we would like to see the performance of the common metric using the theoretical expression for \( K_0 \). The results are shown in Fig. 2.10. Comparison between Fig.’s 2.2(b) and 2.10 indicates that the common metric approach works pretty well, if not better. So, we decide to investigate this problem further.

When the estimated Hessian is used, a special attention is needed to address the problem of how to determine the scale. The estimated Hessian has the form of (2.17), where there are two scaling constants, \( \alpha_i \) and \( \beta_i \), involved. In the previous section, the constant \( \alpha_i \) was ignored, because it was automatically determined by the metric normalization, \( |A_i| = 1 \). However, \( \alpha_i \) can not be overlooked in the common metric approach. Since the common metric \( A \) is determined by \( K_1 - K_2 \), \( \alpha_i \)'s play a very important role here, because they act as weighting factors, which determine the percentage of influence from the \( \omega_1 \) and \( \omega_2 \) neighbors. Therefore, we have two parameters to be estimated to obtain the common metric, which could be very difficult to accomplish.

From previous experiments we know that the performance of the Parzen classifier stays almost the same in the interval between \( \beta_i = 0 \) and \( \beta_i = -0.5 \) (average of eigenvalues). For computational efficiency we prefer the choice of \( \beta_i = 0 \). With \( \beta_i = 0 \) and \( \alpha_i \) of (2.19), we obtain the results in Fig. 2.11(a).

The results in Fig. 2.11(a) are very poor, especially for I-I data. We conjecture that the choice of \( \alpha_i \) may have great influence on the estimated error rate. Since the estimation of \( \rho_i \) is very inaccurate due to the small sample size, we try to improve the estimation of \( \rho_i \) by associating it with the eigenvalues. (B.40) shows that \( \rho_i^2 \) can be approximated by the average of eigenvalues. Therefore,

\[
\alpha_i = \frac{n + 4}{(n + 2)\rho_i^2} \approx \frac{n + 4}{(n + 2)\mu_i^2} \approx \frac{1}{\mu_i^2}
\]  

(2.28)
Fig. 2.10. Parzen classifier with common metric and theoretical Hessian.
(a) \( a_i = \frac{n+4}{(n+2)\rho_i^4} \)

(b) \( a_i = \frac{1}{\rho_i^2} \)

Fig. 2.11. Parzen classifier with common metric and \( \beta_i = 0 \), (a) \( \alpha_i = \frac{n+4}{(n+2)\rho_i^4} \),

(b) \( \alpha_i = \frac{1}{\mu_i^2} \)
With $\alpha_i = 1/\mu_i^{-2}$ and $\beta_i = 0$, we obtain the results in Fig. 2.11(b), which are even worse than the ones in Fig. 2.11(a). Close observation of the experimental values shows that (2.28) is a good approximation of (2.17) with a deviation less than 10%. However, the choice of $\alpha_i$ in (2.28) tends to enlarge the difference between $\alpha_1$ and $\alpha_2$. Comparison of Fig. 2.11(a) and (b) indicates that performance worsens if the difference between $\alpha_1$ and $\alpha_2$ gets large. Intuitively, it is reasonable to try to make $\alpha_1$ stay close to $\alpha_2$. Thus, we select $\alpha_i = 1/\mu_i$. With this selection, $\alpha_1$ and $\alpha_2$ are pulled closer to each other. The results are shown in Fig. 2.12(a). We even tried to fix $\alpha_1 = \alpha_2 = 1$ as shown in Fig. 2.12(b). The performances in Fig. 2.12 are greatly improved over the ones in Fig. 2.11. However, this approach fails to bound the Bayes error of the I-A data no matter what value of $\alpha_i$ we select.

Although the common metric approach is preferred theoretically, it does not perform as well and robustly as the different metric approach. We conjecture that the major reason for this lies in the fact that there are two scaling parameters to be determined in the common metric approach, which makes it too fragile to handle two different kinds of data at the same time. From the above experiments we see that common metric approach can not handle data sets with very different covariances. For I-A data, the Euclidean distance is very different from the normalized distance, which makes it difficult to find a "common region" to select nearest $\omega_1$ and $\omega_2$ neighbors. However, this approach may work if the sample size is extremely large as is the case for the theoretical Hessian in Fig. 2.10, which may be interpreted as the asymptotic results for an infinite number of samples.

### 2.3 K-NN Classifier

#### 2.3.1 K-NN approach

In this section we will demonstrate the results of applying locally estimated metrics to the $k$-NN classifier. The conventional $k$-NN density estimate is defined as
(a) \( a_r = \frac{1}{\mu_j} \)

(b) \( \alpha_1 = \alpha_2 = 1 \)

Fig. 2.12. Parzen classifier with common metric and \( \beta_j = 0 \), (a) \( \alpha_i = \frac{1}{\mu_i} \) (b) \( \alpha_1 = \alpha_2 = 1 \).
where \( k \) means that up to the \((k-1)\)-th nearest neighbors are located inside the \( n \)-dimensional hypersphere centered at \( X \) with the \( k \)-th nearest neighbor on the boundary of the local region. The local region has a radius \( d^{(i)}_{kNN}(X) \) and volume \( v_i(X) \), where

\[
v_i(X) = \frac{\pi^{n/2}}{\Gamma(n/2 + 1)} d^{(i)}_{kNN}^n(X) A_i^{1/2}
\]

(2.30)

and \( d^{(i)}_{kNN}(X) \) is the distance between sample \( X \) and its \( k \)-th \( \omega_i \) nearest neighbor. Intensive discussions on the convergence properties of the nearest neighbor density estimate were done by Moore and Yackel \[15\], Devroye \[16\] and Loftsgaarden and Queensberry \[17\].

The discriminant function for the k-NN classifier is

\[
h(X) = -\ln \frac{\hat{p}_1(X)}{\hat{p}_2(X)} = \ln \frac{N_1}{N_2} + n \ln \frac{d^{(i)}_{kNN}}{d^{(2)}_{kNN}} + \frac{1}{2} \ln \left| A_1 \right|
\]

(2.31)

Since

\[
d^{(i)}_{kNN}(X) = (X^{(i)}_{kNN} - X)^T A_i^{-1} (X^{(i)}_{kNN} - X)
\]

(2.32)

we can rewrite (2.31) as

\[
h(X) = \ln \frac{N_1}{N_2} + \frac{n}{2} \ln \frac{(X^{(1)}_{kNN} - X)^T A_1^{-1} (X^{(1)}_{kNN} - X)}{(X^{(2)}_{kNN} - X)^T A_2^{-1} (X^{(2)}_{kNN} - X)} + \frac{1}{2} \ln \left| A_1 \right|
\]

\[
= \ln \frac{N_1}{N_2} + \frac{n}{2} \ln \frac{(X^{(1)}_{kNN} - X)^T A_1^{-1} (X^{(1)}_{kNN} - X)}{(X^{(2)}_{kNN} - X)^T A_2^{-1} (X^{(2)}_{kNN} - X)}
\]

(2.33)

The discriminant function in (2.33) uses normalized metrics, i.e. \( \frac{A_i}{\left| A_1 \right|^{1/n}} = 1 \), \( i=1,2 \).

Therefore the determinant term can be omitted. We see that the normalization does not affect the value of the discriminant function. This is also verified experimentally.
2.3.2. Global metric

Using the covariance as the global metric and applying the k-NN classifier to the standard data sets, we get the results shown in Fig. 2.13. Comparing Fig. 2.13 to Fig. 2.1, we find that the k-NN classifier seems to be inferior to the Parzen classifier for the data sets we test using an equal number of samples. There exists a positive bias as large as 2% for the 1-41 data. Comparison of Fig's 2.1 and 2.13 suggests two possibilities: 1). The Parzen classifier is superior to the k-NN classifier for small sample size problems; 2). The global covariance matrix is a better global metric for the Parzen classifier than for the k-NN one. We do not have theoretical proof for both of these. However, we find experimentally that kernel-type classifiers constantly perform better when the sample size is small. We will investigate this topic further in Section 2.4. We will see that the Parzen classifier offers more reliable results when estimated local metrics are applied. It can be shown [14] that the expected bias of the k-NN classifier has the following form

\[
\mathbb{E}\{\Delta e\} = b_1 \frac{1}{k} + b_2 \left(\frac{k}{N}\right)^{2/n} + b_3 \left(\frac{k}{N}\right)^{4/n}
\]  

(2.34)

where \(b_1, b_2\) and \(b_3\) are constants.

In order for (2.34) to approach 0, two conditions, \(k \to \infty\) and \(k/N \to \infty\), must be satisfied. However, for finite k, the first term of (2.34) does not become zero, and thus the k-NN may not be able to converge to the Bayes error. Fischer [44] indicates that the optimal k (so that the bias in (2.34) is minimized) should be a function of the dimensionality and the smoothness of the underlying distribution. Fukunaga and Hummels [40] have demonstrated that the k-NN estimates may be severely biased if the dimensionality of the data is large even for large sample sizes.

2.3.3. Local metric - theoretical Hessian

It can be shown [14] that in order to minimize the bias and the minimum mean square error of the discriminant function of the k-NN classifier, the following equality has
Fig. 2.13. k-NN classifier with covariance as global metric.
to be satisfied:

\[
\frac{1}{2} \text{tr}(K_1A_1)r^2(v_1p_1(X))^{-2/n} = \frac{1}{2} \text{tr}(K_2A_2)r^2(v_2p_2(X))^{-2/n}
\]  

(2.35)

where \(v_i\) is given in (2.30). By (2.29), we know that

\[v_ip_i(X) \equiv \frac{k-1}{N_i}
\]

(2.36)

If we select \(N_1 = N_2 = N\) and plug (2.36) into (2.35), we have \(\text{tr}(K_1A_1) = \text{tr}(K_2A_2)\) just the same as we have for the Parzen classifier. Thus, we can apply the same trace equalization algorithm (Appendix C) to the k-NN classifier.

Performing the same procedure as we did on the Parzen classifier, we first apply the theoretical Hessian to the standard data sets. The results are shown in Fig. 2.14. We find that local metrics overcome the difficulty we encounter for the I-41 data using global metrics. This is somewhat expected because the theoretical Hessian is equivalent to the condition that an infinite number of samples are used. Fig. 2.14 also verifies the argument that if there is an infinite number of samples available, the k-NN approach can converge to the Bayes error.

### 2.3.4. Local metric - estimated Hessian

From our experience studying the Parzen classifier, we know that the optimal \(\beta_i\) is somewhere between 0 and -0.5e(average of eigenvalues). In order to verify if this is also true for the k-NN classifier, we perform various experiments using many different values of \(\beta_i\). We find that the optimal \(\beta_i\) for the k-NN classifier is still in the same range as for the Parzen classifier. However, the performance of the k-NN classifier is not as good as that of the Parzen classifier, as is seen in Fig. 2.15. We find that no matter what \(\beta_i\) we choose, there is no way to properly bound the Bayes error for the I-41 data. Any other selection of \(\beta_i\) will degrade the performance of the I-41 or I-A data sets; thus, they are not shown here.
Fig. 2.14. k-NN classifier with theoretical Hessian.
(a). $\beta_i = 0$

(b). $\beta_i = -0.5 \cdot \text{(average of eigenvalues)}$

Fig. 2.15. k-NN classifier with estimated local metric, (a) $\beta_i = 0$; (b) $\beta_i = -0.5 \cdot \text{(average of eigenvalues)}$. 
2.3.5. Common metric

In this section the results of applying the common metric to the k-NN classifier are demonstrated. As before, we would like to investigate this issue first with the theoretical Hessian and then with the estimated one for performance comparison. If the theoretical Hessian (2.8) is used, we obtain the results in Fig. 2.16. Comparison of Fig.'s 2.10 and 2.16 shows that the common metric approach is not a good idea for the k-NN classifier even with the theoretical Hessian. The difference between the R and I, errors of the 1-41 data is simply too large, which makes the result useless, although the Bayes error is bounded. This approach also fails to bound the Bayes error of the I-A data.

Since even the theoretical Hessian does not work for the k-NN classifier, it is not surprising that the performance gets much worse when the estimated Hessian is used. Using the best empirical values for $\alpha_i$ and $\beta_i$ from our previous experiments on the Parzen classifier, we obtain the results in Fig. 2.17. Although the results are poor, it agrees with our previous experiments that $\alpha_1$ should be made close to $\alpha_2$ in order to reduce the estimation bias. Based on the experimental results we obtain, it can be concluded that the common metric approach is not a good method for the Bayes error estimation using the k-NN classifier.

2.4. Kernel-type k-NN and Parzen Classifiers

Comparison of Sections 2.2 and 2.3 convinces us that the performance of the k-NN classifier is not as good as that of the Parzen classifier when the sample size is small. Therefore, we try to improve the performance of the k-NN classifier by modifying its density estimate. The conventional k-NN uses the distance between a test sample and its k-th nearest neighbor, $d_{k\text{NN}}^{(i)}(X)$, $i=1,2$, for density estimation. For data with high dimensionality and under the condition that the number of available samples is small, those samples are sparsely distributed in the hyperspace. Thus, the $d_{k\text{NN}}^{(i)}(X)$ term alone may not be reliable enough to give a correct distance relationship in the discriminant function in
Fig. 2.16. k-NN classifier with common metric and theoretical Hessian.
Fig. 2.17. \(k\)-NN classifier with common metric and estimated Hessian, (a) \(\alpha_i = \frac{1}{\mu}, \beta_i = 0\); (b) \(\alpha_1 = \alpha_2 = 1, \beta_i = 0\).
We therefore impose a general symmetric kernel function centered at the test sample to put proper weighting on the j-th NN, \( j=1,2,\ldots,k \), and sum up those weights for the density estimate. We call this the "group decision". If all the j-th NN's, \( j \) from 1 to k, show some statistical tendency, this group decision is likely to be more accurate than the decision made by a single k-th NN. We define the kernel-type k-NN as

\[
p_i(X) = \frac{c_1}{|A_i|^{1/2}} \sum_{j=1}^{k} \exp \left[ -c_2 \left( (X_j - X)^T A_i^{-1} (X_j - X) \right)^{m} \right] \quad i=1,2, \quad (2.37)
\]

where \( c_1 \) and \( c_2 \) are weighting constants that are the same for both classes. The value of the constant \( c_1 \) does not have any influence on the discriminant function, since they will be canceled out by \( p_1(X)/p_2(X) \). On the other hand, the constant \( c_2 \) does play a role in the classifier performance. However, its effect is not crucial if it is chosen in such a way that the exponential function works in a proper region. Comparing (2.1) and (2.37), we find that there is a striking similarity between these two. For the Parzen classifier, we use a fixed \( l_p \) local samples for summation; whereas we use a varying number k samples in the kernel-type k-NN classifier. The metric size \( r \) in the Parzen classifier is a variable so that the R and L errors are observed with varying values of \( r \). In (2.37), the corresponding parameter \( c_2 \) is fixed or adaptive. We may assign \( c_2 \) to be a proper scaling constant that helps in narrowing the range between the R and L errors. On the other hand, we can make \( c_2 \) able to adjust itself so that it can work as a proper scaling factor. Our experiments show that if \( c_2 \) is too large, the range between the R and L will then become too large. However, if \( c_2 \) is too small, the R error may exceed the Bayes error, i.e., the estimated R error can not be used as the lower bound. We can make the kernel size adaptive by setting

\[
c_2 = \frac{c}{d_{kNN}^2} \quad (2.38)
\]

where \( c \) is a constant and

\[
d_{kNN}^2 = \frac{d_{kNN}^{(1)}^2 + d_{kNN}^{(2)}^2}{2} \quad (2.39)
\]

Note that both \( \omega_1 \) and \( \omega_2 \) have to go through the same scaling factor; otherwise,
the distance relationship will be distorted, causing a large bias in the estimated error rate. This selection will make an exponential function work in a proper region most of the time and reduce the estimation bias. We would like to point out that the selection of constant $c$ is rather insensitive. Unless $c$ is extremely small or large, the performance stays almost the same. If $m$ in (2.37) is set to 1, the kernel function in (2.37) becomes a Gaussian kernel. If $m$ goes to infinity, the kernel becomes a hyperelliptical one. In (2.29) the density is determined by the volume $v_i(x)$, which is determined by the distance from the sample $X$ to its $k$-th NN. The distance from the test sample to its first through the $(k-1)$-th NNs are irrelevant as long as they are located inside the hypersphere with volume $v_i(x)$. To see how (2.37) improves the performance of the k-NN classifier, we perform the following experiments.

Using local Hessian estimation and setting $\beta = 0$ and $c_2 = 0.05/d_{kNN}^2$, where 0.05 is empirically chosen, we get the results in Fig. 2.18, which are pretty good. The conventional k-NN fails to estimate the Bayes error of the 1-41 data as is shown in Fig. 2.15. However, the problem is solved simply by introducing a kernel function. We conjecture that in the hyperspace with such a small number of samples available, it is necessary to employ a proper kernel for density estimation. With proper weighting on the neighboring samples, the density can be estimated in a more reliable way. This means that although samples are distributed in a discrete way with large spacing among one another, the distribution can be made smoother by imposing a kernel function. That also implies that the performance of the Parzen classifier will be better than the k-NN classifier for small sample size.

Since the kernel function can improve performance, one may wonder if the kernel shape is an important factor in the overall performance. Watson and Leadbetter [42] and Epanechnikov [43] have proposed the optimum form of the kernel function for the Parzen estimators. However, for large dimensional data with small sample size, we would like to run experiments to see if the kernel function is data dependent. In order to investigate this issue, we perform experiments with different values of $m$ and the results are shown in Fig.
Fig. 2.18. Kernel-type k-NN classifier with Gaussian kernel \((m = 1)\) and \(\beta_i = 0\).
As the value of \( m \) increases from 2.5 to 25, the performance begins to drop. Our test data is Gaussianly distributed; therefore, the Gaussian kernel \((m=1)\) gives the best performance in the Bayes error estimation. However, for the real data with non-Gaussian distribution the Gaussian kernel may not always give the best results. In the next chapter, we will investigate this issue further using real radar data.

2.5. Conclusion

Estimation of local metrics is an essential step toward solving the problem of estimating the Bayes error of data sets with extremely high dimensionality. In this chapter we proposed an algorithm for estimating the second order derivative of a probability density with any distribution using a small number of local samples. The local metrics are obtained by the trace equalization procedure. There are many other ways to estimate local metrics as we have tested. However, they are either too complicated and unstable or too fragile and only good for some data sets. We find that the trace equalization procedure offers a robust and good performance for all the data sets we tested. In addition, we find that the estimated local metrics should be normalized to reduce the estimation bias, although the traces become not exactly equal. Our experimental results hint that the estimated Hessian implies the "form" of the local metric. However, it gives highly inaccurate information on the kernel size. In order to minimize this distortion, we thus perform metric normalization to reduce the estimation bias. Experiments confirm our conjecture.

Comparing the Parzen with the k-NN classifiers using the same local metrics estimation scheme, we find that the Parzen classifier performs better than the k-NN classifier. The price is paid by spending longer computation time because evaluation and summation of exponential functions are required for the Parzen classifier. We then tried to improve the conventional k-NN by imposing a kernel function to place proper weightings
Fig. 2.19. Kernel-type k-NN with $\beta_i = 0$ and (a) $m = 2.5$; (b) $m = 25$. 
on the nearest neighbors. A proper kernel function boosts the performance of the k-NN classifier so that it becomes as good as the Parzen one. We thus argue that kernel functions are essential for Bayes error estimation.

There are many factors governing the performance of local metrics. A very obvious, and perhaps the most important factor is the selection of \( B \), which is used to build up the local region around each test sample. In this chapter, we use the global metric as \( B \). However, it may not be a good choice for real data. Another important factor is the selection of the number of local samples. There are two kinds of local samples we talk about here. The first one is \( L_S \), which is the number of nearest neighbors used for Hessian estimation. The second one is \( L_p \), which is the number of nearest neighbors for density estimation. It is believed that the selection of \( L_S \) and \( L_p \) will definitely affect the overall performance. Another important issue is the sample size effect. Since we only use computer-generated samples in this chapter, this effect may not be clear to see. However, when high-dimensional real data with intriguing distributions are used, the sample size may have a large impact on the estimation of error rate. Larger sample size implies denser distribution of samples in the \( n \)-dimensional space. The estimated Hessian may be more accurate if the local area is more "local". The kernel shape may also play a role in the Bayes error estimation. Our experience is mainly from working with the Gaussian data, and the Gaussian kernel has proven to work pretty well for the Gaussian data. However, the real radar data may not be Gaussian and thus the Gaussian kernel may not be an optimal choice. All these topics have to be studied intensively in the next chapter.
CHAPTER 3
REFINEMENT OF THE LOCAL METRIC ESTIMATION

3.1 Introduction

In this chapter we discuss several important topics mentioned at the end of Chapter 2. Issues like selection of \( B \), number of local samples, etc., are very crucial to the performance of our algorithm for estimating the Bayes error using local metrics. In addition to the standard data sets that we used in the previous chapter, we improve our algorithm by testing it on the real radar signal data. These real data along with three artificial data sets are described in appendix A. By running experiments on the real and artificial data sets, we obtain the knowledge needed to solve the SAR data. We also learned from the previous chapter that the Parzen classifier is more reliable than the k-NN classifier; therefore, we decide to use the Parzen classifier for our further research. In Section 3.2, we propose several ways to select \( B \), which are proven to perform very well experimentally. The issue of selecting the number of local samples is discussed in Section 3.3. In Section 3.4 we explore the sample size effect using both artificial and real data. The effect of the kernel shape is then demonstrated in Section 3.5. Conclusions are stated in Section 3.6. After studying all the factors that affect the performance of our algorithm for estimating local metrics, we feel much more confident of solving our goal, the SAR data.
3.2. Selection of B

Our experiments show that the most important factor governing the performance of our algorithm is the selection of the local metric $B$, which is used to build up the local region $L(X)$ for the test sample $X$. It is also the most difficult parameter to determine in our algorithm for estimating local metrics. In the previous chapter we found that the global metric $C$ is better than the oversimplified approximation, the Euclidean metric $I$. We would like to know whether the global metric is always the best choice. In order to explore this issue, we designed the following bimodal Gaussian data set. The class 1 ($\omega_1$) has the probability density

$$p_1(X) = 0.5N(M_1,I) + 0.5N(M_3,I)$$

(3.1)

and class 2 ($\omega_2$) has the density

$$p_2(X) = 0.5N(M_2,I) + 0.5N(M_4,I)$$

(3.2)

where

$$M_i = \begin{bmatrix} 3.29(i-1) & 0 & \ldots & 0 \end{bmatrix}^T$$

(3.3)

and $N(M, C)$ stands for a Gaussian distribution with mean $M$ and covariance matrix $C$.

The Bayes error of this data set is 7.5%. Only the first entry of the sample vector offers useful classification information. Plotting this data set in the 2-dimensional domain as shown in Fig. 3.1, we get a clear view of how these samples are distributed. However, without knowing the distribution in advance, which is often the case for real data, we will end up using the global covariance as $B$. The global covariances of this data set are shown in Fig. 3.2. We would like to point out that $B$ has two important functions in our algorithm. First, $B$ determines how distance is defined for finding the nearest neighbors, namely,

$$d^2(X,Y) = (X - Y)^T B^{-1} (X - Y)$$

(3.4)

Second, the local metric $A$ is determined by

$$A = \alpha B^{-1} S B^{-1}$$

(3.5)
Fig. 3.1. The distribution of the bimodal Gaussian data.

Fig. 3.2. The global covariance of the bimodal Gaussian data.
Once the local sample autocorrelation matrix $S$ has been computed, it is reshaped by multiplying it by $B^{-1}$ in front and back. If $B$ is not properly chosen, $A$ will be misshaped and will thus cause extra bias to the estimated density.

One can easily see in Fig. 3.2 that if the global covariance is used, the normalized distance between the samples is distorted. The direction of this distortion is along the $x_1$-axis, which is also the only source that bears useful classification information. It is no wonder that, when the global covariance is used as $B$, the Bayes error is found to be around 35%. However, if we are smart enough to use $I$ as $B$, we can bound the 7.5% Bayes error successfully. This leads us to believe that if we can figure out the Gaussian cluster that each sample belongs to, we can estimate the Bayes error successfully. However, in real world applications, the data dimension is so high that it is almost impossible to visualize the distribution of the samples simply by plotting them. On the other hand, even if we find an effective technique, it is not sure that the clusters found are close to Gaussian. Furthermore, we may mix up the samples in the design and test sets. The consequence is that we can not guarantee that the resubstitution (R) procedure will generate a lower bound and the leave-one-out (L) method will give a proper upper bound for the Bayes error. Since our goal is to estimate the Bayes error rather than classification, we definitely do not want to come up with suspicious upper and lower bounds. Based on our experiments, we believe that the optimal $B$ is the local covariance (like cluster covariance). The problem left to be solved is to figure out how to find the local covariance for each sample.

Fortunately there is a clue for us to solve this problem. What we are interested in are radar signals. When the reflected signal is received, the viewing angle is also known at the same time. In real-time classification, the viewing angle information may not be available; nevertheless, our task is to estimate the Bayes error of the available data set in an off-line operation. Since the detected object can not have any abrupt change in space, its movement can be looked at as a gradual change with respect to time or viewing angle.

Our conjecture is that we have to depend on the viewing angle to find the local covariance to be used as $B$. We thus group a certain number of adjacent samples (with
similar viewing and elevation angles) and compute the covariance of these adjacent samples for each test sample. The number of samples used to form the metric $B$ is denoted as $I_B$. In order to see how this selection of $B$ improves the estimation of the Bayes error, we ran experiments on the real data. For the Radar-32 data (Appendix A), we used the $I_B/2$ samples preceding and succeeding the current test sample to form $B$ as shown in Fig. 3.3.

![Fig. 3.3. Illustration of how to choose local samples to form B for the Radar-32 data.](image)

The performance using the local covariance as $B$ is significantly better than the one employing the global covariance. Picking up 200 test samples from the Radar-32 data set for testing, we found that using the local covariance can reduce bias by as much as 5%. The same procedure is repeated ten times for different sets of 200 test samples to get the averaged result in Fig. 3.4. The result convinces us that we are on the right track. The comparison of the estimated error rates is made in Fig. 3.4. In Fig. 3.4 we use $I_B = 200$. The selection of the number $I_B$ is not very sensitive as long as it is not too small or not too large. We see that, in Fig. 3.4 (b), the R error estimated by using the global covariance as $B$ is about 12%, which is higher than the Bayes error, which is about 9%. This agrees with the conclusion we made for the bimodal example that the local covariance must be used as $B$; otherwise, the Bayes error will be overestimated.

There is one thing that we would like to point out. In the previous chapter, we used the theoretical covariance as $B$; therefore, we did not have to worry about the issue of whether this covariance was obtained by an R or L procedure. However, when we process the real data, there is no "theoretical covariance" available; thus, all the parameters
we need have to be estimated from samples. In order to obtain a lower bound for the Bayes error, we have to make sure that every step involves the test sample that we are dealing with. On the other hand, we have to exclude the test sample in every step to make sure we will get a upper bound for the Bayes error.

(a) $B = \text{local covariance}$

(b) $B = \text{global covariance}$

Fig. 3.4 Performance comparison of using (a). local covariance; (b). global covariance as $B$ for building up local region $L(X)$. Radar-32 data are tested.

Now let us take a look at another real data set, the Radar-64 data (Appendix A). This data set is divided into four windows according to the viewing and elevation angles. Based on our previous experience, we immediately think of using window covariance as the metric $B$. For this case, $I_B = 999$ for the L method and $I_B = 1000$ for the R method. The difference is the test sample itself. Picking up 400 test samples out of the 4000 available samples each time and repeating the procedure five times using different sets of test samples, we obtain the result in Fig. 3.5. The difference of the estimated L errors (upper bounds) between Fig. 3.5(a) and 3.5(b) is as large as about 5%. From these experiments we are convinced that the local covariance is indeed the optimal choice for the metric $B$.

One may wonder how to choose a proper $B$ if the data set bears no information like viewing angles. For the artificial data with a known distribution of the parameters, the local samples should be found on the parameter space. For example, if the parameters are Gaussianly distributed with the covariance $\Sigma_1$ and $\Sigma_2$, the normalized distance can be
computed accordingly to determine the nearest neighbors in the parameter space. However, if the distribution of the parameters is unknown, we can only depend on the information of the sample space to determine nearest neighbors. This is likely to be less reliable because neither the Euclidean metric nor the global covariance are the optimal metric. Nevertheless, many artificially generated data are designed to have unimodal Gaussian distributions, and thus the Euclidean metric I or the global metric $\Sigma$ may sometimes work pretty well without the painful search of nearest neighbors for each test sample. However, as we have pointed out in the previous bimodal example (Fig. 3.1), such a naive way of selecting $B$ may fail to find the Bayes error.

Theoretically, since we use $B$ to build up a local region $L(X)$ and try to find the metric $A$ that best characterizes the distribution of the samples in that area, it seems quite reasonable that we may have $A = B$. Plugging this equality back to (2.17) and setting $\beta_i = 0$, we have

$$A_i^{-1} = \alpha_i A_i^{-1} \hat{S}_i A_i^{-1}$$

which can be simplified to

$$A_i = \alpha_i \hat{S}_i$$

An apparent paradox of (3.7) is how to find $\hat{S}_i$ if we do not have a local metric $A_i$ in

---

Fig. 3.5 Performance comparison of using (a) window covariance; (b) global covariance, as $B$ for building up local region $L(X)$. Radar-64 data are tested.
advance. If we use any metric other than $A_i$ to obtain $\hat{A}_i$, we lose the theoretical justification for using (3.7) because the assumption is not satisfied. Also note that (3.7) is different from the case of $B = I$ where

$$A_i^{-1} = \alpha_i \hat{A}_i. \quad (3.8)$$

Experiments show that (3.8) performs much better than (3.7). The local autocorrelation matrix $\hat{S}_i$ is a positive definite and symmetric matrix; therefore, $\hat{A}_i$ has orthonormal eigenvectors, and so does $\hat{A}_i^{-1}$. Without loss of generality let us assume the first eigenvector of $\hat{S}_i$, $\phi_1$, is the dominant one (with the largest eigenvalue $\lambda_1$). However, this eigenvector $\phi_1$ becomes the least important for $\hat{S}_i$ because its corresponding eigenvalue $1/\lambda_1$ is the smallest among all the eigenvalues. In other words, we emphasize the feature which is in fact least important in discriminating two classes, while the feature that bears the most important classification information is suppressed. This explains why (3.7) does not work at all.

There is another possibility, the iteration procedure, for selecting the metric $B$. The iteration procedure, or the so called "learning procedure", is very popular in signal processing, neural network and artificial intelligence areas. The most difficult part in applying the iteration procedure is the lack of an "optimization criterion", which is used to modify the parameters during the iteration. In our case, the best optimization criterion may be the bias of the estimated error rate, which is not known until all the samples are processed. It is extremely difficult to modify each $B$ after all the work is done. Without setting up an optimization criterion, we tried to make an initial guess, say $B = I$, and plugged it into (2.17) and performed trace equalization to obtain $A$. Then we set $B$ to be the $A$ we have obtained and performed the same computation again. We found that the result was so poor that we had to give up this idea. Even if we could figure out a way to set up the optimization criterion and perform the iteration procedure, we will have to worry about many other new problems. First of all, we have to guarantee the iteration procedure does not mess up the R and L principles, which may be very hard to prove.
Second, the lengthy computation is not desirable, particularly when the dimension gets large. Moreover, new problems like learning rate and number of iteration will occur. Based on our experiments, we determined that iteration is not a feasible method for setting up metric B.

3.3. Number of Local Samples - $I_S$ and $I_p$

3.3.1 Intrinsic dimensionality and number of local samples

Another important factor controlling the performance of our Bayes error estimation algorithm is the number of local samples. Generally, the intrinsic dimensionality of the real data is smaller than its data dimensionality. This gives us an edge to work on the lower-dimensional intrinsic space rather than the sample space. If we choose the number of local samples to be $I$, we can map samples from the $n$-dimensional sample space down to the $I$-dimensional local space, given that $I < n$. Since our strategy for solving SAR data is to work on the much lower-dimensional intrinsic space, instead of the 4096-dimensional sample space, it is intuitive to link $I$ with $n_e$. Experiments confirm that the intrinsic dimension of the data is the most important factor for selecting the number of local samples. There are two kinds of local samples involved in our algorithm. The first one is $I_S$, which is the number of local samples used to compute local autocorrelation $\hat{S}$. The second one is $I_p$, which is the number of local samples used for probability density estimation.

If there are a large number of samples available so that the distribution of samples is dense enough, it is feasible to use a small number of nearest neighbors to find the local space in which the test sample is located. Since the intrinsic dimensionality of the data set is $n_e$, we need at least $I_S = I_p = n_e$ nearest neighbors so that the test sample can be mapped from the $n$-dimensional data space to the $n_e$-dimensional local space. Nevertheless, in practice there are only a limited number of samples available and these samples are sparsely distributed in the hypersphere. It is uncertain how many nearest neighbors are
adequate for determining the intrinsic space in which the test sample is located. Since this topic is too difficult to analyze theoretically, we would like to study it experimentally to get some insight into this problem.

In order to understand how the number of local samples affects the estimated error rate, let us look at Figure 3.6 for selection of neighbors for the test sample $X$.

For the R method, the 1st nearest neighbor of the test sample $X$ is just $X$ itself, and the 2nd and 3rd NNs are $X_1$ and $X_4$ respectively. However, if the L method is applied, the 1st through the 3rd NNs are $X_1$, $X_4$ and $X_5$. If the number of nearest samples, $I_S$, for building up the local region is very small, the local autocorrelation $S$ estimated by the R and L methods are quite different. It means we are going to see a large difference between the estimated distances using the R and L metrics. The large difference between the measured distances translates into a large difference between the estimated R and L errors. On the other hand, a large $I_S$ helps to narrow the difference between the R and L errors. However, a large $I_S$ also implies a loss of locality, which may cause extra error for both the R and L errors and may lead us to overestimate the Bayes error.

Now let us examine the other kind of local samples, $I_p$. If the number of nearest neighbors $I_p$ for estimating probability density is very small, the shifting of nearest neighbors will cause a large difference between the estimated R and L errors. On the
other hand, if $l_p$ gets large, such a shuffling in neighbors has minor impact on the estimated error rate; thus, the L error is only slightly higher than the R error. This effect of the number of local samples, $l_p$, can be analyzed as follows.

Without loss of generality, we assume that the test sample is from class 1. The discriminant function of the Parzen density estimation using a kernel function $K(\bullet)$ can be written as

$$h_R(X^{(1)}) = -\ln \frac{\hat{P}_{1R}(X^{(1)})}{\hat{P}_2(X^{(1)})} = -\ln \frac{\frac{1}{l_p} \sum_{j=1}^{l_p} k_1(X^{(1)} - X_{jNN}^{(1)})}{\frac{1}{l_p} \sum_{j=1}^{l_p} k_2(X^{(1)} - X_{jNN}^{(2)})} \quad \text{(R method)} \quad (3.9)$$

Note that for the R method, $X^{(1)} = X_{1NN}^{(1)}$. Let the optimal threshold for the R method be $t$. Therefore, if $h(X^{(1)}) < t$, this test sample is classified to class 1, and vice versa. If the L method is applied, the discriminant function becomes

$$h_L(X^{(1)}) = -\ln \frac{\hat{P}_{1L}(X^{(1)})}{\hat{P}_2(X^{(1)})} = -\ln \frac{\frac{1}{l_p - 1} \sum_{j=1}^{l_p-1} k_1(X^{(1)} - X_{jNN}^{(1)})}{\frac{1}{l_p} \sum_{j=1}^{l_p} k_2(X^{(1)} - X_{jNN}^{(2)})} \quad \text{(L method)} \quad (3.10)$$

Note that this time $X^{(1)} \neq X_{1NN}^{(1)}$. These NNs are shuffled by one. Furthermore, $\hat{P}_{1L}(X^{(1)})$ can be related to $\hat{P}_{1R}(X^{(1)})$ by

$$\hat{P}_{1L}(X^{(1)}) = \hat{P}_{1R}(X^{(1)}) + \frac{1}{l_p - 1} \left[ \hat{P}_{1R}(X^{(1)}) \cdot k_1(0) \right]$$

$$= \hat{P}_{1R}(X^{(1)}) + \Delta \hat{P}_1 \quad (3.11)$$

where

$$\Delta \hat{P}_1 = \frac{1}{l_p - 1} \left[ \hat{P}_{1R}(X^{(1)}) \cdot k_1(0) \right] \quad (3.12)$$

is the difference between $\hat{P}_{1L}(X^{(1)})$ and $\hat{P}_{1R}(X^{(1)})$, and it is negative because $\hat{P}_{1R}(X^{(1)}) < k(0)$. Note that the smaller $l_p$ is, the larger $|\Delta \hat{P}_1|$ becomes. The discriminant function of
the L method can be rewritten as

\[ h_L(X^{(i)}) = -\ln \frac{\hat{p}_{1L}(X^{(i)})}{\hat{p}_{2}(X^{(i)})} = -\ln \frac{\hat{p}_{1R}(X^{(i)}) + \Delta \hat{p}_1}{\hat{p}_{2}(X^{(i)})} \]  

(3.13)

Since we use the same threshold \( t \) to determine the L error, a smaller \( I_p \) translates into a larger difference between the estimated R and L errors.

In our algorithm, we first determine the local region \( L(X) \) and then compute the autocorrelation \( S \). The estimated metric \( \hat{A} \) is supposed to be optimal for the samples inside \( L(X) \). If we apply this metric \( A \) to the samples outside \( L(X) \), there is no guarantee that this metric \( A \) is still an optimal one. In other words, the following relationship should hold.

\[ I_S \geq I_p \]  

(3.14)

In order to have a better understanding of how \( I_S \) and \( I_p \) affect the estimated error rate, we show in Fig. 3.7 some typical situations of poor selection of \( I_S \) and \( I_p \) when processing the Radar-32 and the Radar-64 data.

Note that in Fig. 3.7 we assume the sample size is large enough for successful Bayes error estimation. Although the Bayes error is properly bounded in Fig. 3.7(a), the range between the R and L errors is too large. It does not offer useful information on the Bayes error. On the other hand, if both \( I_S \) and \( I_p \) are too large, as shown in Fig. 3.7(b), the property of locality is lost, and the estimated error will be larger than the Bayes error, although the difference between the R and L errors is small. A large number of local samples is not a good selection, either. If the estimated metric \( A \) is applied to the samples outside its local region, both the R and L errors will keep going up as \( r \) increases as shown in Fig. 3.7(c). On the contrary, if the estimated metric \( A \) is applied to only a small portion of the samples inside \( L(X) \), we will obtain a curve like the one shown in Fig. 3.7(d). In this case, the Bayes error is properly bounded within certain range of \( r \), yet both the R and L errors keep going up as \( r \) increases. We can not make any decision based on this kind of curve. All of the situations in Fig. 3.7 should be avoided. In order to get a proper range for both \( I_S \) and \( I_p \), we have to run many experiments on various data with various
combinations of $I_s$ and $l$. However, because of the limitation of computing power, we cannot run all cases with a large sample size. Therefore, we look for the combination of $I_s$ and $l_p$ so that the estimated L error is the lowest and the difference between the R and L errors is reasonably small.

(a). Both $I_s$ and $l_p$ are too small. (b). Both $I_s$ and $l_p$ are too large.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(c). $I_s < I_p$.

(d). $I_s$ is too large and $l_p$ is too small.

Fig. 3.7. Typical examples of incorrect selection of the number of local samples.

In order to select a proper number of local samples, we have to avoid the four cases mentioned in Fig. 3.7. Fig. 3.8 shows the region that the good combination of $I_s$ and $l_p$ may fall in. In Fig. 3.8, the areas (a) - (d) correspond to the 4 cases mentioned in Fig. 3.7. For example, the area (a) of Fig. 3.8 has the problem described in Fig. 3.7(a), and so forth. Note that the $l_p$-axis and $I_p$-axis are not drawn to scale because they may depend not only on the data but also on the sample size, intrinsic dimensionality, etc.
Our approach for studying the effect of the number of local samples is based on the two-pass strategy. In the first pass we perform a coarse-grain search on a large area. After the first pass, we can then select a smaller area for the fine-grain search. Let us first look at the Radar-32 data, whose intrinsic dimension is about 13. We pick up 200 test samples each time for analysis and the process is repeated three times and an average is taken. We would like to see some combinations of \((I_S, I_p)\) that will give the smallest bias. Moreover, the difference between the R and L errors should not be too large; otherwise, such a wide range is almost useless because it is hard to tell where the Bayes error is. Taking the Radar-32 data for example, we perform many possible combinations of \(I_S\) and \(I_p\) with step size four and number these combinations from 1 through 21. We plot all the resulting R and L errors with respect to the combination number in Fig. 3.9. In order to proceed with the analysis, we define a performance parameter, the variation range \(\Delta\), to be

\[
\Delta = \frac{1}{2}(L - R) \tag{3.15}
\]

There are two curves in Fig. 3.9. The lower one is the R error and the upper one is the L error. The horizontal line indicates the estimated Bayes error of the Radar-32 data. The Bayes error is estimated in the next section. Since all the combinations have
been chosen to avoid the situations mentioned in Fig. 3.8, most of the combinations in Fig. 3.9 bounds the Bayes error properly. However, most of the R errors are too close to the Bayes error, leading us to overestimate the Bayes error. This phenomenon is due to sample size effect that will be discussed in the next section. The Bayes error of the Radar-32 data is estimated to be around 9%. We see in Fig. 3.9, that the combinations 4–6 and
9-11 give quite good results. A "good result" means the L error is low and the variation range $A$ is reasonably small. Since we have avoided the bad combinations mentioned in Fig. 3.7, all 21 outcomes have reasonable $A$ and have flat curves within certain ranges of metric size $r$. Note that the Bayes error estimated by local metrics tends to be overestimated if any parameters are not properly chosen; we thus look for the ones with the lowest L error.

Looking at the table in Table 3.1, we find that these combinations correspond to $I_p \approx 9$ and 13, and $I_s = 21, 25$ and 29. In order to determine more precisely the optimal combination of $(I_s, I_l)$, we run the same experiment again with smaller step size ($=2$), and the result are given in Fig. 3.10. In Fig. 3.10, we see that the performances are quite similar for the combination number from 5 through 17. All the L errors differ by less than 1%, and so do the R errors. Among all these combinations, we find the optimal combinations are $(I_s, I_p) = (20, 10)$ and $(I_s, I_p) = (28, 10)$. They have almost indistinguishable results. However, we prefer smaller values for $I_s$ and $I_p$ for it requires less computation time. Note that the optimal $I_s$ is about $1.5n_e$ and $I_p$ is 0.51s. Although only 200 samples for this 32-dimensional data were used, our algorithm offers a pretty good lower and upper bounds for the Bayes error with the variation range $A \approx 1.7\%$.

Let us perform the same procedure on the Radar-64 data to see what the optimal $I_s$ and $I_p$ are. The results of the first pass (with step size = 6) are given in Fig. 3.11. Note that for the Radar-64 data, $n_e$ is about 33. The horizontal line in Fig. 3.11 indicates the Bayes error, about 18%, of the Radar-64 data. We see in Fig. 3.11 that if $I_p$ is small, the variation range $A$ is too large. On the other hand, if $I_p$ is larger than its optimal value, not only is $A$ too small, but both the R and L errors are biased high. Fig. 3.11 shows that the pair $(I_s, I_p) = (51, 39)$ is the best combination. So, a second pass is performed around this point with step size = 2. It can be seen in Fig. 3.12 that combination 17, which corresponds to $(I_s, I_p) = (51, 39)$ in Table 3.4, gives the best result. Note that for the Radar-64 data the optimal $I_s = 1.55n_e$ and $I_p = 0.76$ Is.
We would like to point out that the estimated intrinsic dimension of the given data is just an approximation, not an exact value; therefore, a certain degree of uncertainty is expected. On the other hand, the intrinsic dimension of the artificial data is an exact number. So, we would like to perform the same procedure on the artificial data to see if the optimal combination of $(I_S, I_P)$ still falls in the same region.
Fig. 3.11. Study of $I_S$ and $I_p$ on the Radar-64 data with $N=200$ (first pass).

Table 3.3. Look-up table for Fig. 3.11

<table>
<thead>
<tr>
<th>#</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
</tr>
</thead>
<tbody>
<tr>
<td>$I_S$</td>
<td>21</td>
<td>37</td>
<td>33</td>
<td>39</td>
<td>45</td>
<td>51</td>
<td>57</td>
<td>63</td>
<td>69</td>
<td>27</td>
<td>33</td>
<td>39</td>
<td>45</td>
</tr>
<tr>
<td>$I_p$</td>
<td>21</td>
<td>21</td>
<td>21</td>
<td>21</td>
<td>21</td>
<td>21</td>
<td>21</td>
<td>21</td>
<td>27</td>
<td>27</td>
<td>27</td>
<td>27</td>
<td>27</td>
</tr>
<tr>
<td>#</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>26</td>
</tr>
<tr>
<td>$I_S$</td>
<td>51</td>
<td>57</td>
<td>63</td>
<td>69</td>
<td>33</td>
<td>39</td>
<td>45</td>
<td>51</td>
<td>57</td>
<td>63</td>
<td>69</td>
<td>39</td>
<td>45</td>
</tr>
<tr>
<td>$I_p$</td>
<td>27</td>
<td>27</td>
<td>27</td>
<td>27</td>
<td>33</td>
<td>33</td>
<td>33</td>
<td>33</td>
<td>33</td>
<td>33</td>
<td>33</td>
<td>39</td>
<td>39</td>
</tr>
<tr>
<td>#</td>
<td>27</td>
<td>28</td>
<td>29</td>
<td>30</td>
<td>31</td>
<td>32</td>
<td>33</td>
<td>34</td>
<td>35</td>
<td>36</td>
<td>37</td>
<td>38</td>
<td>39</td>
</tr>
<tr>
<td>$I_S$</td>
<td>51</td>
<td>57</td>
<td>63</td>
<td>69</td>
<td>45</td>
<td>51</td>
<td>57</td>
<td>63</td>
<td>69</td>
<td>51</td>
<td>57</td>
<td>63</td>
<td>69</td>
</tr>
<tr>
<td>$I_p$</td>
<td>39</td>
<td>39</td>
<td>39</td>
<td>39</td>
<td>45</td>
<td>45</td>
<td>45</td>
<td>45</td>
<td>45</td>
<td>51</td>
<td>51</td>
<td>51</td>
<td>51</td>
</tr>
</tbody>
</table>
Fig. 3.12. Study of $I_5$ and $I_p$ on the Radar-64 data with $N=200$ (second pass).

Table 3.4 Look-up table for Fig. 3.12.

<table>
<thead>
<tr>
<th>#</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>$I_5$</td>
<td>48</td>
<td>50</td>
<td>52</td>
<td>54</td>
<td>48</td>
<td>50</td>
<td>52</td>
<td>54</td>
<td>48</td>
</tr>
<tr>
<td>$I_p$</td>
<td>36</td>
<td>36</td>
<td>36</td>
<td>36</td>
<td>38</td>
<td>38</td>
<td>38</td>
<td>38</td>
<td>40</td>
</tr>
<tr>
<td>#</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td>$I_5$</td>
<td>50</td>
<td>52</td>
<td>54</td>
<td>48</td>
<td>50</td>
<td>52</td>
<td>54</td>
<td>51</td>
<td></td>
</tr>
<tr>
<td>$I_p$</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>42</td>
<td>42</td>
<td>42</td>
<td>42</td>
<td>39</td>
<td></td>
</tr>
</tbody>
</table>

We choose the Gaussian pulse set $\Pi$ with $n = 12$ for the experiment. This data set has an intrinsic dimensionality $n_e = 3$ and the Bayes error is about 9.6% on the parameter space. The Bayes error on the sample space might be higher than 9.6% due to the nonlinear mapping from the 3-dimensional parameter space onto the 12-dimensional sample space. However, the amount of increase in the Bayes error from the parameter
space to the sample space depends on how dense the time-sampling is and how the range of these three parameters is chosen. Based on the consideration of computation time, we generate only 1000 samples for each class. Each trial we pick up 100 samples from each class to estimate the error rate. The same process is repeated three times by picking up a different set of 100 samples for the trial. Then we take an average of the three outcomes to plot the curves in Fig. 3.13. Since the intrinsic dimension of the Gaussian pulse is only three, we do not have to apply the two-pass strategy this time. The horizontal line in Fig. 3.13 indicates the Bayes error in the parameter space. It is seen that only a few combinations bound the Bayes error properly. The combination number 4 gives the lowest L error, which is 13.83%. From Table 3.5 we find the corresponding values are $l_1 = 6$ and $l_p = 3$.

![Graph](image)

**Fig. 3.13.** Study of $l_3$ and $l_p$ on the Gaussian pulse data with $N=100$ and $n=12$. 
The intrinsic dimension of the Gaussian pulse data is exactly three. Therefore, the optimal $l_S = 2n_e$ and $l_p = 0.5l_S$. These values are very close to the results for the real data mentioned previously.

3.3.2. Sample size and number of local samples

In the previous section we found the optimal number of local samples based on a pretty small sample size, $N=200$ for the Radar-32 and Radar-64 data and $N=100$ for the Gaussian pulse data. These sample sizes are considered quite small compared to their dimensionality. One may wonder whether and how the number of local samples will change as sample size increases. Because the computational complexity of our algorithm is roughly proportional to $N^2$ for fixed $n$, the computation time required to perform the same procedure as in the previous section is astronomical using ordinary workstations. Therefore, we select $N = 1600$ for both the Radar-32 and the Radar-64 data although we can choose larger $N$. Nevertheless, increasing $N$ from 200 to 1600 should be sufficient to observe the trend of change, if any. Furthermore, in real applications, the maximal number of available samples that people normally have is also on the order of thousands. Therefore, we feel that it is appropriate to select $N = 1600$ to study the effect of sample
size on the selection of the number of local samples. For the Gaussian pulse data, we increase the sample size from $N = 100$ to $N = 800$.

When the sample size is increased from $N_1$ to $N_2$, the distribution of samples in the sample space becomes denser. The average distance between any two samples becomes shorter. Therefore, we have two different ways of choosing the local region for each test sample. If the local region for the sample $X$ is $L(X)$ for $N = N_1$, the same local region $L(X)$ now contains more neighbors for $N = N_2$. In this case, both $I_S$ and $I_P$ are expected to increase when the sample size increases. However, there is another possibility. Instead of keeping the same local region, we can keep the same number of local samples and have smaller local region $L'(X)$. The volume of $L'(X)$ is smaller than $L(X)$; therefore, better locality is attained.

Fukunaga and Flick [22] demonstrated that the expected k-th NN distance for a normal distribution can be approximated by

$$E_X E(d_{kNN}(X)) = \Gamma^{1/n} \left[ \frac{n+2}{2} \right]^{1/2} \left[ 1 - \frac{1}{n} \right]^{-n/2} \frac{\Gamma(k+1/n)}{\Gamma(k)} \frac{\Gamma(N+1)}{\Gamma(N+1+1/n)}$$

(3.16)

Let $d_1$ and $d_2$ be the average k-th NN distances for $N = N_1$ and $N = N_2$ respectively. From (3.16), we have

$$\frac{d_2}{d_1} = \frac{\Gamma(N_2 + 1)}{\Gamma(N_2 + 1 + 1/n)} \frac{\Gamma(N_1 + 1 + 1/n)}{\Gamma(N_1 + 1)}$$

(3.17)

For even moderate $N$ and $n$, (3.17) can be approximated by

$$\frac{d_2}{d_1} \approx \left( \frac{N_1 + 1}{N_2 + 1} \right)^{1/n}$$

(3.18)

For example, $N_1 = 200$, $N_2 = 1600$ and $n = 32$, the factor $d_2/d_1$ is about 0.94. If $n = 64$, then $d_2/d_1$ is about 0.97. Although the radii of the local regions for $N=200$ and $N=1600$ are quite close, the volume of $L'(X)$ is only about one eighth the volume of $L(X)$ for both $n = 32$ and $n = 64$.

On the other hand, let $d_{kNN}$ be the expected k-th NN distance for sample size $N$. For the Radar-32 data, we selected $I_p = 10$ for $N=200$, which implies that for each test sample 10 nearest neighbors are included for density estimation. We would like to find
out how many nearest neighbors can be found for $N=1600$ in the same local region. Using (3.16), we find approximately 80 samples can be found in the same local region $L(X)$ when the sample size is increased from $N = 200$ to $N = 1600$. Similarly, $l_s$ is increased from 20 to 160 as $N$ increases from 200 to 1600.

Two possible outcomes are expected when the sample size is increased from 200 to 1600. First, a smaller local region may be preferred. In this case, a similar number of local samples is expected when the sample size is increased. In other words, maintaining locality is important. Second, a larger number of local samples may be preferred. In this case, the size of the local region is not changed much, while the number of local samples changes rapidly as the sample size increases. In order to verify which argument is true, we apply the same procedure as the one in the previous section to the same three data sets.

Let us examine first the Radar-32 data. Experiments show that the optimal number of local samples is still very close to the old one even though the sample size has been increased by 8 times. As the number of local samples increases (higher than its optimal value), the bias also increases. The combination $(l_s, l_p) = (160, 80)$ gives extra high bias to the estimated error rate. From this example we verified that the optimal number of local samples is mainly determined by the intrinsic dimensionality of the data set. The impact from the change of sample size is very small. It also confirms that maintaining locality is more important than maintaining the size of the local region. This is a relief for us since a smaller number of local samples translates into a faster computation. We do not have to increase the number of local samples as the sample size $N$ increases.

In Fig. 3.14 we find that the difference between the estimated error rates using different numbers of local samples is very small. The new optimal combination for $N=1600$ is $(l_s, l_p) = (18, 12)$ as is seen in the combination number 9. The resulting error rate is $(R, L) = (7.53\%, 10.47\%)$. The optimal combination for $N=200$, $(l_s, l_p) = (20, 10)$ in the combination number 11, generates the error rate $(R, L) = (7.06\%, 10.69\%)$ which is a looser bound on the Bayes error. All these combinations give proper upper and lower bounds on the Bayes error. If we take the average of the R and L errors to be the Bayes error, the combination $(l_s, l_p) = (18, 12)$ gives the Bayes error of 9.00% with the variation
range $A = 1.47\%$. Using $(I_S, l_p) = (20, 10)$, we obtain the Bayes error of 8.88% with the variation range $A = 1.81\%$. These results are quite close. Therefore, we conclude that for the Radar-32 data it is reasonable to find the optimal number of local samples for a small sample size and use the number for any sample size. We would like to examine if the same conclusion holds for the Radar-64 data.

Fig. 3.14. Study of $I_S$ and $l_p$ on the Radar-32 data with $N=1600$.

<table>
<thead>
<tr>
<th>#</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>$I_S$</td>
<td>14</td>
<td>14</td>
<td>14</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>18</td>
<td>18</td>
<td>18</td>
<td>20</td>
</tr>
<tr>
<td>$l_p$</td>
<td>8</td>
<td>10</td>
<td>12</td>
<td>8</td>
<td>10</td>
<td>12</td>
<td>8</td>
<td>10</td>
<td>12</td>
<td>8</td>
</tr>
<tr>
<td>#</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td></td>
</tr>
<tr>
<td>$I_S$</td>
<td>20</td>
<td>20</td>
<td>22</td>
<td>22</td>
<td>24</td>
<td>24</td>
<td>26</td>
<td>26</td>
<td>160</td>
<td></td>
</tr>
<tr>
<td>$l_p$</td>
<td>10</td>
<td>12</td>
<td>10</td>
<td>12</td>
<td>14</td>
<td>16</td>
<td>14</td>
<td>16</td>
<td>80</td>
<td></td>
</tr>
</tbody>
</table>
The same procedure is applied to the Radar-64 data and the results are shown in Fig. 3.15. From Fig. 3.15 we can see that the previous optimal combination \((I_S, I_p) = (51, 39)\) for \(N=200\) is still a very good choice for \(N=1600\). The resulting error rate is \((R, L) = (16.0\%, 19.4\%)\). Taking the average of the R and L errors, we get the estimated Bayes error \(\widehat{z} \approx 17.7\%\) with variation range \(\Delta=1.7\%\). Although some other combinations give a little lower L error, we believe that it is not worth the effort to look for the optimal number of local samples for each sample size, because it is extremely time-consuming and the improvement is very trivial. From the study of the Radar-32 and Radar-64 data, we have learned that if the number of nearest neighbors falls in the "good region" in Fig. 3.8, the Bayes error can be properly bounded. Before we make a conclusion for this topic, let us examine the Gaussian pulse data.

\[\text{Fig. 3.15. Study of } I_S \text{ and } I_p \text{ on the Radar-64 data with } N=1600.\]
In the previous experiment, we found that the optimal number of local samples for the Gaussian pulse data is \((l_s, l_p) = (6, 3)\). Now the sample size is increased from \(N=100\) to \(N=800\) and the new results are shown in Fig. 3.16. If we use \((l_s, l_p) = (6, 3)\) for the case \(N=800\), the Bayes error tends to be underestimated because the L error is very close to the Bayes error. This time the optimal choice is \((l_s, l_p) = (6, 4)\), which gives the estimated error rate \((R, L) = (8.00\%, 10.94\%)\). The resulting Bayes error is estimated to be 9.5% with the variation range \(A = 1.47\%\). This value is very good, since the Bayes error is about 9.6% in the parameter space. However, the change in the number of local samples is very small even if the sample size is increased by 8 times.

Based on our study of the real and artificial data sets, we conclude the issue of the selection of the number of local samples as follows.

1). The optimal number of local samples is mainly determined by the intrinsic dimensionality \(n_p\) of the data set.

2). In our algorithm, the inequality, \(l_B \geq l_s \geq l_p\), should hold true when selecting the number of local samples.
The optimal number of local samples is data-dependent and must be chosen experimentally for each data set. However, it is not necessary to search through all the combinations. Our experiments suggest that the optimal one may be found in $1.5 n_e \leq l_S \leq 2 n_e$ and $0.5 l_S \leq l_p \leq 0.8 l_S$. When new data are being processed, we can start to look for optimal number of local samples first with these combinations.
4). The impact of the change of the sample size is very small on the optimal number of local samples.

5). The optimal number of local samples can always be found where the L error is the lowest if the sample size is small. However, it is not always true for a large sample size.

Before the end of this section, we would like to mention a problem related to the issue of the number of local samples. So far our experience with both the artificial and real data sets is limited to the case where the intrinsic dimensionalities of the two classes are the same or almost the same. If one of the classes has a large intrinsic dimensionality, say 100, while the other class has a much smaller intrinsic dimensionality, say 10, we do not know how to select an optimal number of local samples. One possible solution may be that we have to choose different numbers of local samples for different classes. Since we have not seen such data, we can not pursue this topic further. In case we should encounter this kind of situation, we may have to seriously study this issue.

3.4. Sample Size Effect

In this section, the effect of sample size is stated. Take the quadratic classifier for example. When the sample size increases, the estimated R error will increase, while the estimated L error will decrease. These two curves will approach the quadratic error as the sample size increases as shown in Fig. 3.17. For the quadratic classifier, the horizontal line in Fig. 3.17 stands for the quadratic error. The upper and lower curves indicate the L and R errors, respectively. The nonparametric classifiers, such as k-NN and Parzen classifiers, behave similarly if global metrics are used. However, the sample size effect for the nonparametric classifier using local metrics is quite different.

Take the Parzen classifier for example. The global metric is obtained from all the \( N \) samples. The density at the test sample can be obtained by applying (2.1). Imposing this
Fig. 3.17. Sample size effect for the parametric classifier and the nonparametric classifier with global metrics.

metric centered at the test sample and evaluating the kernel function at all other samples, we can obtain the density estimate of the test sample by summing up all the N kernel functions. If the sample size N increases, then the impact of removing one sample from the test set gets less. This explains why the R and L errors get close as N increases. Nevertheless, this is not true when local metrics are used. In the case where local metrics are used, the density estimate of the Parzen classifier in (2.1) has to be modified by replacing N with \( l_p \). In addition, the global metric A has to be replaced with a local metric which is different for each test sample. The summation is performed on the \( l \), nearest neighbors only. As we can see from the previous section that the value of \( l_p \) is determined by the intrinsic dimensionality of the data set. No matter how large the sample size may be, the value of \( l_p \) is almost fixed. Therefore, the variation range \( A \) does not go to zero even as the sample size increases. On the contrary, the variation range \( A \) is almost fixed except for extremely small and large sample sizes. For an extremely small sample size as shown in Fig. 3.6, the selection of neighbors causes extra bias to the estimated error rate.
Therefore, a larger variation range is expected for an extremely small sample size. On the other hand, if the distribution of the samples is so dense that removing one sample from the local region has a very slight impact on the estimated probability density, the variation range can be made as small as we wish. However, for a large $n$, we probably need trillions or even more samples to satisfy the hypothesis that the samples are "densely distributed".

Now let us consider the case where the sample size is very small. No matter how we build up the local region $L(X)$ for the test sample $X$, the samples inside $L(X)$ are in fact not the neighbors of the test sample $X$. These samples can not help find the intrinsic space in which the test sample $X$ is located. Therefore, the estimated metric $A$ is far from its optimal value. Using this poor metric $A$ to measure distances between samples will result in a higher probability of misclassification. Thus, both the $R$ and $L$ errors will be biased positively. This bias increases as the sample size decreases. Based on the reasoning above, we conjecture that the sample size effect for nonparametric classifiers with local metrics may look like the curves in Fig. 3.18. The horizontal line in Fig. 3.18 indicates the Bayes error. If the sample size is small, the $R$ error will be very close to the Bayes error, while the $L$ error will be much higher than the Bayes error. As the sample size increases, both the $R$ and $L$ errors go down, but $L$ error decreases much faster than the $R$ error. They finally stabilize and bound the Bayes error with a constant variation range. Note that if any of the parameters in our algorithm is not properly chosen, we may not be able to bound the Bayes error. Also note that all the test samples must be uncorrelated. The effect of sample correlation will be discussed later.

In order to determine if our conjecture is true, we test all three artificial data sets described in Appendix A in addition to the two real data sets. Since the Bayes errors of the artificial data sets in their parameter spaces are known, it also helps to prove that our algorithm does not overestimate or underestimate the Bayes error. Our experiences with the artificial data sets also help us to determine the Bayes error of the real data sets.

Now let us examine the sample size effect of the Linear data set. The intrinsic dimension of this data set is exactly three. Each 3-dimensional parameter vector is linearly mapped to an 8-dimensional sample space. The Bayes error on the parameter space is
Fig. 3.18. Sample size effect for the nonparametric classifier with local metrics.

about 9.6%. Since all three parameters are independent Gaussian random variables, the samples are also Gaussian random vectors with the same overlap region, i.e., the same Bayes error. The result is given in Fig. 3.19. We see that the Bayes error is properly bounded. For this data, we use $l_\theta=100$, $l_\xi=6$ and $l_p=4$.

Now let us examine the Gaussian pulse data set I with $n=12$. Only one of the three parameters provides useful classification information. The magnitude of the observed sample is linearly proportional to the magnitude of the parameter which bears classification information. Therefore, if the time-sampling is dense enough and the range of the parameters are properly chosen, we should be able to obtain the same Bayes error in the sample space as the one in the parameter space. The Bayes error on the parameter space is exactly 10%. Applying our algorithm, we obtain the results in Fig. 3.20. We see that the curves in Fig. 3.20 resemble the ones in Fig. 3.18. Again the Bayes error is properly bounded. For this data, we use $l_\theta=100$, $l_\xi=6$ and $l_p=4$. 
The Gaussian pulse data set II is more difficult to handle. All three parameters contribute equally to the classification. They go through nonlinear mapping and are then time-sampled to form 12-dimensional sample vectors. Using the conventional Parzen
classifier with the global metric, one may find that the Bayes error on the sample space is as large as 15%. However, our algorithm has proved to be very powerful in handling this data set. The estimated error rate is given in Fig. 3.21. Once again, the Bayes error is properly bounded. For this data, we use $I_B=100$, $I_S=6$ and $I_P=4$.

![Sample size effect for the Gaussian pulse data set II with n = 12.](image)

**Fig. 3.21.** Sample size effect for the Gaussian pulse data set II with n = 12.

Encouraged by the experimental results of the artificial data sets, we go on to determine the Bayes error of the Radar-32 and Radar-64 data sets. All the parameters are selected with the help of the discussion in the previous two sections. We use $I_B=200$, $I_S=20$ and $I_P=10$. For the Radar-32 data, we obtain the R and L errors vs. sample size plotted in Fig. 3.22. The curves in Fig. 3.22 also resemble the ones in Fig. 3.18. When the sample size is increased beyond 800, the error curves stabilize. Since the test samples are uncorrelated, we thus determine the Bayes error of the Radar-32 data set to be about 9%. Note that if we use the global metric, the estimated "Bayes error" is about 14%. The large difference is caused by the fact that the Radar-32 data is multi-clustered and non-Gaussian.
Applying our algorithm to the Radar-64 data, we obtain the curves shown in Fig. 3.23. We use \( I_B = 1000 \), \( I_L = 51 \) and \( I_p = 39 \). The Bayes error is estimated to be around 18%. If the global metric is used, the false "Bayes error" is estimated to be around 20%. The difference between the error rates estimated by the local and global metric is quite small for this data set. The main reason for such a small difference is that this data set is very close to a Gaussian distribution. Although the samples are divided into four windows based on their elevation and azimuth angles, these four windows do not differ very much.

We have emphasized that the samples must be uncorrelated in order to successfully estimate the Bayes error. The effect of sample correlation is not fully known so far. In order to avoid the unknown impact caused by sample correlation, we thus stick to uncorrelated samples. There are many kinds of sample correlations. Let us consider the 2-dimensional distribution depicted in Fig. 3.24 to get some insight into this problem.
Fig. 3.23. Sample size effect for the Radar-64 data.

Fig. 3.24. An example of 2-D distribution of a data set without sample correlation.
A cross represents the position of a $\omega_1$ sample. The position of a $\omega_2$ sample is represented by a circle. There is one sample out of each class falling on the wrong side of the decision boundary. They will be misclassified no matter if a global or local metric is used, because all neighbors are samples from the other class. Let us consider an extreme case where the correlated samples are very close to the sample they are correlated with. These correlated samples are added to the original data set. Fig. 3.25 shows how these correlated samples are distributed. The smaller crosses and circles stand for the added correlated samples. Since we use a local metric to estimate density for those originally misclassified samples, circled with dashed lines, they can find a local region full of samples from the same class. Therefore, they will not be misclassified. In this case, our algorithm will tend to come up with an error rate lower than its actual value. In other words, the Bayes error will be underestimated. In real-world applications, the sample correlation is much more complex than the case we mentioned here, and thus can not be analyzed in terms of mathematical expressions. Therefore, we insist that the test samples must be uncorrelated to avoid the impact of the sample correlation.

Fig. 3.25. An example of 2-D distribution of a data set with correlated samples.
3.5. Kernel Shape Effect

The last factor that may affect the performance of our algorithm is the selection of kernel function. The kernel shape is governed by the parameter $m$ in (2.1). A larger $m$ implies faster roll-off of the kernel function. On the other hand, smaller $m$ implies slower drop-off of the kernel function. When $m$ approaches infinity, the kernel function approaches a hyperelliptical (uniform) kernel. For all samples inside the kernel, they are given the same weighting, while samples outside the kernel are simply excluded from density estimation. There is no theoretical proof governing which kernel function is the optimal one. For simplicity, we used a Gaussian kernel ($m = 1$) for all the experiments. In this section, we want to study the effect of kernel shape on the Bayes error estimation.

The distributions of the artificial data sets are either Gaussian or nearly Gaussian. Therefore, the Gaussian kernel might be the best kernel. Experiments show that a Gaussian kernel does an excellent job in finding the Bayes error of all the artificial data sets. On the other hand, the distribution of the real data is not Gaussian. Therefore, we would like to know if the preference of kernel function is distribution-dependent. We would also like to know whether using a non-optimal kernel function will lead to a great degradation of our algorithm. Varying the kernel shape $m$ from 0.2 to 2 and picking up 200 samples from the Radar-32 data set each time for experiments, we repeat three times to obtain the average R and L error curves vs. the kernel shape $m$ as shown in Fig. 3.26. The lower and upper curves in Fig. 3.26 stand for the R and L errors respectively. The dashed line represents the estimated Bayes error. We notice that for $m$ larger than 0.6 the R and L error curves go up monotonically as $m$ increases. The optimal kernel for the Radar-32 data is about $m = 0.6$. However, the Gaussian kernel, $m = 1$, is acceptable. As $m$ gets smaller than 0.6, the R error goes down rapidly, resulting in a large variation range. Within the range $0.6 \leq m \leq 1.4$, the effect of kernel shape is very small.

Now let us look at the Radar-64 data. Applying the same procedure, we obtain the error curves in Fig. 3.27. Like the Radar-32 data, the R error increases as $m$ increases. However, the L error stays at almost the same value for different values of $m$. It is hard to tell which $m$ is optimal. Based on the consideration of a reasonable variation range, we
Fig. 3.26. Kernel shape effect of the Radar-32 data.

Fig. 3.27. Kernel shape effect of the Radar-64 data.
would prefer to have $0.8 \leq m \leq 1.4$.

From these two examples we have learned that although the Gaussian kernel may not be the optimal kernel function, the performance of the Gaussian kernel is very close to the optimal one. Therefore, we suggest that it is not necessary to search for the optimal kernel function unless the data set is known to have a very unusual distribution.

3.6. Conclusion

In this chapter, we have discussed the selection of the metric $B$ which is used to build up a local region and to determine nearest neighbors. The optimal way of selecting the metric $B$ is to take advantage of the viewing angle information. We have also discussed the relation between the number of local samples and the intrinsic dimensionality of the data. The sample size has only minor impact on the selection of the number of nearest neighbors. We provide an empirically good region that the optimal number of nearest neighbors may fall in. Searching for the optimal number of local samples is a very time-consuming job. Our experiments provide helpful guidelines for how to quickly find the optimal value. We have discussed and demonstrated the distinctive behavior of the sample size effect for the local metric. We have also argued that correlated samples are not suitable to be used as test samples for the purpose of Bayes error estimation. Finally, we demonstrated by two examples that the Gaussian kernel is good for data with non-Gaussian distribution.

In summary, we would like to offer a step-by-step guideline for how to estimate the Bayes error using local metrics.

1). Determine the intrinsic dimension $n_e$ of the data set.

2). Make sure all the test samples are uncorrelated.

3). If the distribution of samples is close to a Gaussian distribution, we may expect the Bayes error to be slightly lower than the quadratic error. If not, the Bayes
error could be much lower than the quadratic error.

4). Use the viewing angle information to build up the local metric $B$ for the test sample. The selection of $l_B$ is data-dependent and it should be determined experimentally.

5). Search for the optimal number of local samples. The optimal number of local samples for computing local autocorrelation, $l_s$, might be found in the range $1.5n_e \leq l_s \leq 2n_e$. Once the value of $l_S$ is determined, the number of local samples used for density estimation, $L$, can be found in $0.5l_S \leq l_p \leq 0.8l_S$.

6). Increase the number of test samples and plot the resulting error rate vs. the sample size. If the curves stay flat or change very slowly with the sample size, the Bayes error is then properly bounded. If the error curves go down rapidly as sample size increases and if the error curves keep going down even though all the samples are used, we can conclude that the sample size is too small to estimate the Bayes error. In this case, the real Bayes error is not known because the given samples are insufficient to represent the distribution of the two classes.

In step 1, the procedure of estimating the intrinsic dimensionality is involved. For data with a low dimensionality and a sufficiently large sample size, the intrinsic dimensionality can be easily estimated using conventional methods. However, if the given data set has only 1,000 samples for each class and its dimension is as high as 4,096, there are no existing methods that can estimate the intrinsic dimension of this data set. In order to perform our algorithm for the Bayes error estimation, we have to solve this problem. The quadratic classification in step 3 is also very difficult as mentioned in Chapter 1. These problems will be solved in the latter chapters.
CHAPTER 4
PARAMETRIC ANALYSIS OF TWO-DIMENSIONAL RADAR IMAGES

4.1. Introduction

From this chapter on we are going to deal with problems relating to 2-dimensional radar images. Normally radar images like SAR or ISAR data have a very large dimension \( n \), while the sample size \( N \) is limited due to the difficulty of data acquisition. Therefore, we inevitably encounter the situation where \( N < n \). At the end of the previous chapter, we mentioned that we had to perform quadratic classification to obtain the quadratic error as an upper bound for the Bayes error. The quadratic error will offer us a very important reference for selecting the parameters of our Bayes error estimation algorithm. For low dimensional data, the quadratic error is very easy to obtain. However, for large dimensional data with \( N < n \), the problem gets complicated. Before we perform nonparametric classification on the 2-dimensional images in Chapter 5, we would like to solve the problems of parametric classification with huge \( n \) and relatively small \( N \).

Our solution that will alleviate the difficulty of quadratic classification for large \( n \) and small \( N \) is to measure distance by window-shifting. The window-shifting algorithm is developed mainly to solve the difficulty of inverting huge covariance matrices and to simplify the computation of distances between samples. It is also intended to be used in parametric classifiers such as a quadratic classifier to achieve better performance than the existing methods. Instead of incorporating the \( n \times n \) full covariance in the computation of distances, we compute the distance between samples by shifting a small window over the
sample images. This approach is described in detail in Section 4.2.

The Bhattacharyya distance is an upper bound of the Bayes error for the data with Gaussian distributions and is often used in real applications to obtain a quick measure for the separability of the two-class problem even though the distributions may not be Gaussian. The computation of the Bhattacharyya distance involves the inversion of covariance matrices. Since we encounter the difficulty of extremely small sample size, we would like to study the sample size effect on the Bhattacharyya distance. Based on the technique introduced in Section 4.2, we would also like to study the effect of window-shifting on the Bhattacharyya distance. These issues are discussed in Section 4.3. A side issue related to the Bhattacharyya distance is the outlier's effect. It is generally believed that outliers may have undesired impact on the estimated Bhattacharyya distance. This issue is investigated in Section 4.4.

Due to the generating process of the SAR images, we encounter the situation that the number of uncorrelated samples is far less than the data dimensionality, i.e., \(N<n\), although a large number of correlated samples may be available. As a result, the covariance matrices obtained only from the uncorrelated samples become singular and the distances normalized by covariance matrices cannot be computed. Even if this problem is solved, we must face another serious problem due to limited sample size in that we can not obtain a reasonably tight upper bound by the leave-one-out (L) method and a lower bound by the resubstitution (R) method for the quadratic error. It is sometimes questioned whether correlated samples are acceptable to help obtain tighter bounds on the quadratic error. We want to discuss this issue in Section 4.5. In Section 4.6, we demonstrate that the quadratic classifier by window-shifting generates better performance than the existing methods do. Conclusion is presented in Section 4.7.

Our goal is that all the algorithms we develop in this thesis can be applied to the SAR images, which usually have the dimension \(n=64\times64=4,096\) and the sample size \(N\geq1,000\). However, at the press time of this thesis, we still have not received the SAR data for processing. Therefore, we have to process the smaller dimensional ISAR images \((n=20\times32)\), which are made available by Lincoln Laboratory, Massachusetts Institute of
Technology. We have 246 uncorrelated samples and 4930 correlated samples for each class (Appendix A). Since these ISAR data are also 2-dimensional radar images, we believe that they provide a good alternative for us to develop and test our algorithms. Furthermore, we downsize the image by averaging neighboring four pixels on the original image to get a new pixel on the new reduced-dimensional image. The downsizing ratio is 4-to-1. Therefore, we have generated two more sets of ISAR data of \( n=10\times16 \) and \( n=5\times8 \), in addition to the original data set of \( n=20\times32 \). In the past, we have accumulated our experiences on high dimensional data. So far the highest dimensional data we have treated was the 64-dimensional radar range-profile data. Also, we have learned that each time the dimension gets higher, new technical problems occur. Therefore, before treating the ISAR data of \( n=20\times32 = 640 \), we felt the need to study the cases of \( n=5\times8 = 40 \) and \( n=10\times16 = 160 \). More detailed information on the ISAR data can be found in Appendix A.

4.2. Window-Shifting Distance Measurement Algorithm

For the data set with \( N < n \), the inverse of the full covariance cannot be obtained. Even when \( N > n \) and the covariance matrix is invertible, the inversion process is very difficult and not accurate for extremely large \( n \). To alleviate this difficulty, many of the classifier designers discard all the inter-pixel correlation information. Namely, they approximate the covariance matrix by a diagonal matrix. This is a very crude approximation and the resulting performance is not satisfactory. Some researchers suggest that the Toeplitz approximation be used to approximate the correlation matrix (p.160 of [14]). The correlation coefficients in the correlation matrix with equal distance to the diagonal axis are set to be equal, and these correlation coefficients decrease as the distance to the diagonal axis increases. For one-dimensional vectors, this may be a feasible solution, since for the one-dimensional data, indeed, the correlation between pixels \( i \) and \( j \) decreases as \( |i-j| \) increases. The correlation between pixels \( i \) and \( j \) directly projects to the \( ij \)-th entry of the covariance matrix. However, for 2-dimensional images we believe that
the Toeplitz approximation is far more complex. We have to consider both the vertical and horizontal correlations among neighboring pixels, and the mapping from the pixel positions to the covariance matrix can not be easily visualized. Therefore, we propose that the approximation be made by shifting a window of size \(wxw\) over the sample images, not over the covariance matrix. By adjusting the window size \(w\), we can control the degree of approximation.

For a window of size \(wxw\), there are \(w^2\) pixels covered by the window. We form a \(w^2\times 1\) vector from these pixels, and the window covariance is of size \(w^2\times w^2\). If the original data dimensionality is \(n\), the covariance matrix is of size \(nxn\). Take the SAR data for example, the dimensionality is \(n = 64\times 64 = 4096\); therefore, the full covariance matrix is of size \(4096 \times 4096\). If we use a window of size \(3\times 3\), we end up by approximating the covariance matrix by many \(9\times 9\) matrices. Obviously, some classification information is lost due to the dimensional reduction. The proposed method with window size \(w=1\) corresponds to the case where the covariance matrix is approximated by a diagonal one. By increasing the window size \(w\), our method can take advantage of more covariance terms. From our study on many real data sets, we observed that the nearby pixels are highly correlated, while pixels that are far apart are generally uncorrelated. Therefore, we believe that with a moderate window size, we can achieve reasonable performance relative to that of the full covariance. Since our data are 2-dimensional images, we decide to shift the window over the sample images rather than over the covariance matrix. However, our approach can be applied to the 1-dimensional sample vectors by choosing a rectangular window of size \(wx1\). For the 1-dimensional samples, moving the rectangular window over the sample vectors is equivalent to moving a square window of size \(wxw\) along the diagonal axis of the covariance matrix. However, this is not true for the 2-dimensional data, because the mapping from the sample pixels to the covariance matrix is far more complicated than that of the 1-dimensional counterpart.

Given images \(Q_k\), \(k = 1, 2, \ldots, N\), each of size \(s_r \times s_c\), where the subscript \(r\) and \(c\) stands for row and column, respectively. The square window to be shifted over the image is of size \(wxw\), as shown in Fig. 4.1, where the constraint, \(1 \leq w \leq \min\{s_r, s_c\}\), must be
satisfied. The window is first placed in the top left position and then shifted to the right by one pixel each time. When the window reaches the top right position, then the window down shifts by one pixel and return to the leftmost position. The window is shifted rowwise until the window reaches the bottom right corner of the image. The number of window positions in the horizontal direction is \((s_c - w + 1)\), while the number in the vertical direction is \((s_c - w + 1)\). Therefore, the total number of window covariances over the whole image is \((s, - w + 1) \times (s_c - w + 1)\).

Let \(\Omega_{y}^{(k)}\) denote the window with its top left corner on the \(i\)-th row and \(j\)-th column of the image \(Q_k\). The pixel on the \(i\)-th row and \(j\)-th column of the image \(Q_k\) can be denoted as \(Q_k(i, j)\). Therefore, the pixel on the \(m_1\)-th row and \(m_2\)-th column of the window \(\Omega_{y}^{(k)}\) has a one-to-one correspondence to the image \(Q\) by

\[
\Omega_{y}^{(k)}(m_1, m_2) = Q_k(i + m_1 - 1, j + m_2 - 1)
\]

where \(1 \leq m_1, m_2 \leq w\) and \(1 \leq i (s, - w + 1)\), \(1 \leq j (s_c - w + 1)\) and \(1 \leq k \leq N\).
The window $\Omega_{y}^{(k)}$ cuts out a $wxw$ area on each of the N images. By concatenating the columns of $\Omega_{y}^{(k)}$, the window $\Omega_{y}^{(k)}$ can be reshaped columnwise to become a $w^2 \times 1$ vector $G_{y}^{(k)}$. Therefore, the mean vector and the covariance matrix of this area can be computed by

$$M_{y} = \frac{1}{N} \sum_{k=1}^{N} G_{y}^{(k)}$$

$$\Sigma_{y} = \frac{1}{N-1} \sum_{k=1}^{N} (G_{y}^{(k)} - M_{y})(G_{y}^{(k)} - M_{y})^T$$

The size of the mean vector $M_{y}$ is $w^2 \times 1$, while the covariance matrix $\Sigma_{y}$ is of size $w^2 \times w^2$. These mean vectors and covariance matrices are computed and stored in memory before starting to compute the distances between samples. Although we may encounter the case that $N < n$ such that the full covariance is singular, all these small window covariance matrices are invertible because $w^2 << N$.

Now we are ready to compute the distance between images $Q_k$ and $Q_l$:

$$d^2(Q_k,Q_l) = \frac{1}{a} \sum_{i=1}^{s_r-w+1} \sum_{j=1}^{s_c-w+1} (G_{y}^{(k)} - G_{y}^{(l)})^T \Sigma_{y}^{-1} (G_{y}^{(k)} - G_{y}^{(l)})$$

The constant $a$ in (4.4) is a scaling factor introduced to counteract the effect that each pixel is included in the computation of distance $a$ times on average. For $w = 1$, each pixel is involved in the distance computation only once; therefore, $a = 1$. As window size increases, $a$ increases also. Let us look at an example where $w = 3$ and the image size is $8 \times 8$. Fig. 4.2 demonstrates the number of times each pixel is involved in the distance computation during window-shifting. In Fig. 4.2, each pixel is covered by the shifting window 5.06 times on average. Therefore, $a = 5.06$ in this case. We see that the pixels close to the center of the image are covered by the shifting window $w^2$ (=9) times. However, the pixels at the fringe of the image are covered by the window less than $w^2$ times. In general, $a$ can be expressed as

$$a = \frac{w^2(s_r-w+1)(s_c-w+1)}{s_r s_c} \quad \text{for } s_r \geq 2w \text{ and } s_c \geq 2w$$

(4.5)
Fig. 4.2. The number of times each pixel of an 8x8 image involved in the distance computation with w=3.

Eq.(4.5) shows that if the window size w is much smaller than the image size, each pixel is involved in the computation of distance approximately $w^2$ times. However, this is not true when the window size is comparable to the image size.

Mathematically, this constant term a is meaningful in defining the distance measured by window-shifting. However, in practice we find this constant term a redundant for quadratic classifiers because we select the same window size for computing distances in both interclass and intraclass samples. Although this constant term a does affect the value of $\ln |\Sigma|$, it can be ignored, because the decision threshold is adjusted experimentally. In this case, this scaling factor a can be dropped for computational efficiency without affecting the results. For quadratic classifiers, the distance between the test sample $Q_k$ and the mean vector of class $I$, $I=1$, 2, has to be computed; thus, the distance can be obtained by

$$d^2(Q_k, M^{(I)}) = \frac{1}{\alpha} \sum_{i=1}^{w} \sum_{j=1}^{w+1} (G_{ij}^{(k)} - M_{ij}^{(I)})^\top \Sigma_{ij}^{-1} (G_{ij}^{(k)} - M_{ij}^{(I)})$$  (4.6)

Another practical concern about the window-shifting method is the computational complexity. We do not want to see that the overall computation time is greatly increased
due to the window-shifting distance measurement. Since the mean and the covariance matrices are computed in advance, we do not have to worry about the computation time it takes with or without window-shifting. We thus concentrate on the number of scalar multiplications and additions required to complete each distance measurement, namely, to compute $d = \sqrt{\mathbf{X}^T \Sigma^{-1} \mathbf{X}}$. Assume the data dimensionality is $n$, $n = s_r \times s_c$, where $s_r$ and $s_c$ are the number of rows and columns of the image, respectively. The number of scalar multiplications is $n^2 + n$ and the number of scalar additions is $n^2 - 1$. Using the window-shifting approach with window size $w$, the total number of window positions is $(s_r - w + 1)(s_c - w + 1)$. The size of these small covariances is $w^2 \times w^2$. For each sub-distance, the number of multiplications is $w^4 + w^2$ and the number of additions is $w^4 - 1$. In addition, we need to sum up these sub-distances, which requires $(s_r - w + 1)(s_c - w + 1) - 1$ additions. After collecting all the terms and rearranging, we find that

the total number of multiplications $= (s_r - w + 1)(s_c - w + 1)(w^4 + w^2)$ \hspace{1cm} (4.7)

and

the total number of additions $= (s_r - w + 1)(s_c - w + 1) w^4 - 1$ \hspace{1cm} (4.8)

Let us look at some examples to see the comparison of the computational complexity between the conventional and window-shifting methods. For example, the ISAR images are of the dimension 20x32. The computational complexities of the window-shifting method with window size from 1 to 5 are shown in Table 4.1. The computational complexity of the conventional method (using the fill covariance) is included in the last column for comparison. Table 4.1 shows that much less computation time is required for smaller windows compared with the one using the fill covariance. However, the computational complexity of the window-shifting method grows rapidly as $w$ increases. When $w=15$, the number of multiplications and additions is 5,491,800 and 5,467,499, respectively. It is more time-consuming than the conventional method by over ten times. In addition, there is some overhead involved in determining which pixels are to be included in each window. This overhead depends on the programming language. If
complied codes, such as C, are used, this overhead is believed to be quite small. However, if non-compiled codes, such as MATLAB, are used for programming, this overhead can be very severe. Thus, we do not intend to discuss the overhead here.

Table 4.1. The computational complexity of the window-shifting method for ISAR data, $n=20\times32$.

<table>
<thead>
<tr>
<th>computational complexity</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>full cov.</th>
</tr>
</thead>
<tbody>
<tr>
<td># of multiplication</td>
<td>1,280</td>
<td>11,780</td>
<td>48,600</td>
<td>134,096</td>
<td>291,200</td>
<td>410,240</td>
</tr>
<tr>
<td># of addition</td>
<td>639</td>
<td>9,423</td>
<td>43,739</td>
<td>126,207</td>
<td>279,999</td>
<td>409,599</td>
</tr>
</tbody>
</table>

Let us look at another example, the SAR data. The dimensionality is $64\times64$. Applying (4.7) and (4.8), we obtain the computational complexity shown in Table 4.2. Note that the computational complexity is proportional to $n^2$ if full covariance is used. We see an increase of about 41 times in computation time, although the data dimension increases by only 6.4 times (from $n=640$ to $n=4096$). As in the former example, we will have an advantage on the computation time when a small window is used. For larger window size like $w=30$, the number of multiplications and additions rises to about $9.9\times10^8$. For such an astronomical number, we believe that it is impractical to use large windows in any case.

Table 4.2. The computational complexity of the window-shifting method for SAR data, $n=64\times64$.

<table>
<thead>
<tr>
<th>computational complexity</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>full cov.</th>
</tr>
</thead>
<tbody>
<tr>
<td># of multiplication</td>
<td>8,192</td>
<td>79,380</td>
<td>345,960</td>
<td>1,012,112</td>
<td>2,340,000</td>
<td>16,781,312</td>
</tr>
<tr>
<td># of addition</td>
<td>4,095</td>
<td>63,503</td>
<td>311,363</td>
<td>952,575</td>
<td>2,249,999</td>
<td>16,777,215</td>
</tr>
</tbody>
</table>
4.3. Bhattacharyya Distance by Window-Shifting

Many works have been done to derive an upper bound for the Bayes error. Among these works, the Chernoff [19] and the Bhattacharyya [20] bounds are the most commonly used ones, where the Bhattacharyya bound is a special case of the Chernoff bound. In many applications, the Bhattacharyya bound is computed prior to any further processing of the collected samples to get a quick insight into the separability of the two distributions. For Gaussian distributions, the Bhattacharyya distance $\mu$ can be defined as

$$
\mu = \frac{1}{8} (M_2 - M_1)^T \left[ \frac{\Sigma_1 + \Sigma_2}{2} \right]^{-1} (M_2 - M_1) + \frac{1}{2} \ln \frac{\Sigma_1 + \Sigma_2}{2 \sqrt{\Sigma_1 \Sigma_2}} 
$$

where $\mu_1$ and $\mu_2$ are the first and second terms respectively.

The term $\mu_1$ represents the separability contributed from the mean vectors. The farther apart these two mean vectors are, the larger the term $\mu_1$ is. If $M_1 = M_2$, then $\mu_1$ is exactly zero. On the other hand, the term $\mu_2$ contains the separability information from the covariance matrices. If $\Sigma_1 = \Sigma_2$, the term $\mu_2$ is exactly zero. From the magnitude of the terms $\mu_1$ and $\mu_2$, we know immediately where the separability comes from.

In fact, it is often encountered that, for data with non-Gaussian distributions, the estimated Bhattacharyya distance might not be an upper bound for the Bayes error. Also, the Bhattacharyya distance is very sensitive to the sample size. Since our goal is to estimate the Bhattacharyya distance for data with a very high dimension and a small sample size, we would like study the behavior of the estimated Bhattacharyya distance under these circumstances.

Without knowing the density distributions of the data sets, we can only use the samples to estimate the sample means and sample covariances and use these to estimate the Bhattacharyya distance. Fukunaga and Hayes [45] pointed out that the asymptotic Bhattacharyya distance, $\mu$, is approximately related to the estimated Bhattacharyya distance using $N$ samples, $\mu_N$, by
\[ \mu_N \approx \mu + \frac{c}{N} \]  

(4.10)

where \( c \) is a scaling constant and the sample size \( N \) is assumed to be large enough to represent the underlying density. Therefore, by estimating \( \mu_N \) with different sample sizes, we can obtain an approximate value for \( \mu \) using linear regression. If the sample size is extremely small, the Bhattacharyya distance will be severely overestimated and the linear extrapolation may project to a completely erroneous result. Let us take the Radar-32 and Radar-64 data for examples. We plot the estimated Bhattacharyya distance of these two data sets vs. the reciprocal of sample size, \( 1/N \), in Fig. 4.3.

(a). Radar-32 data. 
(b). Radar-64 data.

Fig. 4.3. Estimated Bhattacharyya distance vs. \( 1/N \) for (a) Radar-32 data; (b) Radar-64 data.

Let us look at Fig. 4.3(a). The crosses 'x' represent the Bhattacharyya distance estimated using various numbers of samples. The solid line represents the minimum mean square error (MMSE) line estimated by those discrete points in the graph with sample sizes from 200 to 6,400. We see that the Bhattacharyya distances estimated using 50 and
100 samples lie well above the line. The Radar-64 data in Fig. 4.3(b) also show the same trend. The asymptotic Bhattacharyya distance for both data sets may be estimated by using (4.10) with \( N=200 \) and 400. However, it must be pointed out that (4.10) with \( N=100 \) and 200 give us erroneous Bhattacharyya distances as the dashed lines indicate in both Fig. 4.3(a) and (b). We conjecture that as the data dimension increases, a larger number of samples are needed in order to apply (4.10) to estimate the Bhattacharyya distance. We foresee that there will be a difficulty in estimating the Bhattacharyya distance for the SAR data because of the small sample size and the large dimensionality.

In order to alleviate this difficulty, we would like to see if the window-shifting method can help to estimate the Bhattacharyya distance for small sample size problems. If the window-shifting approach is applied to approximate the covariance matrices, the terms \( \mu_1 \) and \( \mu_2 \) can be rewritten as

\[
\mu_{1w} = \frac{1}{\alpha} \sum_{i=1}^{s_2} \sum_{j=1}^{s_1} \frac{1}{8} (M_{2ij} - M_{1ij}) \left[ \frac{\Sigma_{1ij} + \Sigma_{2ij}}{2} \right]^{-1} (M_{2ij} - M_{1ij}) \tag{4.11}
\]

and

\[
\mu_{2w} = \frac{1}{\alpha} \sum_{i=1}^{s_2} \sum_{j=1}^{s_1} \frac{1}{2} \ln \left[ \frac{\Sigma_{1ij} + \Sigma_{2ij}}{2} \sqrt{\frac{\Sigma_{1ij}}{\Sigma_{2ij}}} \right] \tag{4.12}
\]

where \( M_{ij}, \Sigma_{ij} \) and the constant \( \alpha \) are defined in (4.2), (4.3) and (4.5) respectively. The subscript \( w \) is added to \( \mu_1 \) and \( \mu_2 \) to distinguish them from the original ones. The new Bhattacharyya distance becomes \( \mu_w = \mu_{1w} + \mu_{2w} \), and the new Bhattacharyya bound becomes \( c_{Uw} = 0.5 \exp(-\mu_w) \). Note that this time the constant \( \alpha \) in (4.11) and (4.12) plays an important role in the estimated Bhattacharyya bound because \( \alpha \) is inside the exponential function. It is very difficult to verify theoretically whether the new Bhattacharyya distance defined in (4.11) and (4.12) is still an upper bound for the Bayes error for data with Gaussian distribution. Therefore, we do not intend to investigate this issue. Instead, we would like to see whether the Bhattacharyya distance by window-shifting can serve as a more robust measure for the separability between 2 distributions. Applying (4.11) and
(4.12) to estimate the Bhattacharyya distance for the Radar-32 and Radar-64 data, we obtain the results shown in Fig. 4.4. Note that the curves in Fig. 4.4 simply connect those discrete points on the graph to show the tendency of the window-shifting effect with respect to the sample size. No linear extrapolation is performed in these figures.

(a). Radar-32 data. (b). Radar-64 data.

![Graphs showing the effect of window-shifting on the Bhattacharyya distance for Radar-32 and Radar-64 data.](image)

Fig. 4.4. Effect of window-shifting on the Bhattacharyya distance for (a). Radar-32 data; (b). Radar-64 data.

It is clearly seen in Fig. 4.4 that as the sample size decreases, the estimated Bhattacharyya distance with the full covariance increases most rapidly. When \( w=1 \), the curve is almost flat for all sample sizes; while the rate of increase for \( w=3 \) is between the ones of \( w=1 \) and the full covariance. This is somewhat expected because the \( n \)-dimensional problem is downsized to a \( w \)-dimensional one (for \( 1 \)-dimensional vectors). For small windows like \( w=3 \), 100 samples suffice to give an accurate estimate for the \( 3 \)-dimensional mean and covariance matrix. Therefore, for small and large sample sizes, the estimated Bhattacharyya distance stays almost the same. We conjecture that the combination of window-shifting with small window sizes and linear projection might help to get a more reliable number for the Bhattacharyya distance if the sample size is very...
(a). Radar-32 data.  
(b). Radar-64 data.

Fig. 4.5. Estimation of the Bhattacharyya distance by window-shifting and extrapolation for (a). Radar-32 data; (b). Radar-64 data.

small. Therefore, we perform linear extrapolation with window shifting using $N=50$ and 100 for the Radar-32 data, and $N=100$ and 200 for the Radar-64 data. The results are shown in Fig. 4.5.

We see in Fig. 4.5 that the Bhattacharyya distance with $w=1$ is constantly overestimated for both data sets. Nevertheless, it does not generate a ridiculous value like a negative number for the Radar-64 data using the full covariance. In comparison between Fig.’s 4.3 and 4.5, we find that $w=3$ constantly generates a very good estimation of the Bhattacharyya distance. Since the Bhattacharyya distance is only a measure of separability between two distributions, we only need to know roughly how far apart these two classes are located with respect to each other. Therefore, we can accept the result estimated by $w=3$ as long as the estimate is robust even for extremely small sample sizes. As the dimension gets large, the results estimated by the full covariance may become very unstable and unreliable even though the covariance matrix is still invertible. Another concern is whether we can find a way to estimate the Bhattacharyya distance for the case $N<n$ where the covariance is not invertible and the determinant of the covariance matrix
can not be computed. The window-shifting method may offer a feasible solution to this problem. Now let us apply this method to the ISAR data to see if this method can give a reliable estimate of the Bhattacharyya distance.

For 2-dimensional images, the n-dimensional problem is downsized to the \( w^2 \)-dimensional problem using the window size \( w \). We try to perform linear extrapolation using small window sizes to obtain a more robust estimate for the Bhattacharyya distance using a small number of samples.

For the ISAR data with \( n=5\times8 \), the covariance matrix is invertible for \( N=246 \) and 123. In Fig. 4.6(a), we plot the linear extrapolation estimated by window sizes \( w=1\sim3 \) and the full covariance. All four of these lines predict the asymptotic Bhattacharyya distance at around 2. The corresponding Bhattacharyya bound \( \varepsilon_u \) is about 6.8%. Based on our analysis of this data, we believe that 2 is a reliable value for the Bhattacharyya distance. For the ISAR data with \( n=10\times16 \), the covariance matrix is only invertible for \( N=246 \). We tried to use another sample size \( N=183 \) so that the covariance matrix is invertible. However, the projected Bhattacharyya distance based on these two points is -19.58. It is completely useless; thus, it is not plotted in Fig. 4.6(b). Therefore, we use only three small window sizes, \( w=1\sim3 \), to estimate the Bhattacharyya distance. In Fig. 4.6(b), we see that these three lines project to almost the same value of about 6. The corresponding Bhattacharyya bound \( \varepsilon_u \) is about 0.12%. For the ISAR data with \( n=20\times32 \), the covariance matrix is not invertible for the sample size \( N=246 \). We must completely rely on the window-shifting to estimate the Bhattacharyya distance. The linear projection is performed in Fig. 4.6(c). From our experience we know that \( w=1 \) overestimates the Bhattacharyya distance; therefore, we prefer the results estimated by \( w=2 \) and 3 which predict the asymptotic Bhattacharyya distance to be around 13 and the corresponding Bhattacharyya bound \( \varepsilon_u \) is about \( 1.1 \times 10^{-4} \). Although the estimated Bhattacharyya distance using the window-shifting method only gives a rough measure on the separability of two distributions, our experimental results suggest that the ISAR data with \( n=10\times16 \) and \( n=20\times32 \) should be very separable. The Bhattacharyya bound, \( \varepsilon_u \), is virtually zero.
In conclusion, if the sample size is very small, the Bhattacharyya distance estimated using the full covariance is unreliable, particularly for large dimensionality $n$. The projection made using the full covariance may lead to an erroneous result as seen in Fig. 4.5(b). The window-shifting method offers a reliable way to estimate the
Bhattacharyya distance even for the case $N<n$. The projection using small window sizes requires only a small number of samples; therefore, it offers a robust solution to estimating the Bhattacharyya distance with small sample sizes.

### 4.4. Outlier’s Effect on The Bhattacharyya Distance - A Side Issue

It has been observed that there exist some outliers in the ISAR data. These outliers are quite different from most of the other samples in their class in the sense of images observed. We expect that the SAR data will also contain outliers caused by misalignment or radar noise. We are afraid that the Bhattacharyya distance might be distorted by the outliers. Thus, we believe that it is important to run experiments to have a better understanding of this issue.

It is generally believed that outliers may distort various estimated parameters and thus cause extra errors and deteriorate the system performance. In the previous section we have studied the effect of window-shifting on the Bhattacharyya distance. We would like to explore the outliers’ effect on the Bhattacharyya distance (without window-shifting) in this section to make our study more complete.

We choose the I-I and 1-41 data (Appendix A) for experiments because they represent mean-separation and covariance-separation respectively. These two data sets should be able to provide us useful information about the outliers' effect on the Bhattacharyya distance. We generate $N$ samples for each class and then add one outlier to the class 1 distribution. First, we compute the sample mean $\hat{\bf M}$ and sample covariance $\hat{\Sigma}$ of the distribution. For each class, we then compute the distance between each sample and the mean vector. The computed distances are $N$ scalars. We then compute the mean $m$ and the variance $\sigma^2$ of these $N$ distances. The outlier $X_i$ is said to be “go” away from the class mean if

$$g = \frac{1}{\sigma^2} \left| (X - \hat{\bf M})^T \hat{\Sigma}^{-1} (X - \hat{\bf M}) - m \right|$$  \hspace{1cm} (4.13)
The larger the value $g$ is, the farther the outlier is away from the class mean. The outlier is generated by multiplying a random vector with a scalar. Adjusting the magnitude of the scalar, we can obtain an outlier with the desired $g_0$ distance.

The first test data, $I-I$ data, is mean-separable with $\mu_1=0.8192$ and $\mu_2=0$. Based on 50 samples, the measured Bhattacharyya distances are $\hat{\mu}_1=0.9862$ and $\hat{\mu}_2=0.1934$. Adjusting the outlier distance $g$, we can calculate the amount of change in the estimated Bhattacharyya distance. The experimental results for the $I-I$ data are shown in Table 4.3. On the other hand, the second test data, $I-41$ data, is covariance-separable with $\mu_1=0$ and $\mu_2=0.8926$ and the measured Bhattacharyya distances are $\hat{\mu}_1=0.0378$ and $\hat{\mu}_2=1.2108$ based on 50 samples. The amount of change in the estimated Bhattacharyya distance with various outlier distance $g$ is shown in Table 4.4. We find out from these two experiments that an added outlier has a very trivial effect on the estimated Bhattacharyya distance. Take the $I-I$ data in Table 4.3, for example. Even though a large outlier ($64\sigma$ distance) is added to a mere 50 samples, the Bhattacharyya distance is only changed from 1.1796 to 1.2322. The corresponding Bhattacharyya bound is changed from 15.37% to 14.58%. As the sample size increases, the effect of one single outlier will get smaller. Therefore, we believe that the outlier's effect on the Bhattacharyya distance is negligible in practice unless the outliers are extraordinarily large and the sample size is very small.

Table 4.3. Outlier's effect on the Bhattacharyya distance for 8-dimensional $I-I$ data, $N=50$.

<table>
<thead>
<tr>
<th>outlier distance ($g$)</th>
<th>$\Delta \mu_1$</th>
<th>$\Delta \mu_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>-0.0131</td>
<td>-0.0070</td>
</tr>
<tr>
<td>16</td>
<td>-0.0183</td>
<td>8.49x10^{-5}</td>
</tr>
<tr>
<td>32</td>
<td>-0.0239</td>
<td>0.0253</td>
</tr>
<tr>
<td>64</td>
<td>-0.0283</td>
<td>0.0809</td>
</tr>
</tbody>
</table>
Table 4.4. Outlier's effect on the Bhattacharyya distance for 8-dimensional I-I data, N=50.

<table>
<thead>
<tr>
<th>outlier distance (g)</th>
<th>Δμ₁</th>
<th>Δμ₂</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>0.0013</td>
<td>-0.0556</td>
</tr>
<tr>
<td>16</td>
<td>0.0018</td>
<td>-0.0894</td>
</tr>
<tr>
<td>32</td>
<td>0.0025</td>
<td>-0.1252</td>
</tr>
<tr>
<td>64</td>
<td>0.0034</td>
<td>-0.1461</td>
</tr>
</tbody>
</table>

In the previous experiments, the dimension n of the test data is very low (=8). It may be questioned whether the outlier's effect will become severe as the data dimension gets large. This is a very practical issue because we need to estimate the Bhattacharyya distance for large dimensional data like SAR or ISAR data with a certain number of outliers. Therefore, we would like to explore this topic further. The 1.28-dimensional I-I data have the same separability as the 8-dimensional I-I data, and we choose it for experiments. The experimental results are shown in Table 4.5. In order to make the covariance matrices invertible, we had to choose a larger sample size (N=200) than the one of the previous case. We see once again that the outlier has a very trivial effect on the Bhattacharyya distance. Experimentally, we confirm that the outlier's effect on the Bhattacharyya distance does not get severe as data dimension grows. Therefore, we believe that outliers have only a very slight effect on the estimated Bhattacharyya distance regardless of the data dimension and can be neglected in real applications.

4.5. Sample Correlation and Quadratic Classifiers

It is easy to generate uncorrelated artificial data of any size for analysis. However, in the real world acquiring a large number of uncorrelated samples may be very difficult and expensive. For example, the ISAR data, in matrix form of size 20x32, are generated
Table 4.5. Outlier’s effect on the Bhattacharyya distance for 128-dimensional I-I data, $N$=200.

<table>
<thead>
<tr>
<th>outlier distance (g)</th>
<th>$\Delta \mu_1$</th>
<th>$\Delta \mu_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>-0.0073</td>
<td>-0.0651</td>
</tr>
<tr>
<td>16</td>
<td>-0.0089</td>
<td>-0.0724</td>
</tr>
<tr>
<td>32</td>
<td>-0.0115</td>
<td>-0.0764</td>
</tr>
<tr>
<td>64</td>
<td>-0.0145</td>
<td>-0.0655</td>
</tr>
</tbody>
</table>

From High Resolution Range-profile (HRR) data vectors by nonlinear transformation (Appendix A). Every pair of adjacent two images is highly correlated because they share 19 HRR data as mentioned in Appendix A. Picking up every 20th image from these 4930 correlated ISAR images, we obtain only 246 uncorrelated images. It is extremely difficult to calculate the quadratic classifier error for a 640-dimensional system using merely 246 samples. This problem could be overcome if we could fully utilize these 4930 correlated samples. Our task in this section is to verify whether or not correlated samples will cause us to overestimate or underestimate the quadratic error of the ISAR data. It is believed that the neighboring samples of the SAR data may also be correlated in a similar way as the ISAR data. Therefore, we decide to study this issue and to build up the necessary understanding of the effect of sample correlation for the ISAR and SAR data.

In order to generate correlated samples for our study, we use the following linear procedure. As seen in ISAR data, this procedure may not resemble how the samples of the ISAR data are correlated, but we hope that this procedure gives us a simple way to study this topic. First, we generate independent X samples with normal distribution $N(M_X, \Sigma_X)$. Then $Y$'s and $Z$'s are generated as $Y_1 = (X_1 + X_2 + \cdots + X_c) / c$, $Y_2 = (X_2 + X_3 + \cdots + X_{c+1}) / c$, and so forth, while $Z_1 = (X_1 + X_2 + \cdots + X_c) / c$, $Z_2 = (X_{c+1} + X_{c+2} + \cdots + X_{2c}) / c$, and so on. There is a certain degree of correlation between $Y_1$ and $Y_2$ because they share $(c-1)$ samples, $X_2$ through $X_c$. $Y_1$ and $Y_2$ tend to be neighbors in the Y-space. The spatial locality increases as $c$ increases. The scalar $c$ is
called a correlation index. On the other hand, there are no overlapping $X$'s between $Z_1$ and $Z_2$, and thus the 2's are uncorrelated.

Let us examine the moments up to the second order for the $Y$'s and 2's. All the $X$'s are uncorrelated; therefore,

$$E\{Y\} = E\{Z\} = M_X$$  \hspace{1cm} (4.14)

and

$$\text{Cov}\{Y\} = \text{Cov}\{Z\} = \frac{1}{c} \Sigma_X$$  \hspace{1cm} (4.15)

where $\text{Cov}\{Y\}$ is the covariance of $Y$. $Y$ and $Z$ samples have exactly the same first and second order moments. However, taking a closer look at the distribution, we find that these moments do not reveal the whole story and a more detailed study is needed. As mentioned shortly before, the adjacently indexed $Y$'s tend to be neighbors because they have a certain number of $X$'s in common. In order to visualize this, we generate 2-dimensional $Y$'s and $Z$'s with various correlation indices $c$. Fig. 4.7(a) shows the distribution of one set of $X$'s which are generated from $N(0,1)$. Fig. 4.7(b)-(d) show that, as $c$ increases, the distribution of the corresponding $Y$'s gets more textured, while the distribution of the $Z$'s is still very close to $\Sigma_X/c$. After repeated trials, we find that, although the covariance of $Y$ for each trial may not be close to the theoretically expected distribution $\Sigma_X/c$, the average of the covariance matrices from many trials is very close to $\Sigma_X/c$. Since the $Y$'s and $Z$'s have expected covariance of $\Sigma_X/c$, we generate $X$'s with covariance $c \Sigma_X$ in order to have $Y$'s and $Z$'s with covariance $\Sigma_X$.

Since for a large correlation index $c$, the $Y$'s get distinctive textures in the distribution, the $Y$'s are more classifiable within the same set than the $Z$'s. It translates into a lower $R$ error for $Y$'s than for $Z$'s. On the other hand, another set of independently generated $Y$'s may have a quite different texture compared to the previous set (design set). Therefore, the quadratic classifier designed by the design set may cause a larger error for $Y$'s than it does for $Z$'s. In this case, a larger hold-out (H) error is expected for $Y$'s than distribution as shown in Fig. 4.8(a). If the samples are correlated, each trial produces different covariances such as the ones in Fig. 4.8(b) and (c). The vertical line in Fig. 4.8(a)
indicates the decision boundary for the uncorrelated samples. For the distribution in Fig. 4.8(b), we can find a decision boundary based on this design set and the resulting classification error, the R error, is smaller than the one in Fig. 4.8(a). However, using the classifier designed by the samples in Fig. 4.8(b) to classify the samples in Fig. 4.8(c), we get a classification error, the H error, larger than the one in Fig. 4.8(a). Therefore, we see
that correlated samples will generate a smaller R error and a larger H error than those of the uncorrelated samples.

As we observed, correlated samples tend to have spatial locality. Therefore, if the samples in the design set are not independent of the samples in the test set, the estimated L or H error will not be a proper upper bound for the quadratic error. In real applications, we may not have independent design and test sets for analysis. It is a concern whether the presence of correlation between the design and test sets will affect the estimated quadratic error. Therefore, we have to run experiments to verify our conjectures.

Since $Y$s and $Z$'s have the same mean and covariance as in (4.14) and (4.15), we use uncorrelated $Z$'s as the reference data for comparison with the experimental result of the correlated $Y$s. We use the standard data sets (Appendix A) for experiments. First, we would like to see what kind of effect we will get if the design set is not independent of the test set. When the $Y$s are generated, we use the odd-number indexed samples as the design set and the even-number indexed samples as the test set. Since the adjacently indexed $Y$s are correlated, the design set is not independent of the test set and the samples within their own set are correlated. The experimental results are shown in Tables 4.6 and 4.7. Note that for $c=2$, samples in the design and test sets are uncorrelated, but the design set is not independent of the test set.

We see in Table 4.6 that as the correlation index $c$ increases, the estimated quadratic error bounds go down. For comparison, Table 4.7 shows that the quadratic
error bounds are not underestimated because all the $Z$s are uncorrelated. The explanation for this effect is quite intuitive, as we have mentioned before that the correlated samples tend to be local neighbors. For a large correlation index $c$, the samples; in the test set are almost identical in spatial position with the samples in the design set if these two sets are not independent; therefore, the L or H error is close to the R error. In other words, the estimated upper bound will be very close to the lower bound, the R error. Since a highly textured distribution provides extra classification information, especially in the covariance term of the classifier, the R error goes down as $c$ increases. Therefore, a lower R error is expected for larger $c$. Based on the factors mentioned above, both the estimated lower and upper bounds for the quadratic error will go down as the correlation index $c$ increases.

The conclusion we draw from this experiment is that we have to make sure that the design and test sets are independent in order to estimate the quadratic error correctly. If these two sets are not independent, a higher sample correlation index translates into a lower estimated quadratic error which is far below its actual value.

<table>
<thead>
<tr>
<th>$c$</th>
<th>I-I data $\varepsilon_0 = 10%$ R error (%)</th>
<th>H error (%)</th>
<th>I-41 data $\varepsilon_0 = 9%$ R error (%)</th>
<th>H error (%)</th>
<th>I-A data $\varepsilon_0 = 1.9%$ R error (%)</th>
<th>H error (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>7.54</td>
<td>10.60</td>
<td>6.60</td>
<td>9.59</td>
<td>1.20</td>
<td>1.88</td>
</tr>
<tr>
<td>4</td>
<td>6.21</td>
<td>7.87</td>
<td>5.83</td>
<td>7.77</td>
<td>0.87</td>
<td>1.60</td>
</tr>
<tr>
<td>6</td>
<td>5.98</td>
<td>7.37</td>
<td>5.24</td>
<td>6.54</td>
<td>0.87</td>
<td>1.20</td>
</tr>
<tr>
<td>8</td>
<td>5.18</td>
<td>6.38</td>
<td>4.81</td>
<td>6.18</td>
<td>0.74</td>
<td>1.12</td>
</tr>
<tr>
<td>10</td>
<td>4.29</td>
<td>5.33</td>
<td>3.79</td>
<td>4.78</td>
<td>0.56</td>
<td>0.88</td>
</tr>
<tr>
<td>20</td>
<td>2.60</td>
<td>3.27</td>
<td>2.05</td>
<td>2.69</td>
<td>0.16</td>
<td>0.32</td>
</tr>
</tbody>
</table>

Table 4.6. The R and H errors of the correlated samples $Y$s with dependent design and test sets, $N=200$. 


Next, we investigate the case where the design set is independent of the test set. Since the data are computer generated, the design and test data are generated independently, and the quadratic errors are computed by both the R and H methods. The results are plotted in Fig.'s 4.9 through 4.11 for various sample sizes N. As the correlation index c increases, more samples are required for the estimated error rates to converge to the quadratic error. This is similar to the trend observed in estimation problems where correlated samples reduce the effective sample size. That is, in order to achieve the same variances in estimation, more correlated samples are required than uncorrelated samples.

We also see that if the design and test sets are independently generated, the estimated R and H errors can serve as the lower and upper bounds of the quadratic error properly even though the samples are correlated. Note that in Fig.'s 4.9 - 4.11 the case c=1 corresponds to uncorrelated Z samples. The horizontal lines in these three figures represent the quadratic error of the test data.

Looking at Fig.'s 4.9 - 4.11, we wonder whether there is a connection between the correlation index c and the sample size N. To have a clearer picture of how these two variables relate to each other, we plot in Fig. 4.12 the required N for I-I data to obtain $\epsilon_R = 12\%$ and $\epsilon_H = 7\%$ for various c's. The same procedure is repeated for the I-4I data and I-A data, and the results are plotted in Fig.'s 4.13 and 4.14 respectively.

<table>
<thead>
<tr>
<th>c</th>
<th>I-I data $\epsilon_Q = 10%$</th>
<th>I-4I data $\epsilon_Q = 9%$</th>
<th>I-A data $\epsilon_Q = 1.9%$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>R error (%)</td>
<td>H error (%)</td>
<td>R error (%)</td>
</tr>
<tr>
<td>2</td>
<td>7.61</td>
<td>12.25</td>
<td>6.45</td>
</tr>
<tr>
<td>4</td>
<td>7.67</td>
<td>11.90</td>
<td>6.66</td>
</tr>
<tr>
<td>6</td>
<td>7.51</td>
<td>12.29</td>
<td>6.77</td>
</tr>
<tr>
<td>8</td>
<td>7.55</td>
<td>12.18</td>
<td>6.93</td>
</tr>
<tr>
<td>10</td>
<td>7.80</td>
<td>12.31</td>
<td>6.62</td>
</tr>
<tr>
<td>20</td>
<td>7.65</td>
<td>12.24</td>
<td>6.79</td>
</tr>
</tbody>
</table>
Fig. 4.9. Quadratic errors of the correlated 1-1 data with independent design and test sets.

Fig. 4.10. Quadratic errors of the correlated 1-41 data with independent design and test sets.
Fig. 4.11. Quadratic errors of the correlated I-A data with independent design and test sets.

Fig. 4.12. Sample size vs. correlation index for correlated I-I data.
Fig. 4.13. Sample size vs. correlation index for correlated I-41 data.

Fig. 4.14. Sample size vs. correlation index for correlated I-A data.
We see that these discrete points in Fig.'s 4.12-4.14 can be fitted by straight lines for given error rates. Let us take the I-I data in Fig. 4.12 as an example. If we want the estimated $R$ error to converge to the given error of 7%, we need about 200 samples with a correlation index $c=4$, while 300 samples with correlation index $c=6$ are required to converge to the same $R$ error of 7%. This means that in order to converge the given error rate, the required sample size is linearly proportional to the correlation index $c$. Let $\varepsilon_R(N)$ and $\varepsilon_H(N)$ be the estimated $R$ and $H$ errors of a quadratic classifier using uncorrelated $N$ samples. For correlated samples with sample size $N_1$ and $N_2$,

$$\varepsilon_R(N_1/c_1) \approx \varepsilon_R(N_2/c_2) \quad \text{and} \quad \varepsilon_H(N_1/c_1) \approx \varepsilon_H(N_2/c_2) \quad (4.16)$$

where $c_1$ and $c_2$ are the corresponding correlation indices.

The sample correlation will not mislead us into underestimating the quadratic error of the given data set as long as the independency of the design and test sets is guaranteed. The higher the correlation between samples, the looser the error bounds will be. This effect is also intuitively clear, since the effective uncorrelated sample size of a correlated data set with sample size $N$ and correlation index $c$ is roughly $N/c$. Although in our procedure correlations are introduced by a linear operation which may be different from the operations in real data, the real data with correlated samples seem to behave similarly.

Let us examine the Radar-32 data based on the finding of this section. According to the data provider, the adjacent samples for one class are correlated, but the samples from the other class are not. Detailed information about how the samples are correlated is not available. Although the structure of the real data is more complex than the controlled data of the previous discussion, we would like to see if the previous finding can help to find out the degree of sample correlation. We believe that samples with non-consecutive indices are uncorrelated; therefore, we can form independent design and test sets with uncorrelated samples in each set. For example, we pick up samples indexed 1, 21, 41, ..., as the design set and samples indexed 11, 31, 51, ..., as the test set. For comparison, we form independent design and test sets with the correlated samples in each set. For example, we can put samples indexed from 1 though 50 in the design set, then put samples indexed from 101 through 150 in the test set, etc. By adjusting the index spacing between
the design and test sets, we can control the sample size of both sets. The experimental results are shown in Fig. 4.15(a). The horizontal line in Fig. 4.15(a) indicates the quadratic error of the Radar-32 data. The solid curves represent the error rates estimated by uncorrelated samples, while the dashed curves stand for the error rates estimated by correlated samples. Without knowing the sample correlation index of the Radar-32 data, we only use up to 600 independent samples for experiments. We are afraid of getting slightly correlated samples if more samples are used. As seen in Fig. 4.15(a) that the Radar-32 data set indeed contains correlated samples. For the same number of samples, the H error estimated by correlated samples is consistently higher than the one estimated by uncorrelated samples. On the other hand, the R error estimated by correlated samples is much lower than the one estimated by uncorrelated samples for the same number of samples. The equivalent correlated index obtained from the R error curves is about 6, while the one estimated by the H error curves is only slightly larger than 1. We could not come up with one clean number for the correlation index of this data. However, we have verified that this data set consists of correlated samples. For the Radar-64 data, the provider believes that adjacent samples should be uncorrelated. We perform the same experiment on the Radar-64 data and obtain the results in Fig. 4.15(b). The R errors estimated by correlated and uncorrelated samples are very close. From the H error curves, the equivalent correlation index is estimated to be about 1.5. As with the Radar-32 data, we can not come up with a number for the correlation index of this data. However, we believe that this data have only slight correlation between adjacent samples. From these two experiments, we believe that this technique offers another way of detecting the presence of sample correlation of a data set.

For the ISAR data, we have correlated Y's with \( N_r = 4930 \) and uncorrelated Z's with \( N_f = 246 \). If we try to use these uncorrelated Z's to estimate the quadratic error, we will come up with very loose bounds; however, the quadratic error will be bounded properly. If we want to take advantage of the large correlated set Y, we have to make sure that the design and test sets are independent. Otherwise, we will underestimate the quadratic error, although the upper and lower bounds are very tight. Since the Y's contain 360 degrees of
viewing angles, we can not simply assign the samples indexed 1 through 2455 for the design set and the samples indexed 2476 though 4930 for the test set, even though the design set becomes indeed independent of the test set. Instead, a feasible way would be, for example, to assign the samples indexed 1 through 100 to be in the (design set, samples indexed 121 through 220 to be in the test set, samples indexed 241 through 340 to be in the design set, samples indexed 361 through 460 to be in the design set and so forth. The samples indexed 101 through 120 can not be included in either set as a safeguard for independency between these two sets. In this way, the samples in the design and test sets are more representative of the 360-degree view of the object, and also the independency of the design and tests set is achieved. Adjacent samples in the design and test sets share 19 HRR data. Therefore, the samples within their own set are correlated. In this example, the number of design samples, approximately 2,000, is equal to the number of the test samples. About 900 samples are discarded to guarantee independency between the design and test sets. With 2,000 samples in each set, we believe that the quadratic error will be more closely bounded than when using the 246 uncorrelated samples, particularly for
To verify this, we perform quadratic classification on these two sets. Note that if all the samples are uncorrelated, we can apply the L method to fully utilize the available samples as long as the covariance matrix is invertible. Therefore, we can apply the L method to the 246 uncorrelated samples for \( n=5 \times 8 \) and \( n=10 \times 16 \). The results are given in Table 4.8. We see that the correlated samples do help to offer tighter bounds on the quadratic error except for \( n=10 \times 16 \). However, without using the correlated samples, the covariance matrices of the ISAR data with \( n=20 \times 32 \) are singular and the lower and the upper bounds cannot be determined. Using a large number of correlated samples, the covariance matrices are no longer singular and the quadratic classification can be performed. In the next section, we will introduce the window-shifting method which is even more powerful than employing correlated samples to get tighter bounds on the quadratic error.

<table>
<thead>
<tr>
<th>( n )</th>
<th>correlated samples, ( N=2000 )</th>
<th>uncorrelated samples, ( N=246 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( 5 \times 8 )</td>
<td>1.13</td>
<td>5.99</td>
</tr>
<tr>
<td>( 10 \times 16 )</td>
<td>0</td>
<td>8.30</td>
</tr>
<tr>
<td>( 20 \times 32 )</td>
<td>0</td>
<td>11.79</td>
</tr>
</tbody>
</table>

**Table 4.8.** Comparison of R and H errors with correlated and uncorrelated samples for various dimensions of ISAR data.

4.6. Quadratic Classifier with Window-shifting

In this section, the window-shifting method is applied to design a quadratic classifier. To estimate the lower bound of the quadratic error, we have to use the R procedure to evaluate the classification error. Using all the samples to compute the mean and covariance and testing the same design set, we can compute the distances by summing
up the sub-distances. The discriminant function for the sample $Q_i^{(k)}$, $I = 1, \ldots, N$ and $k=1,2$, can be obtained by

$$h_R(Q_i^{(k)}) = d_1^2(Q_i^{(k)}) - d_2^2(Q_i^{(k)})$$

(4.17)

After all the samples have been tested, we get two sequences of discriminant functions, namely, $h_R(Q_i^{(1)})$ and $h_R(Q_i^{(2)})$, $I = 1, 2, \ldots, N$. The smallest classification error of these two sequences can be found by adjusting the decision threshold $t$. Therefore, the error count is incremented if $h_R(Q_i^{(1)}) \geq t$ or $h_R(Q_i^{(2)}) < t$, $I = 1, 2, \ldots, N$. This decision threshold $t$ is retained for evaluating the H or L error.

The upper bound of the quadratic error can be obtained by the L or H procedure. In real applications, the situation is often encountered where the number of samples is so limited that we cannot afford to divide these samples into two sets, the design and test sets. If there are a total of $N$ samples available, the H procedure requires $N/2$ samples for classifier design and the other $N/2$ samples for testing. The effective sample size in both the R and H procedures is only $N/2$. Therefore, loose error bounds are expected. Based on this concern, we will opt for the L procedure to estimate the quadratic error. The effective sample size of the R and L procedures is $N$ rather than $N/2$. Thus, tighter bounds can be achieved. The L procedure for the window-shifting distance is similar to the conventional L procedure for ordinary quadratic classifiers. The only difference is that for the window-shifting L procedure we have to modify the discriminant function obtained by the R method for the pixels covered by the window.

Let $M_y^{(k)}$ and $\Sigma_y^{(k)}$ be the mean and covariance of the class $k$ pixels covered by the window and $G_y^{(k)}$ be the vector obtained from the pixels covered by the window. Now let us look at each window area. Removing the $Z$th sample $G_y^{(k)}(Z)$, the perturbation equation [21] can be obtained by

$$g_y(G_y^{(k)}(I)) = \frac{N^2 - 3N + 1}{N - 1} d_y^{(k)2}(G_y^{(k)}(I)) + \frac{Nd_y^{(k)4}(G_y^{(k)}(I))}{(N - 1)^2 - Nd_y^{(k)2}(G_y^{(k)}(I))}$$
\[ +w^2 \ln \left[ 1 + \frac{1}{N-2} \right] + \ln \left[ 1 - \frac{N}{(N-1)^2} d_y^{(k)} (G_y^{(k)}(I)) \right] \]  

(4.18)

where

\[ d_y^{(k)} (G_y^{(k)}(I)) = (G_y^{(k)}(I) - M_y^{(k)})^T \Sigma_y^{(k)-1} (G_y^{(k)}(I) - M_y^{(k)}) \]  

(4.19)

Therefore, the discriminant function of the L method can be obtained by

\[ h_L(Q_t^{(k)}) = h_R(Q_t^{(k)}) \pm \sum_{i=1}^{s} \sum_{j=1}^{s} g_y (G_y^{(k)}(I)) \]  

(4.20)

where the plus sign is for \( \omega_1 \) and the minus sign is for \( \omega_2 \). It can be shown [21] that the perturbation equation in (4.18) is always nonnegative as long as the window covariance is nonsingular.

For the 1-dimensional data vectors, a rectangular window of size \( wxl \) is shifted over the sample vectors to compute distances. As the window size \( w \) increases, more pixel correlations are involved in distance computation. The correlation among \( w \) neighboring pixels can be easily mapped to \( wxw \) sub-matrices along the diagonal axis of the covariance matrix. Therefore, we would like to examine the feasibility of the window-shifting method to the 1-dimensional vectors before moving on to the 2-dimensional images.

Let us use the Radar-32 and Radar-64 data sets for experiments. For the Radar-32 data, the quadratic error is estimated to be about 14%. Picking up 400 samples from each class and performing quadratic classification with window-shifting, we obtain the error rates shown in Table 4.9. As we have expected, if we use only the variance information (\( w=1 \)), we can get very tight bounds, but some classification information is lost due to such a crude approximation. Therefore, both the R and L errors are higher than the actual quadratic error, 14%. As window size increases, we get looser bounds on the quadratic error, but the quadratic error is bounded properly. The Radar-64 data behave similarly as shown in Table 4.10.

Now let us perform quadratic classification with window-shifting distance measurement on the ISAR data to see how much our approach improves the performance.
Table 4.9. Quadratic classification errors for the Radar-32 data with window-shifting, \( N=400 \).

<table>
<thead>
<tr>
<th>window size w</th>
<th>R error (%)</th>
<th>L error (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>18.72</td>
<td>19.72</td>
</tr>
<tr>
<td>2</td>
<td>16.08</td>
<td>17.71</td>
</tr>
<tr>
<td>3</td>
<td>15.95</td>
<td>16.46</td>
</tr>
<tr>
<td>5</td>
<td>14.32</td>
<td>15.83</td>
</tr>
<tr>
<td>10</td>
<td>13.94</td>
<td>16.83</td>
</tr>
<tr>
<td>32 (full cov.)</td>
<td>9.92</td>
<td>16.08</td>
</tr>
</tbody>
</table>

Table 4.10. Quadratic classification errors for the Radar-64 data with window-shifting, \( N=400 \).

<table>
<thead>
<tr>
<th>window size w</th>
<th>R error (%)</th>
<th>L error (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>27.38</td>
<td>28.50</td>
</tr>
<tr>
<td>2</td>
<td>26.75</td>
<td>28.25</td>
</tr>
<tr>
<td>3</td>
<td>26.63</td>
<td>28.13</td>
</tr>
<tr>
<td>5</td>
<td>25.13</td>
<td>28.38</td>
</tr>
<tr>
<td>10</td>
<td>23.60</td>
<td>29.63</td>
</tr>
<tr>
<td>30</td>
<td>11.88</td>
<td>30.63</td>
</tr>
<tr>
<td>64 (full cov.)</td>
<td>3.75</td>
<td>28.00</td>
</tr>
</tbody>
</table>

The window-shifting method does not require a large sample size, and thus the use of a large number of correlated samples is less effective. Therefore, we use here only 246 uncorrelated samples to estimate the quadratic error. Note that, for \( N=246 \), both the covariance matrices of the 5x8 and 10x16 ISAR data are invertible; however, for the 20x32 ISAR data, the covariance matrix of size 640x640 is singular. The advantage of using window-shifting distance measurement is particularly obvious for the 20x32 ISAR data. The experimental results are shown in Tables 4.11 - 4.13.
Table 4.11. Quadratic classification errors for the ISAR data, \( n = 5 \times 8, N = 246 \).

<table>
<thead>
<tr>
<th>window size w</th>
<th>R error (%)</th>
<th>L error (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>7.89</td>
<td>9.92</td>
</tr>
<tr>
<td>2</td>
<td>6.28</td>
<td>9.11</td>
</tr>
<tr>
<td>3</td>
<td>5.06</td>
<td>8.10</td>
</tr>
<tr>
<td>4</td>
<td>3.44</td>
<td>5.87</td>
</tr>
<tr>
<td>5</td>
<td>2.63</td>
<td>5.67</td>
</tr>
<tr>
<td>full cov.</td>
<td>1.62</td>
<td>7.69</td>
</tr>
</tbody>
</table>

Table 4.12. Quadratic classification errors for the ISAR data, \( n = 10 \times 16, N = 246 \).

<table>
<thead>
<tr>
<th>window size w</th>
<th>R error (%)</th>
<th>L error (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3.64</td>
<td>5.06</td>
</tr>
<tr>
<td>2</td>
<td>1.21</td>
<td>3.04</td>
</tr>
<tr>
<td>3</td>
<td>0.61</td>
<td>3.04</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>2.63</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>2.23</td>
</tr>
<tr>
<td>full cov.</td>
<td>0</td>
<td>7.09</td>
</tr>
</tbody>
</table>

Table 4.13. Quadratic classification errors for the ISAR data, \( n = 20 \times 32, N = 246 \).

<table>
<thead>
<tr>
<th>window size w</th>
<th>R error (%)</th>
<th>L error (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.23</td>
<td>3.64</td>
</tr>
<tr>
<td>2</td>
<td>0.61</td>
<td>1.21</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>2.02</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>2.23</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>2.43</td>
</tr>
<tr>
<td>15</td>
<td>0</td>
<td>13.36</td>
</tr>
<tr>
<td>full cov.</td>
<td>cannot be determined</td>
<td>cannot be determined</td>
</tr>
</tbody>
</table>
There are two things that we are concerned with about the quadratic classifier. First, we would like to estimate the quadratic error, which would be the error rate that the lower and upper bounds converge to if there are an infinite number of samples. On the other hand, we would like to know the classification accuracy if we apply this classifier to a new set of samples of the same target. This error is the L or H error. Therefore, we would like to have the L or H error to be as low as possible. With these two criteria in mind, we go on analyzing the experimental results.

Let us examine the results in Table 4.11 first. If we only utilize the variance information of each pixel, i.e. \( w=1 \), we can get a pretty tight bias between the upper and lower bounds. However, since we give away all the inter-pixel correlation information, we get a pretty high classification error (\( L = 9.92\% \)). This is expected because the 40-dimensional problem is downsized to a 1-dimensional one. As the window size increases, both the R and L errors go down but the bounds get loose. Using the full covariance, we get very loose bounds, \( R=1.62\% \) and \( L=7.69\% \), and it is difficult to pin down a precise value for the quadratic error. However, the window-shifting approach helps us to get tighter bounds. When we use a window whose size is smaller than the full covariance, we give away some useful classification information; therefore, the estimated error rate contains a positive bias. In other words, the quadratic error is overestimated. The computed R error may not be a real lower bound for the quadratic error, but the L error is definitely above the quadratic error. In terms of mathematics, the following relation must hold

\[
L_w \geq \varepsilon_{Q_w} \geq \varepsilon_{Q}
\]  

(4.21)

where \( L_w, \varepsilon_{Q_w}, \) and \( \varepsilon_{Q} \) are the L error estimated by window-shifting, the quadratic error with a certain window size w and the quadratic error with full covariance, respectively. Thus, it is reasonable to use the lowest L error obtained with the window-shifting method as the upper bound for the quadratic error, while we must still take the R error obtained by using the full covariance as the lower bound. For the ISAR data with \( n=5 \times 8 \), the lower and upper bounds can be narrowed down to 1.62\% and 5.67\%, respectively.

As the data dimension gets larger, the advantage of using the window-shifting
distance measurement gets obvious. Let us take a look at the results shown in Table 4.12. The data dimensionality is $n = 10 \times 16 = 160$; thus, the full covariance is invertible and the L procedure can be applied. It is clear that by increasing the window size, namely, by using more inter-pixel correlation information, we can lower the error rate significantly. In this case, using a small window of the size $w = 4$ can generate pretty good classification accuracy. In the sense of quadratic error estimation, the window-shifting method also helps to narrow down the error bounds. With the R error from using full covariance and the L error from window-shifting with $w = 5$, we can pin down the quadratic error to be within 0 and 2.23% rather than 0 and 7.09% (with the full covariance).

As the data dimensionality increases to $n = 20 \times 32 = 640$, as shown in Table 4.13, the Hughes phenomenon [36] becomes severe. Due to the insufficient sample size and the large data dimensionality, the error rate increases although the pixel resolution is also increased. Note that for the full covariance with $n = 640$, these 246 samples are not enough to form an invertible covariance matrix. Therefore, we can only apply the L method and the window-shifting method with the window size $w$ up to 15. Beyond that, the window covariance becomes singular and the denominator of the first term in (4.18) becomes zero. Note that in Table 4.11, both the R and L errors keep going down as the window size increases from 1 to 5; however, in Table 4.13, we see that the L error goes down when the window size increases from 1 to 2, but after that the error rate increases. As the window size is increased to $w = 15$, the L error rises to 13.36%. When $N < n$, the covariance is singular and the inverse of the covariance does not exist. An alternative way to calculate the normalized distances is to use the pseudo inverse. Thus we can only apply the H method with pseudo inverse to get the upper bound for the quadratic error. With the help of the window-shifting method, we can pin down the quadratic error of the ISAR data with $n = 20 \times 32$ to be within 0% and 1.21%. This is much more powerful than employing correlated samples (Table 4.8).

When we apply the R and H methods to the ISAR data with $n = 20 \times 32$ to estimate the quadratic error using the pseudo inverse of the covariance as the metric, we encounter a very strange phenomenon that the R error is even higher than the H error. We find that
the interclass distance is shorter than the intraclass distance. This is contradictory to our knowledge that the expected value of most of the samples should be closer to the mean of the same class than the one of the other class. In order to understand the effect of insufficient design/test samples and the behavior of the quadratic classifier using the pseudo inverse as the distance measure, we generate various sizes of 50-dimensional I-I data and apply a quadratic classifier to estimate the quadratic error. The theoretical quadratic error of this data is known to be 10%. The experimental results are shown in Fig. 4.16.

In Fig. 4.16 we plot the $\omega_1$ and $\omega_2$ samples on the d-display [38]. The horizontal axis is the distance between samples and $M_1$ using $\Sigma_1$ as the metric, while the vertical axis is the distance between samples and $M_2$ using $\Sigma_2$ as the metric. The $\omega_1$ samples are marked with "x" and the $\omega_2$ samples are represented by "o". Since these two classes are separable with 10% of error, the $\omega_1$ samples should lie in the upper left corner of the d-display because they should be closer to the mean of their own class than the mean of the other class. Similarly, the $\omega_2$ samples should lie in the lower right corner of the d-display. What is shown here is the distance measured with the R method. The R error should be smaller than the 10% quadratic error. First we start from 400 samples and obtain the results in Fig. 4.16(a). Since the sample size of 400 is larger than the data dimension of 50, the covariance matrices are invertible and the distribution of the samples on the d-display is just as we have expected. As the sample size decreases from 400 to 100 and then to 60, the variation of the class i distance for the class i samples, $i=1,2$, gets smaller as shown in Fig. 4.16(b)-(c). When the sample size gets smaller than the data dimension, the covariance matrices of both classes become singular; therefore, a pseudo inverse has to be used as the metric. For the data with sample size N smaller than the data dimension n, the normalized distances between the samples and the class mean using a pseudo-inverse as the metric are constant. We state this as a theorem.
Fig. 4.16 (a)-(d). The d-display of the 50-dimensional I-I data with various sample size.
Fig. 4.16 (e)-(g). The d-display of the 50-dimensional I-I data with various sample size.
Theorem 4.1:

Given \( n \times 1 \) vectors \( X_i, i=1,2, \ldots, A \). Let \( \hat{M} \) and \( \hat{C} \) be the sample mean and the sample covariance. If \( N < n \), then

\[ (X_i - \hat{M})^T \hat{\Sigma}^- (X_i - \hat{M}) = \frac{(N-1)^2}{N}, \quad i=1,2,\ldots, N. \]  

(4.22)

where \( \hat{\Sigma}^- \) is the pseudo-inverse of \( \hat{C} \), obtained by

\[ \hat{\Sigma}^- = \sum_{i=1}^{N-1} \frac{1}{\lambda_i} \phi_i \phi_i^T \]  

(4.23)

where \( \lambda_i \) and \( \phi_i \) are the nonzero eigenvalues and corresponding eigenvectors of \( \hat{C} \).

Proof: see Appendix E.

Therefore, the samples on the d-display is aligned on straight lines. We see that in Fig. 4.16(d) that, for \( N=40 \), the samples are located properly although their intraclass distance is a constant. Nevertheless, as the sample size shrinks further, as shown in Fig. 4.16(e)-(g), these samples begin to move toward the direction where the intraclass distance is longer than the interclass distance. Finally, for extremely small sample size, we will obtain the R error = 100%. The conclusion of this experiment is that if the sample size is smaller than the data dimension, the quadratic classifier designed based on the pseudo inverse of the covariance is not usable. It no longer provides useful information for the upper and lower bounds of the quadratic error.

Now let us take a look at the d-display of the ISAR data with \( n=20 \times 32 \) in Fig. 4.17. The class 1 samples, Camaro, are marked with "\( \times \)" and the class 2 samples, Van, are marked by "\( \circ \". As before, the class 1 samples should lie in the upper left corner of the d-display, while the class 2 samples should lie in the lower right portion of the graph. Since the sample size \( N=246 \) is smaller than the data dimension \( n \), we can not apply the L method described in this section to obtain the upper bound for the quadratic error. Splitting the samples into design and test sets, we have 123 samples in each set. Applying the R and H methods, we obtain the R error = 47.6% and H error = 46.3%. This is similar
to what we observed from the 50-Dimensional I-I data with \( N=20 \). From our experiments with singular covariance matrices, we determine that the R and H errors we estimated are unreliable. Once again, this indicates that the window-shifting method is a very powerful algorithm to help in estimating the upper bound of the quadratic error even for data sets where the conventional methods fail to work.

(a). R method, R error = 47.6%

(b). H method, H error = 46.3%

Fig. 4.17. The d-display of the ISAR data with \( n=20 \times 32 \). (a) R method; (b) H method.

4.7 Conclusion

Parametric problems with a high \( n \) and a limited \( N \) are discussed in this chapter. We have introduced a new method, the window-shifting method, to measure the distance between samples. The development of this method is aimed mainly at large dimensional systems with insufficient samples. We have illustrated in this chapter how to perform the window-shifting method to compute distances and have analyzed the computational complexity for each distance calculation. For large dimensional systems, the inverse of the covariance matrix is very difficult to calculate even if it is invertible; however, the
window-shifting method is shown to have better performance and low computational complexity compared with those of using the full covariance of the system.

We have also demonstrated that by using the window-shifting method we are able to obtain a quick measure of separability of two distributions by computing the Bhattacharyya distance. For the ISAR data with $n=20 \times 32$, the covariance is singular and not invertible; however, by using the window-shifting method and lineal extrapolation, we can estimate the Bhattacharyya bound without the difficulty of matrix singularity because each window covariance is invertible.

An often encountered situation in pattern recognition is the estimation of the Bhattacharyya distance with the presence of outliers. It is commonly believed that these outliers have an unpleasant effect on system performance as well as on estimated parameters. The actual outlier's effect on the Bhattacharyya distance is found to be extremely small even for large outliers and for large-dimensional systems. We conclude that the outlier's effect can be neglected when estimating the Bhattacharyya distance.

Sample correlation is another important issue for the design of parametric classifiers. We have done a systematic study on this problem by assuming linear correlation between samples. We found that the samples in the design set must be completely independent of the samples in the test set in order to estimate the quadratic error. Correlation between the design and test sets will generate a false quadratic error rate lower than its actual value. The magnitude of distortion is proportional to the degree of correlation between these two sets. We have also discovered that sample correlation has a direct influence on the equivalent sample size for the quadratic classifier. Higher sample correlation indices translate into smaller equivalent sample sizes (compared to the uncorrelated samples); therefore, more samples are required to converge to a certain error rate. If we can guarantee the independency between the design and test sets, the quadratic error can be properly bounded even though the samples within their own set are correlated. This finding is very helpful to us for analyzing and processing real data sets with nonlinear sample correlations. This finding can also be applied to detect the presence of sample correlation of a data set.
We also demonstrated how to use the window-shifting method to design a classifier and estimate the quadratic error. The performance improvement by making use of more pixel correlation information is significant over the one by pixel variance alone. For large dimensional systems, not only is the covariance matrix singular, but also it is difficult to perform quadratic classification with a pseudo inverse of the covariance and with small sample size. Our proposed method helps to get a tighter upper bound for the quadratic error. We believe this approach is a feasible and promising solution to the problem encountered while processing the SAR data.

At this point we are almost ready to estimate the Bayes error of the ISAR data. The only problem left to be solved is the estimation of intrinsic dimensionality which is very important in determining the number of local samples in our algorithm. We will propose a solution in the next chapter based on the window-shifting algorithm.
CHAPTER 5
BAYES ERROR ESTIMATION OF THE ISAR DATA

5.1. Introduction

After the preliminary parametric study of the ISAR data, we would like to apply nonparametric estimation techniques to the ISAR data in this chapter. However, in order to apply the Bayes error estimation technique effectively, a set of parameters has to be properly selected. Many of these parameters depend on the intrinsic dimension of the data. So we begin the discussion of this chapter with the estimation of the intrinsic dimensionality. Estimation of the intrinsic dimensionality of a data set with a very large dimension and a very small sample size is very difficult using conventional methods. In order to overcome this difficulty, we developed a new method based on the window-shifting approach mentioned in Chapter 4. The new estimation method is applied first to data with known intrinsic dimensions to test its feasibility and then applied to the data with unknown intrinsic dimensions. This issue is pursued in Section 5.2.

In our algorithm, as we have discussed in Chapter 2, the most difficult and important step is the determination of the local metric B, which determines the way of searching for neighbors. With the presence of a large number of correlated samples in ISAR data, we would like to know if these correlated samples can help to form a better metric B to serve our purpose. This topic will be discussed in Section 5.3. Although we are ready to estimate the Bayes error of large dimensional data, we would like to propose a scheme for reducing the computation load. We demonstrate this procedure in Section
5.4. Finally, we move on to estimate the Bayes error of the ISAR data in Section 5.5. The steps of performing Bayes error estimation are summarized in Section 5.5. Conclusion is stated in Section 5.6.

5.2. Estimation of Intrinsic Dimensionality

5.2.1 Background study

The intrinsic dimensionality of a data set refers to the minimum number of parameters needed to generate this data set. When confronted by large dimensional data sets, it is usually advantageous to discover or impose some structure on the data. In our local metric estimation algorithm, the number of local samples is determined by the intrinsic dimensionality of the data. Therefore, it is important to develop an efficient and reliable method to figure out the intrinsic dimensionality of the given data set.

The initial work on this problem was done by Shepard [23], [24] and Krusal [25], [26]. They developed multidimensional scaling algorithms to build the proximity matrix based on the distance between points. Their methods provide only the linear dimensionality which is equivalent to that given by the global Karhunen-Loeve expansion. This original work is later modified by Shepard and Carroll [27] and Bennett [28]. Some references on this subject are Sammon [29], Krusal [30], Chang and Lee [31] and Krusal and Hart [32]. The idea of multidimensional scaling was explored intensively in [49],[51]. Trunk [33] proposed a statistical approach based on hypothesis testing, which looks for the most likely local dimensionality. Although this work has a substantial theoretical background, many ad hoc assumptions are required in real applications.

Several other ideas for estimating the intrinsic dimensionality have been suggested, such as indices of condensation [47], parametric mapping [48] and minimum spanning trees [50]. However, some of these algorithms end up determining the intrinsic dimensionality by the eigenvalues of a covariance matrix. Ball [46] demonstrated the perils of such an approach. Fukunaga and Olsen [34] got around the inherent problem of...
using eigenvalues of a global covariance matrix and proposed an algorithm for finding the intrinsic dimensionality of data by a local Karhunen-Loeve expansion. Although this method performed better than that of the global Karhunen-Loeve expansion, its result is strongly affected by the number of samples enclosed in the local region. A major drawback of this method is that as the data dimension gets large, it becomes more and more difficult to determine the intrinsic dimensionality by the magnitude of the eigenvalues. Applying this algorithm to the Radar-32 and Radar-64 data, we found we could not determine the intrinsic dimensionality by inspecting the magnitude of the eigenvalues.

Pettis, Bailey, Jain and Dubes\cite{35} used the near-neighbor information to estimate the intrinsic dimensionality. They started from the k-nearest neighbor (k-NN) density estimate and assumed the distribution was continuous and nonzero at the test sample. They further assumed that samples fell in the local region as a sequence of Bernoulli trials. They simplified the mathematical derivation by a Poisson approximation, assuming that samples are densely distributed. They then used an iterative process to compute the intrinsic dimensionality. Their work proved to work very well for artificial data with data dimension and intrinsic dimension no greater than three. A major concern is that for large dimensional data this iteration process may not be able to reach the pre-set error threshold. In addition, their algorithm contains a preprocessing procedure to eliminate outliers. If the data dimension is extremely large, the effectiveness of such a preprocessing step is dubious. Fukunaga and Flick \cite{22} also started from the k-NN density estimate and used the fact that the probability mass inside the local region has a beta-distribution. By measuring the average nearest neighbor distances, one can estimate the intrinsic dimensionality of the data set. This method also proved to work very well for low dimensional data like the 8-dimensional Gaussian pulse data mentioned in Appendix A.

However, all these working algorithms face a difficulty in real applications, i.e., the estimated intrinsic dimensionality is metric-independent. Both \cite{35} and \cite{22} come to the same simple conclusion that
where the inner expectation $E$ is applied to obtain the expected k-th NN distance for a given sample $X$, and the outer expectation $E_x$ is performed to average the k-th NN distances over the entire data set. Note that, for the equality "$\equiv$" to hold in (5.1), we assume that the number of samples is large enough so that certain approximations can be made during the derivation.

Before we go on examining why these theories fail to apply to the case where the dimension $n$ is very high and the sample size $N$ is limited, let us take a look at the experimental results of the Radar-64 data to get some insights into this problem. Although the measured distance depends on the metric, the terms that contain the metric can be factored out and canceled in (5.1). Therefore, the metric used for determining distance does not play a role in the estimated intrinsic dimensionality. We thus use two metrics, the global covariance $\Sigma$ obtained from the 400 test samples and the Euclidean metric $I$, to compare the differences of the estimated intrinsic dimension using (5.1). We then decrease the dimension of the sample vectors by a factor of two and repeat the same procedure to find the intrinsic dimension of the down sampled data. The results are shown in Table 5.1. For 64-dimensional data, 400 samples are sparsely distributed in the sample space. We see that there is a large discrepancy between the estimated intrinsic dimensions using different metrics. However, as data dimension gets small, the difference also gets small. When the dimension $n$ is less than 8, the differences become negligible. For fixed sample size, smaller dimensions means a denser distribution of samples in the space. Thus, we can conclude from this experiment that (5.1) holds if the data dimension is small compared to the sample size.

<table>
<thead>
<tr>
<th>metric</th>
<th>64</th>
<th>32</th>
<th>16</th>
<th>8</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Sigma$</td>
<td>32.00</td>
<td>20.38</td>
<td>13.04</td>
<td>7.15</td>
<td>4.02</td>
</tr>
<tr>
<td>$I$</td>
<td>19.89</td>
<td>16.00</td>
<td>12.05</td>
<td>7.20</td>
<td>3.98</td>
</tr>
</tbody>
</table>

Table 5.1. Metric effect on the intrinsic dimension of the Radar-64 data, $N=400$. 

\[
\frac{E_x E\{d_{(k+1)NN}(X)\}}{E_x E\{d_{kNN}(X)\}} \equiv 1 + \frac{1}{kn_x}
\]
Now let us take [22] for example to illustrate why the theory fails to estimate the intrinsic dimensionality of a data set with large \( n \) and small \( N \). In order to simplify the derivation, they approximated \( u(X) \), the probability mass that the \( k \)-th NN can be found on the boundary of the local region \( L(X) \), by its first term of the Taylor expansion. In fact, this is where the large error kicks in. Expanding \( u(X) \) by its Taylor expansion, \( u(X) \) can be expressed as

\[
u(X) = p(X)v(X)\left[1 + \frac{1}{2} \text{tr}\left(\frac{\nabla^2 p(X)}{p(X)} A\right)d^2_{k\text{NN}}(X) + \text{H.O.T.}\right]
\]

where \( p(X) \) is the probability density at sample \( X \), \( A \) is the metric for measuring distance and \( v(X) \) is the volume of the local region \( L(X) \). The value of the trace term is not known, if the distribution is unknown. If the distribution is Gaussian with \( M=0 \) and \( \Sigma = I \), this trace term is \( X^T X - n \) whose absolute value is greater than one most of the time, especially for large dimensional data. The second order term has \( d^2_{k\text{NN}}(X) \) involved, which is much greater than 1 for large dimensional data. For example, the average 1st NN distance of the ISAR data with \( n = 640 \) is about 17. After taking the square of the \( k \)-th NN distance and multiplying by the trace term, we find that the second term is much larger than the first term. The higher terms will have even larger values. Therefore, approximating the probability mass \( u(X) \) by its first term only is erroneous for large dimensional data. However, if we include the second term in the derivation, the mathematical equation becomes too complicated to solve and we simply can not obtain an explicit expression for the \( k \)-th NN distance. Even if we could find an explicit expression for the \( k \)-th NN distance, this expression would definitely be very complicated and the terms that contain the metric \( A \) could not be factored out so that they could be canceled out in the denominator and numerator. Thus, we believe the estimated intrinsic dimensionality is actually dependent on the metric \( A \). When the sample size is large enough (local region is small enough), the impact of the selection of the metric is almost negligible. Nevertheless, the dependency on the metric \( A \) becomes clear as data dimension increases. For the ISAR data, the sample size is limited. Therefore, we can not neglect the effect of the metric \( A \).
5.2.2. Intrinsic dimensionality estimation by window-shifting method

The ordinary SAR data has an image size of 64x64. If we want to apply (5.1) to obtain the intrinsic dimensionality, we probably need billions or even trillions of samples to achieve a "small" local region. As many papers have shown, there must be a certain degree of approximation involved in order to obtain solvable equations. The problem is that all the approximations they make cause immense error in the estimated intrinsic dimensionality. Since our local metric estimation algorithm involves intrinsic dimension estimation, we inevitably face this issue: how to estimate the intrinsic dimension of 4,096-dimensional data using only 1,000 samples?

Before we present our solution to this problem, we would like to show the effect of the sample size on the estimated intrinsic dimensionality. Taking the Radar-32 data for example, we employ (5.1) and the sample covariance as the metric to compute the intrinsic dimensionality with various sample sizes. We plot the estimated intrinsic dimension vs. the ratio of $N/n$ in Fig. 5.1, where $n = 32$.

![Fig. 5.1. Sample size effect on the estimated intrinsic dimension of the Radar-32 data.](image-url)
Note that the estimated intrinsic dimension goes up very quickly as the sample size approaches the data dimension, i.e., the N/n ratio approaches 1. If the sample size is large enough, the curve stabilizes and stays at around 13. If the N/n ratio is less than 1, the result is highly unpredictable because the covariance matrix is singular. Even if the pseudo inverse is used, the result still offers no useful information to us. For example, if N = 31, \( \hat{n_e} = 19.9 \). If N is down to 28, \( \hat{n_e} = 13.9 \). However, if N = 25, \( \hat{n_e} \) goes up to 17.7. The curve in the region N/n < 1 simply fluctuates randomly and does not bear any useful information. Therefore, we conclude that if N/n is smaller than a certain number, say 2 for the Radar-32 data, the estimated intrinsic dimension is highly unreliable and overestimated most of the time. Let us look at some other examples to see if they have a similar sample size effect. The estimated intrinsic dimensions of the Radar-64 data and the ISAR data with \( n = 5 \times 8 \) are shown in Fig.'s 5.2 and 5.3 respectively. Note that the curves in Fig.'s 5.1 through 5.3 show the same trend as the N/n ratio approaches 1: the intrinsic dimension of the data set will be overestimated. The closer the N/n ratio gets to 1, the larger the bias is. Therefore, we can not apply (5.1) to data with a small N/n ratio using the full covariance as the metric. Unfortunately, for large dimensional data such as the SAR data we have to operate at the region where the N/n ratio is close to 1 or even less than 1.

Our solution is based on a very simple idea and experimental observations. Although we can not offer the theoretical justification, it has been verified to work very well for all the data sets we tested with known intrinsic dimensionality. As discussed in Chapter 4, the window-shifting method can downscale an n-dimensional problem into a \( w^2 \)-dimensional problem, where \( w^2 << n \). Even if \( N < n \), we can have \( w^2 << N \). Therefore, (5.1) is valid for small window size. We also observed that, for small window sizes, the estimated intrinsic dimension is almost a linear function of the square root of the number of pixels in the window, i.e.,

\[
\hat{n_e} \cong c_1 + c_2 \sqrt{n_w}
\]

where \( c_1 \) and \( c_2 \) are constants and \( n_w \) is the number of pixels in the window.
Fig. 5.2 Sample size effect on the estimated intrinsic dimension of the Radar-64 data.

Fig. 5.3 Sample size effect on the estimated intrinsic dimension of the ISAR data, $n=5\times8$. 
If each window is of size \(wxw\), then \(n_w = w^2\). However, if a non-square window is used, such as the rectangular window of size \(wx1\) for the 1-dimensional data, then \(n_w = w\). We can rely on the intrinsic dimensionality estimated by small windows and extrapolate to obtain the intrinsic dimension of the full-size image. In Fig's 5.1 - 5.3, we see that the curves in the region \(1 < N/n < 2\) rise very quickly as the sample size decreases. This means that, in this region, applying (5.1) with the full covariance as the metric will generate erroneous results. For the ISAR data with dimension \(n=10x16\), the \(N/n\) ratio is \(246160/16=1.54\). Therefore, we are particularly interested in the case \(1 < N/n < 2\). On the other hand, if the ratio \(N/n < 1\), the covariance matrix is singular and, as we have mentioned shortly before, the intrinsic dimensionality estimated using the pseudo-inverse of the covariance as the metric is highly unreliable. Therefore, we are also interested in solving the problems that fall in this category. We thus demonstrate how our method works for the following two cases: 1) \(1 < N/n < 2\) and 2) \(N/n < 1\).

Let us examine the Radar-32 data in the first case, \(1 < N/n < 2\), to see how our method overcomes the difficulty of an insufficient sample size. We pick up 36 samples from the Radar-32 data set for experiments so that \(N/n = 36/32 = 1.125\) belongs to the first case. The result is shown in Fig. 5.4. In order to compare the experiments of our method with the conventional ones, we call the intrinsic dimension estimated by a large number of samples with the full covariance the "reference intrinsic dimension". The reference intrinsic dimension of the Radar-32 data is about 13. Note that there are one curve and one straight line in Fig. 5.4(a). The curve is the estimated intrinsic dimensions for various sizes of windows. The straight line is the MMSE (Minimum Mean Square Error) line generated from the portion where the \(\hat{n}_e\) curve is roughly a straight line. When the full covariance \((w=32)\) with \(N=36\) is used, the estimated intrinsic dimension is 18.6. However, if we use (5.3), we get \(n_e = 13.2\) for \(\sqrt{n_w} = \sqrt{32} = 5.66\) which is about equal to the reference intrinsic dimension of the data. The result is the same as we have expected. For small windows, the ratio \(N/n_e\) is quite large; thus, it is all right to accept the result given by (5.1). As window size increases, the ratio \(N/n_w\) gets smaller, and finally we
see overestimated intrinsic dimensions. The estimation bias gets large as the window size increases. The full covariance corresponds to the window size of $32 \times 1$. Now let us check Case 2. We pick up 20 samples from the Radar-32 data set for the experiment. The $N/n$ ratio is $20/32 = 0.625$. Note that the full covariance obtained by these 20 samples is singular, yet the window covariance with small window size is invertible. Therefore, we cannot compute the intrinsic dimension using the full covariance ($w=32$) for Case 2. Performing extrapolation again as shown in Fig. 5.4(b), we obtain the estimated intrinsic dimension $\hat{n}_c = 14.1$. Using merely 20 samples, this result is considered as reasonably close to the reference of 13.

(a). $N/n=1.125$

(b). $N/n=0.625$

Fig. 5.4. Estimation of intrinsic dimension of the Radar-32 data by window-shifting method.
Applying the same technique to the Radar-64 data, we obtain the results shown in Fig. 5.5. Employing the whole set, \( N = 4000 \), we estimate the reference intrinsic dimension of the Radar-64 data to be 33. For Case 1 (Fig. 5.5(a)), we use 80 samples for the experiment with the ratio \( N/n = 80/164 = 1.25 \). Picking up 40 samples out of the Radar-64 data set and performing the same procedure, we obtain the results for Case 2 shown in Fig. 5.5(b). The \( N/n \) ratio is only \( 40/64 = 0.625 \). For both cases, the intrinsic dimension estimated by our method is very close to the reference intrinsic dimension. Now let us look at a 2-D example. The ISAR data with \( n = 5 \times 8 = 40 \) has reference intrinsic dimension \( n^* = 22 \), which is estimated by the whole data set, \( N = 246 \). We use 45 samples for Case 1 \( (N/n = 30/40 = 1.125) \) and 30 samples for Case 2 \( (N/n = 30/40 = 0.75) \). The experimental results are shown in Fig. 5.6.

\[ \text{(a). } N/n = 1.25 \quad \text{ (b). } N/n = 0.625 \]
In these examples we have also observed that the discrepancy between the estimated intrinsic dimensionality using (5.1) with and without window-shifting distances gets large as the data dimension increases. This coincides with our argument on the invalidity of conventional methods for handling samples with small $N$ and large $n$. We have demonstrated that our method for estimating the intrinsic dimension of the data with insufficient samples works very well for both Cases 1 and 2. The experimental results also confirm our conjecture that the window-shifting method can downscale an $n$-dimensional problem to an $n_w$-dimensional problem, where $n_w = w$ for 1-D data and $n_w = w^2$ for 2-D data.

So far all our work has been done for the data set whose reference intrinsic dimension can be established. We have shown that, even with very small sample sizes, our method consistently generates results close to the references for both 1-D and 2-D data sets. Therefore, we are confident in applying our method to estimate the intrinsic dimension of the data set where conventional methods fail to work. The data sets that we
have at hand for analysis are ISAR data with \( n=10 \times 16 \) and \( n=20 \times 32 \). In order to estimate the Bayes error of these two data sets, we have to know their intrinsic dimensions in order to select parameters like the number of local samples. With only 246 samples available, the \( N/n \) ratios for these two data sets are 1.54 and 0.38 respectively. These two \( N/n \) ratios happen to be in the "difficulty zone" where conventional methods fail to give any reliable numbers for the intrinsic dimension. If the full covariance is used as the metric for measuring distances for the 10x16 dimensional ISAR data, the intrinsic dimension is estimated to be as high as 143. However, our method obtains the estimated intrinsic dimension \( \hat{n}_e = 39.1 \) as shown in Fig. 5.7. Observing the curve of \( \hat{n}_e \) vs. window size, we are convinced that 39.1 is much more likely to be the intrinsic dimension of the 160-dimensional ISAR data than 143. The 640-dimensional ISAR data belongs to Case 2. Due to the limited computing power of our workstation, we can only perform window-shifting using window sizes from 1 to 5. Nevertheless, the curve in Fig. 5.8 is almost a straight line; thus, from our previous experience, it suffices to use these five points to extrapolate the intrinsic dimension at the point \( n_w = 640 \). The estimated intrinsic dimension of the 640-dimensional data is about 188.1. From our previous experiments, we are sure that this number should be close to the reference intrinsic dimension of the 640-dimensional ISAR data.

![Fig. 5.7. Estimation of intrinsic dimension of the ISAR data, \( n=10 \times 16 \), by window-shifting method with \( N/n=1.54 \).](image)
Fig. 5.8. Estimation of intrinsic dimension of the ISAR data, $n=20 \times 32$, by window-shifting method with $N/n=0.38$.

The experimental result of this section is summarized in Tables 5.2 and 5.3 for convenience of comparison. The reference intrinsic dimensions of these data sets are obtained by the whole data set with the full covariance as the metric. The sample size is large enough compared to the data dimension. It is clearly seen that our method provides a very stable and powerful solution to this problem where the conventional method gives erroneous results or even fails to give any results. Therefore, we believe that we have found a promising method of estimating the intrinsic dimension of data sets with an insufficient number of samples. This method works for both 1-D vector signals and 2-D images as we have shown in this section.
Table 5.2. The summary of experimental results of Case 1: $1 < N/n < 2$.

<table>
<thead>
<tr>
<th>Data</th>
<th>$N/n$</th>
<th>$\hat{n}_e$</th>
<th>conventional</th>
<th>window-shifting</th>
<th>reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Radar-32</td>
<td>1.125</td>
<td>18.6</td>
<td>13.2</td>
<td>13</td>
<td></td>
</tr>
<tr>
<td>Radar-64</td>
<td>1.25</td>
<td>50.0</td>
<td>34.4</td>
<td>33</td>
<td></td>
</tr>
<tr>
<td>ISAR, $n=5\times8$</td>
<td>1.125</td>
<td>45.1</td>
<td>21.1</td>
<td>22</td>
<td></td>
</tr>
<tr>
<td>ISAR, $n=10\times16$</td>
<td>1.54</td>
<td>143.0</td>
<td>39.1</td>
<td>unknown</td>
<td></td>
</tr>
</tbody>
</table>

Table 5.3. The summary of experimental results of Case 2: $0 < N/n < 1$.

<table>
<thead>
<tr>
<th>Data</th>
<th>$N/n$</th>
<th>$\hat{n}_e$</th>
<th>conventional</th>
<th>window-shifting</th>
<th>reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Radar-32</td>
<td>0.625</td>
<td>cannot be determined</td>
<td>14.1</td>
<td>13</td>
<td></td>
</tr>
<tr>
<td>Radar-64</td>
<td>0.625</td>
<td>cannot be determined</td>
<td>36.9</td>
<td>33</td>
<td></td>
</tr>
<tr>
<td>ISAR, $n=5\times8$</td>
<td>0.75</td>
<td>cannot be determined</td>
<td>19.2</td>
<td>22</td>
<td></td>
</tr>
<tr>
<td>ISAR, $n=20\times32$</td>
<td>0.38</td>
<td>cannot be determined</td>
<td>188.1</td>
<td>unknown</td>
<td></td>
</tr>
</tbody>
</table>

5.3. Sample Correlation and Local Metric B

Based on our derivation of the local metric estimation in Chapter 2, the second order derivative of the probability density can be approximated by

$$\hat{K} \equiv \alpha B^{-1}\hat{S} B^{-1}$$  \hspace{1cm} (5.4)

where $\alpha$ is a scaling constant determined by equating the determinant of $K$ to 1, $B$ is the
local metric used to measure the distance between neighbors and $\hat{S}$ is the sample autocorrelation of the local region. The selection of $B$ is very crucial to the performance of the Bayes error estimation algorithm. We have discussed methods of choosing $B$ for Radar signals in Section 3.2. In the past, we had thousands of samples available for data with a dimension lower than 100. Therefore, it is easy to find sufficient neighboring samples to form the local metric $B$. When we process the ISAR data, the number of uncorrelated samples is so small that it is very difficult to gather enough neighboring samples to build the local metric $B$. The consequence is that the estimated error rate will contain a positive bias in both the R and L errors. Therefore, the Bayes error may be overestimated. This sample size effect has been described in Section 3.4. Nevertheless, there are thousands of correlated samples available for the ISAR data. We have also mentioned in Section 3.4 that the test samples must be uncorrelated, otherwise the Bayes error might be underestimated. In order to alleviate the difficulty of having insufficient samples, we would like to know what might happen if we use correlated samples to estimate the local metric $B$ while using uncorrelated samples for testing. In Fig. 5.9 the solid circles are uncorrelated samples which are used as test samples, while the empty circles represent correlated samples which should not be used as test samples. If we use uncorrelated samples to form $B$ for the local region around $X_0$, we can only find 3 samples in the local region. However, if we allow correlated samples to compute the local metric $B$, we can have 9 samples in the same region.

Fig. 5.9. How to obtain $B$ from correlated and uncorrelated samples.
Since all the test samples are uncorrelated, the Bayes error will not be underestimated. Our study on sample correlation in Section 4.5 shows that, if the samples are linearly correlated, the covariance of these samples is similar to the covariance of the uncorrelated samples, differing only by a scaling constant. We see that, if the metric $B$ in (5.4) is replaced by $B/c$, $\hat{K}$ will not be affected because this scaling factor will be automatically taken care of by metric normalization. In order to verify the validity of employing correlated samples to form the local metric $B$, we have to run experiments on the standard data sets whose Bayes errors are known.

First, we generate independent samples $X_i$, $i=1,2,...,cN$, and then use the method introduced in Section 4.5 to generate correlated samples $Y_i$, $i=1,2,...,N$ and uncorrelated samples $Z_i$, $i=1,2,...,N$. The sample size of $Y$ is the same as that of $Z$ for performance comparison. We test 1-1, 1-41 and I-A data to see if correlated samples in $B$ will cause any undesired results. The experimental results are shown in Tables 5.4 - 5.6 for the same sample size $N=200$. Note that $c=1$ corresponds to uncorrelated samples. We see that, whether correlated samples or uncorrelated samples are used, the Bayes error is always bounded properly with our algorithm as long as the test samples are uncorrelated. This is a favorable development for us when we process the ISAR data.

Now we would like to perform the same experiments on the real radar data to verify our conjecture. For Radar-32 data, the adjacently indexed samples are correlated. The first choice is to select adjacently indexed samples (correlated samples) to form $B$. On the other hand, if we pick up samples with non-consecutive indices, for instance, $i=1,4,7,...$, etc., we can obtain uncorrelated samples to build $B$. Using these two different ways of forming a local metric $B$, we obtain the experimental results shown in Table 5.7. Both methods offer proper lower and upper bounds for the Bayes error, but by employing correlated samples we can get tighter bounds on the Bayes error. Although the Radar-64 samples are independent (according to the data provider), we would like to adopt the same strategy of selecting samples to form $B$ to see what kind of error bound we will get. Using consecutively indexed samples, we obtain correlated samples. Picking up every other samples, we get uncorrelated samples. The results are shown in Table 5.8 for these
Table 5.4. Bayes error estimation of I-I data, $\varepsilon = 10\%$, with correlated samples for metric $B, N=200$.

<table>
<thead>
<tr>
<th>$c$</th>
<th>R error (%)</th>
<th>L error (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8.40</td>
<td>12.15</td>
</tr>
<tr>
<td>2</td>
<td>9.05</td>
<td>12.40</td>
</tr>
<tr>
<td>3</td>
<td>8.00</td>
<td>11.25</td>
</tr>
<tr>
<td>4</td>
<td>7.45</td>
<td>10.25</td>
</tr>
<tr>
<td>5</td>
<td>9.00</td>
<td>12.70</td>
</tr>
<tr>
<td>8</td>
<td>8.05</td>
<td>11.00</td>
</tr>
<tr>
<td>10</td>
<td>8.05</td>
<td>10.70</td>
</tr>
<tr>
<td>20</td>
<td>7.95</td>
<td>10.90</td>
</tr>
</tbody>
</table>

Table 5.5. Bayes error estimation of 1-41 data, $\varepsilon = 9\%$, with correlated samples for metric $B, N=200$.

<table>
<thead>
<tr>
<th>$c$</th>
<th>R error (%)</th>
<th>L error (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>7.45</td>
<td>10.95</td>
</tr>
<tr>
<td>2</td>
<td>7.25</td>
<td>10.25</td>
</tr>
<tr>
<td>3</td>
<td>8.00</td>
<td>10.55</td>
</tr>
<tr>
<td>4</td>
<td>8.00</td>
<td>10.60</td>
</tr>
<tr>
<td>5</td>
<td>6.95</td>
<td>10.00</td>
</tr>
<tr>
<td>8</td>
<td>8.40</td>
<td>11.15</td>
</tr>
<tr>
<td>10</td>
<td>7.30</td>
<td>10.30</td>
</tr>
<tr>
<td>20</td>
<td>7.70</td>
<td>10.20</td>
</tr>
</tbody>
</table>
Table 5.6. Bayes error estimation of I-A data, $\varepsilon = 1.9\%$, with correlated samples for metric B, $N=200$.

<table>
<thead>
<tr>
<th>$c$</th>
<th>R error (%)</th>
<th>L error (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.75</td>
<td>2.90</td>
</tr>
<tr>
<td>2</td>
<td>1.45</td>
<td>3.10</td>
</tr>
<tr>
<td>3</td>
<td>1.35</td>
<td>3.75</td>
</tr>
<tr>
<td>4</td>
<td>1.20</td>
<td>3.35</td>
</tr>
<tr>
<td>5</td>
<td>1.35</td>
<td>2.60</td>
</tr>
<tr>
<td>8</td>
<td>1.35</td>
<td>2.30</td>
</tr>
<tr>
<td>10</td>
<td>1.45</td>
<td>3.25</td>
</tr>
<tr>
<td>20</td>
<td>1.45</td>
<td>2.15</td>
</tr>
</tbody>
</table>

Table 5.7 Bayes error estimation of Radar-32 data with correlated and uncorrelated samples for metric B, $N=200$, $\varepsilon \leq 9\%$.

<table>
<thead>
<tr>
<th>Samples for B</th>
<th>R error (%)</th>
<th>L error (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>correlated</td>
<td>7.75</td>
<td>11.50</td>
</tr>
<tr>
<td>uncorrelated</td>
<td>7.00</td>
<td>12.75</td>
</tr>
</tbody>
</table>

Table 5.8 Bayes error estimation of Radar-64 data with correlated and uncorrelated samples for metric B, $N=200$, $\varepsilon \leq 18\%$.

<table>
<thead>
<tr>
<th>Samples for B</th>
<th>R error (%)</th>
<th>L error (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>correlated</td>
<td>16.17</td>
<td>20.03</td>
</tr>
<tr>
<td>uncorrelated</td>
<td>13.81</td>
<td>20.31</td>
</tr>
</tbody>
</table>

two cases. The Bayes error is properly bounded no matter whether the samples used to form B are correlated or not. From the experiments on the standard data set and the real data sets we are assured that correlated samples may be allowed to build up the local metric B. Therefore, we are going to apply the finding of this section to estimate the Bayes error of the ISAR data.
5.4. Fast Computation Method of The Bayes Error Estimation Algorithm

So far we have fully explored the problems involved in estimating the Bayes error of 2-dimensional images such as the ISAR data. Still, we believe that the computational load of estimating the Bayes error of high dimensional data like SAR data is too heavy for ordinary workstations, although the computation speed and memory requirement may not be a concern for some research institutions. Therefore, we decide to increase the computational efficiency of our algorithm by dimensional reduction, which is achieved by replacing the matrix multiplication with vector multiplication. To do so we have to compute the eigenvalues and eigenvectors of the local autocorrelation matrix \( \hat{S} \) and the local metric \( B \). Since the matrices \( B \) and \( \hat{S} \) are symmetric and positive semidefinite, their eigenvectors corresponding to the non-zero eigenvalues are orthonormal.

In our algorithm, we have to compute the local autocorrelation matrix \( \hat{S} \) of the local samples for each test sample, and the estimated optimal local metric \( A^{-1} \) is related to \( B^{-1} \hat{S} B^{-1} \) by

\[
A^{-1} = \alpha B^{-1} \hat{S} B^{-1}
\]  

(5.5)

where \( \alpha \) is a scaling constant so that the determinant of \( A \) is 1.

If, for example, the SAR data with \( n=4096 \) is processed, these matrices \( B \) and \( \hat{S} \) are size \( 4096 \times 4096 \). The computation required for computing the matrix multiplication of such a huge size is painfully slow. However, these two matrices \( B \) and \( \hat{S} \) are highly singular. The ranks of \( B \) and \( \hat{S} \) are \( l_B \) and \( l_S \) respectively, which are both much smaller than the dimension \( n \). The multiplication will run much faster if we make use of this property. Therefore, we have to compute the eigenvalues and eigenvectors of \( \hat{S} \) and \( B^{-1} \).

Let there be \( l_S \) nearest neighbors, \( X_i, i = 1, 2, \ldots, l_S \), enclosed in the local region of the test sample \( X_t \). The fast procedure for computing the eigenvectors; and eigenvalues of the local autocorrelation \( S \) is stated as follows [37]:

Let \( U = [X_1 - X, X_2 - X, \ldots, X_{l_s} - X] \) be the local sample matrix with
respect to the test sample $X_t$. The size of the matrix $U$ is $n \times I_S$, where $I_S$ is much smaller than $n$. The local sample autocorrelation $\hat{S}$ can be obtained by

$$\hat{S} = UU^T / I_S$$

(5.6)

where $\hat{S}$ is an nxn matrix with rank $I_S$.

Instead of computing $S$ using (5.6), we compute the eigenvalues and eigenvectors of the matrix $U^T U / I_S$ of size only $I_S \times I_S$ by

$$(U^T U / I_S) \Xi = \Xi \Lambda$$

(5.7)

where $\Xi$ and $\Lambda$ are the eigenvector and eigenvalue matrices of $U^T U / I_S$ respectively. These two matrices $\Xi$ and $\Lambda$ are only size $I_S \times I_S$. Multiplying $U$ by both sides of (5.7), we have

$$(UU^T / I_S)(U\Xi) = (U\Xi) \Lambda$$

(5.8)

Therefore, by computing the eigenvalues and eigenvectors of the much smaller matrix $U^T U / I_S$ and multiplying them by the sample matrix $U$, we can obtain the eigenvalues and eigenvectors of the desired autocorrelation matrix $\hat{S}$. Note that the matrix $U\Xi$ consists only of the eigenvectors corresponding to non-zero eigenvalues. The size of $U\Xi$ and $\Lambda$ is $n \times I_S$ and $I_S \times I_S$ respectively. Since the eigenvectors $U\Xi$ are orthogonal but not orthonormal, we can obtain a set of orthonormal eigenvectors by

$$R = \frac{1}{\sqrt{I_S}} U\Xi \Lambda^{-1/2}$$

(5.9)

Thus, the orthonormal eigenvectors and eigenvalues of the autocorrelation matrix $\hat{S}$ are $R$ and $\Lambda$ respectively. A similar procedure can also be applied to the metric $B$ to compute its eigenvalues and eigenvectors. Taking the reciprocal of the magnitude of the eigenvalues of $B$, the pseudo-inverse of $B$ can be easily obtained. Now we are ready to demonstrate how to compute $B^{-1} \hat{S} B^{-1}$ by vector multiplication.

Applying the fast method mentioned above to $B$ and $\hat{S}$, we have

$$\hat{S} = \sum_{i=1}^{I_S} \lambda_i \theta_i \theta_i^T$$

(5.10)
and

\[ B = \sum_{j=1}^{I_f} \mu_j \phi_j \phi_j^T \]  \hspace{1cm} (5.11)

where \( \lambda_i \) and \( \theta_i \) are the eigenvalues and eigenvectors of \( S \), and \( \mu_j \) and \( \phi_j \) are the eigenvalues and eigenvectors of \( B \). The product \( B^{-1} S B^{-1} \) becomes

\[
B^{-1} S B^{-1} = \left( \sum_{i=1}^{I_f} \frac{1}{\mu_i} \phi_i \phi_i^T \right) \left( \sum_{j=1}^{I_f} \lambda_j \theta_j \theta_j^T \right) \left( \sum_{k=1}^{I_f} \frac{1}{\mu_k} \phi_k \phi_k^T \right)
\]

\[ = \sum_{j=1}^{I_f} \lambda_j \eta_j \eta_j^T \]  \hspace{1cm} (5.12)

where

\[ \eta_j = \left[ \sum_{i=1}^{I_f} \frac{1}{\mu_i} (\phi_i^T \theta_j) \phi_i \right]. \]  \hspace{1cm} (5.13)

In (5.13) the term \( \phi_i^T \theta_j \) is a vector multiplication which takes only \( n \) scalar multiplications and \( (n-1) \) scalar additions and the result is a scalar. Instead of expressing (5.12) by an \( nxn \) matrix, we would rather keep the product in the form of \( \lambda \)'s and \( \eta \)'s. Note that \( \eta_j, j=1, 2, \ldots, I_f \), may not be orthogonal. Therefore, we have to find a set of orthonormal eigenvectors of \( B^{-1} S B^{-1} \) for the purpose of metric normalization. Rewriting (5.12) we have

\[
K = B^{-1} S B^{-1} = \sum_{j=1}^{I_f} \left( \sqrt{\lambda_j} \eta_j \right) \left( \sqrt{\lambda_j} \eta_j \right)^T
\]

\[ = \sum_{j=1}^{I_f} v_j v_j^T
\]

\[ = YY^T. \]  \hspace{1cm} (5.14)

The eigenvalues and eigenvectors of \( K \) may be computed as in (5.6) and (5.7). The \( I_g \times I_g \) matrix \( V^T V \), instead of the \( nxn \) matrix \( YY^T \), is diagonalized by

\[ V^T V = \Psi \tilde{D} \Psi^T \]  \hspace{1cm} (5.15)

where \( Y \) and \( D \) are the eigenvectors and eigenvalues of \( V^T V \) respectively. By (5.14) and
(5.15) we have

\[ KV = \Psi \Psi^T. \]  \hspace{1cm} (5.16)

Since the eigenvectors \( \Psi \) are orthonormal, we have \( \Psi^T \Psi = I \). Multiplying both sides of (5.16) by \( \Psi \) from the right, (5.16) becomes

\[ K \Psi \Psi = \Psi \Psi D \]  \hspace{1cm} (5.17)

We can find the orthonormal matrix \( H \)

\[ H = \Psi \Psi^{-1/2} \]  \hspace{1cm} (5.18)

such that

\[ KH = K \Psi \Psi^{-1/2} = \Psi \Psi D^{-1/2} = \Psi \Psi \Psi^T \Psi^{-1/2} = HD. \]  \hspace{1cm} (5.19)

Therefore, the matrices \( H \) and \( D \) are the eigenvectors and eigenvalues of \( K \), where \( D \) is the eigenvalue matrix of \( \Psi \Psi^T \) as shown in (5.15) and \( H \) can be computed by (5.18).

Let \( H = [h_1 \ h_2 \ \cdots \ h_s] \) and \( D = \text{diag}(\delta_1, \delta_2, \ldots, \delta_s) \). The normalized metric \( A \) can be obtained by

\[ A^{-1} = \frac{1}{V_s} \sum_{j=1}^{l_s} \delta_j h_j h_j^T. \]  \hspace{1cm} (5.20)

Note that we have to keep \( A \) in the form of \( 6's \) and \( h's \) so that the computation of distance can be also simplified to vector multiplication. After the metric \( A \) has been computed, we have to measure the distance between the test sample and its neighbors using metric \( A \). Let the test sample be \( X_t \) and its neighbor be \( X \). The distance between \( X_t \) and \( X \) is

\[ d^2(x, X_t) = (X_t - X)^T A^{-1} (X_t - X) \]

\[ = \frac{1}{V_s} \sum_{j=1}^{l_s} \delta_j (X_t - X)^T h_j h_j^T (X_t - X) \]

\[ = \frac{1}{V_s} \sum_{j=1}^{l_s} \delta_j [(X_t - X)^T h_j]^2. \]  \hspace{1cm} (5.21)

Note that the term inside the square bracket of (5.21) is a scalar. Therefore, the evaluation
of (5.21) is extremely fast. Now we would like to compare the computational complexity of direct computation and our fast method. By direct computation, the computation of $K = B^{-1} \hat{S} B^{-1}$ requires $2n^3$ scalar multiplications and $2n^2(n-1)$ scalar additions. By our method, we have to compute $\lambda_i, \theta_i, \mu_i$ and $\phi_i$ in (5.10) and (5.11) first. Since the intrinsic dimensionality is in general much smaller than the data dimension, $I_S$ and $I_B$ are much smaller than $n$. Therefore, the time required for computing these eigenvalues and eigenvectors of the small matrix of size $I_S \times I_S$ or $I_B \times I_B$ is negligible. Then we have to compute $\eta_i$ in (5.13) which takes $(2n+1)I_S I_B$ scalar multiplications and $(n-1)I_S I_B$ scalar additions. Then we have to compute $V^T V$ in (5.15) which takes $I_S^3$ scalar multiplications and $I_S^2(I_S - 1)$ scalar additions, which can be neglected for they have nothing to do with the data dimension $n$. Since $V^T V$ is only size $I_S \times I_S$, the time required for computing $\eta_i$ and 6, can also be neglected. In (5.18) the computation of $H$ takes $(n+I_S)I_S^2$ scalar multiplications and $(n+I_S)I_S(I_S - 1)$ scalar additions. Note that in Chapter 3 we have shown that $I_S$ and $I_B$ are directly related to the intrinsic dimension of that data. It can be clearly seen that the computational complexity of our method is at most only $O(m^2)$, where $O(.)$ means "on the order of", while the computational complexity of direct computation is $O(n^3)$. For data with extremely large dimension $n$ and small intrinsic dimension $n$, our method will be much faster than direct computation.

Another great advantage of our method is the greatly enhanced efficiency in making use of the computer system memory. Take the computation of $K$ and the SAR data with $n=4096$ for example. If each number is represented by 8-byte double precision floating numbers, then each nxn matrix needs 128 Mbytes of system memory to hold it. Therefore, we need at least 384 Mbytes to hold $B$, $\hat{S}$ and $K$, not to mention the other variables. This will cause memory swapping all the time and the computation will be painfully slow. In contrast, our method only gets hold of the eigenvalues and eigenvectors. If, for example, $I_S = I_B = 200$, each matrix takes only $8(n+1)I_S = 6.4$ Mbytes of the system memory. Therefore, the memory requirement is down from 384 Mbytes to
about 20 Mbytes. Without memory swapping, our method will definitely run much faster than direct computation which suffers from memory swapping.

5.5. Bayes Error Estimation of The ISAR Data

Because of the absence of the SAR data at the press time of this thesis, we can only apply our Bayes error estimation algorithm to the ISAR data to test its performance. Like the SAR data, the ISAR data are also a collection of 2-dimensional images containing 360 degrees of viewing angles. They have the same characteristic that the intrinsic dimensionality is lower than the data dimension. Therefore, testing our algorithm on the ISAR data may give us useful information to predict its performance when it is applied to the SAR data. The test objects are two commercial vehicles: the Chevrolet Camaro and Dodge Van.

From the histogram of some selected pixels, we realize that the distribution of the ISAR data is unimodal and non-Gaussian. Nevertheless, since the data set is a collection of 360 degrees of viewing angles, the actual distribution should be able to be better approximated by multi-cluster distribution. Therefore, we believe that local metrics should perform better than the global ones. Applying our algorithm to the ISAR data of different dimensions, we obtain the upper and lower bounds for the Bayes error shown in Table 5.9. The error rates estimated by the Parzen classifiers with global metrics and quadratic classifiers are also shown for comparison. The Bayes error for the ISAR data estimated by our algorithm is 0%. However, if a global metric is applied, it will overlook some local details and result in a higher error rate; thus, the Bayes error will be overestimated.

Due to the limited computing power and memory capacity of the computers available in the Electrical and Computer Engineering Department, it is prohibitive for us to pursue the Bayes error estimation for the ISAR data with n=20×32. Since the data with reduced dimension and resolution already have a Bayer error of 0%, the data of the
original dimension are expected to be 100\% classifiable.

Let us summarize the procedure for Bayes error estimation here.

1). Estimate the intrinsic dimension of the data set. If the sample size is very large compared to the data dimension, (5.1) works fine; otherwise, we have to use the window-shifting method (Section 5.2) to estimate the intrinsic dimension.

2). Make sure the test samples are uncorrelated.

3). Perform quadratic error estimation to obtain an upper bound for the Bayes error. The window-shifting method (Section 4.6) helps to narrow down the range of the quadratic error. If the distribution of the samples is close to Gaussian, the Bayes error may be only a little lower than the quadratic error; otherwise, the Bayes error may be considerably lower than the quadratic error.

4). Select the local metric $B$ by using the viewing angle information. Correlated samples are allowed to build the local metric $B$. The determination of $I_B$ is data-dependent. For the ISAR data, we choose to use the samples spanning about 30 degrees of viewing angles ($I_B=400$) to form the local metric $B$.

5). Select the number of local samples, $I_S$ and $I_p$. These two numbers are directly related to the intrinsic dimension estimated in step 1. Section 3.3 offers a guideline how to avoid a poor selection of the number of local samples. Different data sets may have different optimal number of local samples.
Experiments are required to obtain the optimal combination of $I_S$ and $I_p$. From our experiments with the Radar-32 and Radar-64 data, the optimal combination of $I_S$ and $I_p$ may be found in the region $1.5 n_e \leq I_S \leq 2 n_e$ and $0.5 I_S \leq I_p \leq 0.8 I_S$. For the ISAR data, we select $I_S = 1.5 n_e$ and $I_p = 0.8 n_e$.

6). Compute the local autocorrelation $S$ for each test sample and then estimate the local metric $A$. Follow the steps in Section 5.4 to compute the distance between samples and apply the Parzen classifier with the R. and L methods to obtain the lower and upper bounds for the Bayes error.

7). Reduce or increase the number of test samples to see if there is a large difference between the current result and the result obtained in step 6. If the difference is quite small, we know that Bayes error has been properly bounded. If the difference is quite large, we have to check if the test samples are correlated. In case the test samples are guaranteed to be uncorrelated and the estimated error rates keep going down as sample size increases, a larger number of samples is required to bound the Bayes error properly.

5.6. Conclusion

In this chapter, we have demonstrated how to estimate the intrinsic dimensionality of data with extremely small sample size. The discussion is divided into two cases: one with non-singular covariance and one with singular covariance. Our method is proven to work very well and robustly by various experiments. We have also demonstrated the importance of local covariance $B$ in our algorithm. We have shown by experiments on the standard data set and two real data sets that employing correlated samples in $B$ will not cause any undesirable bias to the estimated Bayes error if all the test samples are uncorrelated. In order to speed up the process and reduce the system memory requirement, we proposed a fast method to compute matrix multiplications in our algorithm. This method is very efficient especially for data with extremely large data
dimension \( n \) and very small intrinsic dimensionality \( n_e \). Finally, we analyzed and estimated the Bayes error of the ISAR data due to the absence of the SAR data. The process of Bayes error estimation is memory and computation intensive; therefore, we, restricted by the equipment that we have, can only process the ISAR data with reduced dimension. However, our method is proven to work very well. We believe that we are able to estimate the Bayes error of our goal, the SAR data, when the data become available.
CHAPTER 6
SUMMARY AND FUTURE WORK

6.1. Summary of Contribution

The main purpose of this thesis is the Bayes error estimation of huge dimensional data. We have derived an algorithm to estimate the second order derivative of the probability density from the neighboring samples. The local metric is estimated by trace equalization and metric normalization. We have also found that the Parzen classifier is more robust and performs constantly better than the k-nearest neighbor (k-NN) classifier in the sense of Bayes error estimation. We established this in Chapter 2. The data under testing are the standard data sets whose Bayes errors are known and the distributions are Gaussian.

Our algorithm is refined further in Chapter 3. We discussed issues like the connections between the number of local samples and the intrinsic dimensionality of the data, how to select the local metric B, the kernel shape effect and the sample size effect, etc. These properties related to the local metrics help us to understand how to estimate the Bayes error of the given data set. Our algorithm has also been applied to real radar signals and is verified to work very well.

Another problem we have solved is the design of a quadratic classifier for large dimensional data like SAR data. Instead of discarding all the off-diagonal terms of the covariance matrix, as is done by many classifier designers, we developed a new way to measure distance by window-shifting. By increasing the window size, we can make use of more correlation information among neighboring pixels, thus achieving a better...
classification performance. The window-shifting distance measurement also helps to estimate the Bhattacharyya distance when the covariance matrices are singular. We also found out that outliers have little effect on the estimated Bhattacharyya distance; thus, we do not have to worry about the outliers' impact when we estimate the Bhattacharyya distance in real applications. The window-shifting method can reduce the effect of insufficient sample size on the estimated error bounds for the quadratic error, thus achieving a much tighter upper bound. This is what we established in Chapter 4.

Since many of the parameters of our Bayes error estimation algorithm are strongly connected to the intrinsic dimensionality of the data, we have to find an effective and reliable way to estimate the intrinsic dimension for the situation where the sample size is extremely small and the covariance is even singular. Based on the window-shifting method and extrapolation, we successfully developed a new way to estimate the intrinsic dimensionality. The intrinsic dimensionality of the ISAR data is estimated, while the conventional method fails to work. We have also developed a fast method to perform matrix multiplication and distance computation to save time and memory. Finally, at the end of Chapter 5, the Bayes error of the ISAR data is estimated and compared to the results achieved by conventional method.

6.2. Future Work

We have proven that by going into local metric estimation we can achieve a better classification result than the global metric does. Therefore, more work can be done in this direction to understand the behavior of local metrics. A more systematic discussion on the properties of local metrics can be pursued both in experiments and in theory. If the SAR data become available, we would like to apply our algorithm to this huge data set to see how well our method can do.

The problems of sample correlation definitely deserve more attention. The effect of
sample correlation on many parameter estimation problems is not fully known so far. **Modeling** different kinds of sample correlation may be an important step to study the **effect** of sample correlation.

The window-shifting distance measurement method offers another good way to **classify** targets quickly with higher accuracy. Since this method involves only simple fixed-sized matrix multiplication, it is especially suitable for VLSI implementation such that it can be used as a real-time classifier with little effort. With highly parallel and pipelined architecture, this algorithm can be implemented as an add-on board to the computer so that real-time multiple-target classification is possible. **This** method is also very suitable for parallel computers like **MASPAR**. This technique will definitely be a good solution to the SAR data related real-time classification problems.
LIST OF REFERENCES
LIST OF REFERENCES


[4]. E. Fix and J. L. Hodges, "Discriminatory analysis, nonparametric discrimination, consistency properties," School of Aviation Medicine, Randolph Field, TX, Report No. 4, Project 21-49-004, 1951.


APPENDICES
Appendix A
The Data Sets Used in This Thesis

1. Standard Data Sets:

There are three kinds of data used in Chapter 2. They are called standard data sets, which are generated as random variables with computer software. The detailed specification of this data set is described as follows.

- Type of distribution: Gaussian,
  Dimension: \( n = 8 \),
  Number of classes \( L = 2 \),
  Distribution parameters:

\[
M_1 = \mu_1 = [0 \ldots 0]^T, \quad M_2 = \mu_2 = [m_1 \ldots m_8]^T,
\]

\[
\Sigma_1 = \Lambda = [1 \ldots 0], \quad \Sigma_2 = \Lambda = \begin{bmatrix} \lambda_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_8 \end{bmatrix}
\]

1). \( I-I \) data:

\( m_1 = 2.56, \quad m_2 = \ldots = m_8 = 0 \),

\( \lambda_1 = \ldots = \lambda_8 = 1 \)

Bayes error = \( 10\% \)

2). \( I-A \) data:

\( m_1 = m_2 = \ldots = m_8 = 0 \),

\( \lambda_1 = \ldots = \lambda_8 = 4 \)

Bayes error \( \approx 9\% \)
3). I-A data:

<table>
<thead>
<tr>
<th>i</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>m_i</td>
<td>3.86</td>
<td>3.10</td>
<td>0.84</td>
<td>0.84</td>
<td>1.64</td>
<td>1.08</td>
<td>0.26</td>
<td>0.01</td>
</tr>
<tr>
<td>λ_i</td>
<td>8.41</td>
<td>12.06</td>
<td>0.12</td>
<td>0.22</td>
<td>1.49</td>
<td>1.77</td>
<td>0.35</td>
<td>2.73</td>
</tr>
</tbody>
</table>

Bayes error ≅ 1.9%.

II. Artificial Data Sets:

This data set is used in Chapter 3. In order to simulate real data, we have to generate various artificial data for experiments. These artificial data have the same characteristic that the intrinsic dimensionality \( n_e \) is less than the data dimensionality \( n \).

From the data acquisition point of view, it is extremely difficult to obtain a large number of samples. However, this constraint does not exist for the artificial data. We can generate as many uncorrelated samples as we want. In order to study the sample size effect, it is necessary to generate good artificial data that are similar to the real data. To achieve this, we generate three artificial data sets as follows.

1). Linear data set:

This data set is a linear mapping from the 3-dimensional parameter space to the 8-dimensional sample space. The input-output relation is defined by

\[
Y = TX
\]  \hspace{1cm} (A.1)

where \( Y \) is the observed sample of size 8×1 and \( X \) is the input parameter of size 3×1. \( T \) is the linear transformation matrix of size 8×3 and is randomly generated to have rank = 3. Each entry of the output vector is simply a linear combination of the 3 parameters. Class 1 is Gaussianly distributed with covariance \( \Sigma = I \). Class 2 is also Gaussianly distributed with covariance \( \Sigma = 10I \). They are not mean-separable. The Bayes error on the parameter
space is approximately 9.6%. We expect that the Bayes error on the sample space is also 9.6%.

2) Gaussian pulse data set I:

The Gaussian pulse is defined by

\[ x(t) = ae^{-\frac{(t-m)^2}{2\sigma^2}} \]  \hspace{1cm} (A.2)

where the parameters \( a, m \) and \( \sigma \) are independent random variables. The output is a function of time \( t \).

In this data set, there are three independent parameters governing the behavior of the observed output. The parameter \( a \) controls the magnitude of the pulse. The parameter \( m \) determines the position where the peak of the pulse occurs, while another parameter \( \sigma \) decides the width of the pulse. The output vector \( X \) is a collection of the time samples of the observed waveform by

\[ X = [x(t_1) \ x(t_2) \ x(t_3) \ldots x(t_n)]^T \]  \hspace{1cm} (A.3)

where

\[ t_i = 0.1667(i - 1), \quad i = 1, 2, 3, \ldots, n \]  \hspace{1cm} (A.4)

where \( n \) can be any positive integer larger than 3.

Since the exponential function gives a very small number for a large negative argument, thus causing classification problems, we must control the value of the three parameters to be in a proper region so that the Bayes error on the parameter space is close to the Bayes error on the sample space. The classification information of the two classes is given below.

Table A.1. The parameter list of the Gaussian pulse data set I.

<table>
<thead>
<tr>
<th></th>
<th>class ( a )</th>
<th>class ( m )</th>
<th>class ( \sigma )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( a )</td>
<td>( U[0.2, 1.2] )</td>
<td>( U[0.3, 0.7] )</td>
<td>( U[0.2, 0.4] )</td>
</tr>
<tr>
<td>( m )</td>
<td>( U[0.3, 0.7] )</td>
<td>( U[0.3, 0.7] )</td>
<td>( U[0.2, 0.41] )</td>
</tr>
<tr>
<td>( \sigma )</td>
<td>( U[0.3, 0.7] )</td>
<td>( U[0.3, 0.7] )</td>
<td>( U[0.2, 0.41] )</td>
</tr>
</tbody>
</table>
The notation $U[c_1, c_2]$ in Table A.1 stands for uniform distribution in the interval $[c_1, c_2]$ with probability density $1/(c_2 - c_1)$.

We see that the classification information comes only from the parameter $a$. The Bayes error on the parameter space is 10%. The other two parameters do not offer useful classification information. The estimated Bayes error on the sample space should be also 10%.

3). Gaussian pulse data set II:

This data set is also generated with (A.2). The difference is that this time all the three parameters give useful classification information. The following table shows the distribution of these parameters.

<table>
<thead>
<tr>
<th></th>
<th>class 1</th>
<th>class 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a$</td>
<td>$N[5, 0.2]$</td>
<td>$N[5, 2]$</td>
</tr>
<tr>
<td>$m$</td>
<td>$N[1, 0.1]$</td>
<td>$N[1, 1]$</td>
</tr>
<tr>
<td>$\sigma$</td>
<td>$N[1, 0.02]$</td>
<td>$N[1, 0.21]$</td>
</tr>
</tbody>
</table>

The notation $N(c_1, c_2)$ in Table A.2. stands for Gaussian distribution with mean $c_1$ and variance $c_2$. The Bayes error on the parameter space is 9.6%. Since the parameters go through nonlinear transformation, the observed output is no longer Gaussian, and thus, the Bayes error on the sample space might not be 9.6%. It could be somewhat higher if the ranges of the parameters are not properly chosen or $n$ is not large enough.
III. Radar-32 Data:

This real data set, provided by Navy, is the range-profile high-resolution millimeter-wave radar data consisting of two classes with unknown distribution. The dimension n is 32. However, the intrinsic dimension $n_e$ is estimated to be only around 13 using all the available samples. The number of samples available is about 6,700 for each class. This set of data has gone through energy normalization and power transform $y = x^\nu$ with $\nu = 0.4$, where $x$ and $y$ are the input and output pixels respectively. The actual Bayes error is unknown. The quadratic error is estimated to be around 14%. There are certain degree of correlation between adjacent samples due to the way these samples are generated. The Bayes error of this data set is estimated to be around 9% using our algorithm.

IV. Radar-64 Data:

This 64-dimensional range-profile radar signal data set, provided by Air Force, consists of two classes with 4,000 samples in each class. These samples are divided into four windows according to the azimuth and elevation angles. The intrinsic dimension is about 33. This data set has gone through the same power transform as the one used for the Radar-32 data. Although the real distribution is unknown, various statistical parameters suggest that this data is close to a Gaussian distribution. The quadratic error is estimated to be 20%. All the samples in this set are uncorrelated according to the data provider. We have also verified that the correlation between adjacent samples is indeed very trivial. The Bayes error of this data set is estimated to be around 18%.
V. ISAR Data:

The ISAR (Inverse Synthetic Aperture Radar) data is obtained by transforming the one-dimensional High Resolution Range-profile (HRR) data, which are made available by Lincoln Laboratory of Massachusetts Institute of Technology, to 2-dimensional images. The data were collected in the early 1980's. Placing the objects, Camaro and Dodge Van, on a turntable and rotating them by 360 degrees with a fixed depression angle, they recorded about eight thousand one-dimensional HRR data samples using a radar operating at 35GHz. The HRR data vectors contain both the real and imaginary parts of the received radar signals. Every 20 HRR signals are then grouped together and undergo a transformation (FFT and taking magnitude) to obtain a two-dimensional ISAR image. The ISAR images are magnitude only and do not contain any phase information. The ISAR image number 1 is formed by HRR data numbered 1 through 20, and ISAR image number 2 is obtained exactly the same way using HRR data numbered 2 through 21 and so forth. By shifting HRR data by one each time, we obtain a new 2-dimensional ISAR image. Therefore, we obtained about 8,000 ISAR images from the HRR data. However, there are phase jumps every 2 degrees of the viewing angle and each ISAR image should not consist of HRR with phase jumps. Thus, eliminating these images with phase jumps, we obtain only 4930 ISAR images out of the HRR data. These 4930 images are correlated because adjacent images share 19 samples of HRR data. Picking up every 20th image out of the 4930 images, we have 246 uncorrelated images. Every pixel of the images undergoes the same power transform as the one used for the Radar-32 data.

The original dimension of the ISAR data is 20x32. In order to speed up the analysis, we downsize the ISAR images. Experimentally, we found that averaging every neighboring four pixels into one pixel gives a better performance than simple downsampling. Using this approach, we downsize the ISAR data to 10x16 and 5x8 for intensive analysis.
Appendix B

Derivation of The Local Estimation of \( \frac{\nabla p(X)}{p(X)} \) and \( \frac{\nabla^2 p(X)}{p(X)} \)

Let

\[
L(X) = \{ Y: (Y-X)^T B^{-1}(Y-X) \leq \rho^2(n+2) \} \tag{B.1}
\]

be the local region built up by the metric B centered at \( X \), and

\[
M(X) = E\{ (Y-X) \mid Y \in L(X) \} \tag{B.2}
\]

be the local mean vector, and

\[
S(X) = E\{ (Y-X)(Y-X)^T \mid Y \in L(X) \} \tag{B.3}
\]

be the local autocorrelation matrix.

Consider

\[
M(X) = \int_{L(X)} \frac{(Y-X)p(Y)}{u(X)} dY, \tag{B.4}
\]

where \( u(X) \) is the probability mass in \( L(X) \), that is,

\[
u(X) = \int_{L(X)} p(Y) dY \tag{B.5}
\]

Using a Taylor expansion, \( p(Y) \) can be approximated by

\[
p(Y) \approx p(X) + \nabla p(X)(Y-X) + \frac{1}{2} (Y-X)^T \nabla^2 p(X)(Y-X) \tag{B.6}
\]

Plugging (B.6) into (B.4), we have

\[
M(X) \equiv \int_{L(X)} \frac{(Y-X)(Y-X)^T \nabla p(X)}{u(X)} dY \tag{B.7}
\]

Notice that

\[
\int_{L(X)} (Y-X)(Y-X)^T dY = v \rho^2 B \tag{B.8}
\]

where \( \rho \) and \( B \) are defined in (B.1), and \( v \) is the volume of \( L(X) \) expressed by

\[
v = \frac{\pi^{n/2} |B|^{1/2}}{\Gamma(n+2)} \left( \rho \sqrt{n+2} \right)^n. \tag{B.9}
\]
Combining (B.7) and (B.8) and using the approximation of \( u(X) \equiv v p(X) \), we have

\[
\frac{\nabla p(X)}{p(X)} \equiv \frac{B^{-1}}{\rho^2} M(X).
\]  

(B.10)

For the simplest case of \( B=I \),

\[
\frac{\nabla p(X)}{p(X)} \equiv \frac{1}{\rho^2} M(X)
\]  

(B.11)

That is, the local mean represents the local derivative normalized by \( p(X) \).

Now let us consider

\[
S(X) = \int_{L(X)} (Y - X)^T \frac{p(Y)}{u(X)} dY
\]  

(B.12)

Plugging (B.6) into (B.12), we have

\[
S(X) = \int_{L(X)} (Y - X)^T \frac{p(Y)}{u(X)} dY + \frac{1}{2} \int_{L(X)} (Y - X)^T \nabla^2 p(X)(Y - X) \frac{u(X)}{u(X)} dY.
\]  

(B.13)

Without loss of generality, we may let \( X=0 \) for simplicity. Thus from (W. 13), we have

\[
S(0) \equiv \frac{p(0)v(0)}{u(0)} \int_{L(0)} v(0) Y^T dY + \frac{1}{2} \frac{p(0)v(0)}{u(0)} \int_{L(0)} Y^T Y^T \nabla^2 p(0) Y \frac{1}{u(0)} dY
\]

where

\[
\frac{p(0)v(0)}{u(0)} \equiv 1.
\]  

(B.15)

In order to eliminate the correlation terms, and thus simplify the analysis, \( Y \) is linearly transformed to \( Z \) by

\[
Z = \Lambda^{-1/2} \Phi^T Y
\]  

(B.16)

where \( \Phi \) and \( A \) are the eigenvector and eigenvalue matrices of \( B \) respectively, and the eigenvectors are orthonormal, i.e., \( \Phi^T \Phi = I \).

The Jacobian of the linear transformation of (B16) is

\[
|J| = |\Lambda^{-1/2}| = |B|^{-1/2}
\]  

(B.17)

Changing variables in (B.14), we have
\[ S(0) = \Phi \Lambda^{1/2} \left| \int_{Z^T Z \in (\Re^2)^{n+1}} ZZ^T \frac{dZ}{v_Z} + \frac{1}{2} \int_{Z^T Z \in (\Re^2)^{n+1}} ZZ^T (Z^T \Lambda^{1/2} \Phi^T \frac{\nabla^2 p(0)}{p(0)} \Phi \Lambda^{1/2} Z) \frac{dZ}{v_Z} \right| \Lambda^{1/2} \Phi^T \]  

(B.18)

or

\[ S_Z = \Lambda^{-1/2} \Phi^T S(0) \Phi \Lambda^{-1/2} = \int_{Z^T Z \in (\Re^2)^{n+1}} ZZ^T \frac{dZ}{v_Z} + \frac{1}{2} \int_{Z^T Z \in (\Re^2)^{n+1}} ZZ^T (Z^T C_Z Z) \frac{dZ}{v_Z} \]  

(B.19)

where

\[ v_Z = |\mathbf{B}|^{-1/2} v(0) \]  

(B.20)

\[ C_Z = \Lambda^{1/2} \Phi^T \frac{\nabla^2 p(0)}{p(0)} \Phi \Lambda^{1/2} \]  

(B.21)

where \( C_Z \) is the counterpart of \( \frac{\nabla^2 p(0)}{p(0)} \) on the Z-domain.

The computation of the second term on the right hand side of (B.19) is demonstrated as follows:

\[ Z^T C_Z Z = \sum_{k=1}^{n} \sum_{l=1}^{n} c_{kl} z_k z_l \]  

(B.22)

where \( c_{kl} \) is the \( kl \)-th component of \( C_Z \).

The second term on the right hand side of (B.19) involves the fourth power of the radius of the hypersphere. Thus, it is appropriate to rewrite it as

\[ \frac{1}{2} \int_{Z^T Z \in (\Re^2)^{n+1}} ZZ^T (Z^T C_Z Z) \frac{dZ}{v_Z} = \frac{1}{2} \rho^4 W \]  

(B.23)

where \( W \) is a matrix to be determined.

The fourth order moments of a uniform distribution in a hyperspherical region are known as

\[ \int_{Z^T Z \in (\Re^2)^{n+1}} z_i^2 z_j^2 \frac{dZ}{v_Z} = \frac{n+2}{n+4} \rho^4 \]  

(B.24)

\[ \int_{Z^T Z \in (\Re^2)^{n+1}} z_i^4 \frac{dZ}{v_Z} = \frac{2(n+2)}{n+4} \rho^4 \]  

(B.25)

and all others are zero. Thus we have
where \( w_{ij} \) is the \( ij \)-th element of \( W \). From (B.26) and (B.27), we find

\[
W = \frac{2(n+2)}{n+4} C_Z + \frac{n+2}{n+4} (\text{tr}C_Z) I .
\]  

(B.28)

Thus, (B.19) becomes

\[
S_Z = \rho^2 I + \frac{1}{2} \rho^4 W .
\]  

(B.29)

Taking the trace of (B.28), we have

\[
\text{tr}W = \frac{2(n+2)}{n+4} \text{tr}C_Z + \frac{n(n+2)}{n+4} \text{tr}C_Z = \frac{(n+2)^2}{n+4} \text{tr}C_Z .
\]  

(B.30)

Plugging (B.30) into (B.29) and taking the trace, we have

\[
\text{tr}S_Z = np^2 + \frac{1}{2} \rho^4 \left( \frac{(n+2)^2}{n+4} \text{tr}C_Z \right) .
\]  

(B.31)

Thus, from (B.28), (B.30) and (B.31), we have

\[
C_Z = \frac{n+4}{\rho^4(n+2)} \left( S_Z - (\rho^2 + \frac{\text{tr}S_Z - \rho^2}{n+2}) I \right) .
\]  

(B.32)

Transforming from Z-space back to the original X-space, we have the final expression for the second order derivative of the probability density function

\[
\frac{\nabla^2 p(X)}{p(X)} = \frac{n+4}{\rho^4(n+2)} \left( B^{-1} S_X B^{-1} - \left( \frac{\text{tr}(S_X B^{-1})}{n+2} + 2\rho^2 \right) B^{-1} \right) .
\]  

(B.33)

For \( B=I \), we have

\[
\frac{\nabla^2 p(X)}{p(X)} = \frac{n+4}{\rho^4(n+2)} \left[ S_X - \left( \frac{\text{tr}S_X + 2\rho^2}{n+2} \right) I \right] .
\]  

(B.34)

We can rewrite (B.34) as

\[
\frac{\nabla^2 p(X)}{p(X)} = \alpha(S_X - \beta I)
\]  

(B.35)

where
\[ \alpha = \frac{n+4}{\rho^4(n+2)} \quad (B.36) \]
\[ \beta = \frac{tr(S_X) + 2\rho^2}{n+2} \quad (B.37) \]

The parameter \( \beta \) has a very important physical meaning that we would like to point out. In order to get this insight, we perform the first order approximation of (B.19)

\[ S_X \approx \rho^2 I. \quad (B.38) \]

Taking the trace of (B.38),

\[ trS_X \approx np^2. \quad (B.39) \]

Thus,

\[ \rho^2 \equiv \frac{trS_X}{n} = \text{average of the eigenvalues of } S_X. \quad (B.40) \]

Plugging (B.39) into (B.37), we have \( \beta \equiv \rho^2 \). Hence we may interpret \( \beta \) as the average of the eigenvalues of \( S_X \). In practice, since the probability density is unknown, the autocorrelation \( S_X \) is approximated by the sample autocorrelation

\[ \hat{S}_X(X) = \frac{1}{I_S} \sum_{j=1}^{I_S} (X_{jNN} - X)(X_{jNN} - X)^T \quad (B.41) \]

where \( I_S \) is the number of nearest neighbors in \( L(X) \).
Appendix C
Derivation of The Trace Equalization Algorithm

1). Different metrics:

Our goal is to obtain metrics $A_1$ and $A_2$ by sustaining the relationship $\text{tr}(K_1A_1) = \text{tr}(K_2A_2)$. Moving the term on the right-hand side to the left-hand side of the equation, we have

$$\text{tr}(K_1A_1 - K_2A_2) = 0. \quad (C.1)$$

Let

$$K_i = \sum_{j=1}^{n} \lambda_{ij} \phi_i \phi_j^T, \quad i=1,2 \quad (C.2)$$

$$A_i = \sum_{j=1}^{n} a_{ij} \phi_i \phi_j^T, \quad a_{ij} > 0, \quad i=1,2. \quad (C.3)$$

Then (C.1) can then be rewritten as

$$\sum_{\lambda_{ij} > 0} a_{1j} - \sum_{\lambda_{ij} < 0} a_{1j} - \sum_{\lambda_{ij} > 0} a_{2j} + \sum_{\lambda_{ij} < 0} a_{2j} = 0 \quad (C.4)$$

These coefficients $a_{ij}$ can be determined by minimizing

$$J = \sum_{j=1}^{n} (a_{1j} - 1)^2 + \sum_{j=1}^{n} (a_{2j} - 1)^2 - \mu \left[ \sum_{\lambda_{ij} > 0} a_{1j} - \sum_{\lambda_{ij} < 0} a_{1j} - \sum_{\lambda_{ij} > 0} a_{2j} + \sum_{\lambda_{ij} < 0} a_{2j} \right] \quad (C.5)$$

where $\mu$ is the Lagrangian multiplier. The constant 1 is included in (C.5) for we wish to have those $a_{ij}$ be as close to 1 as possible. Differentiating (C.5) relative to $a_{ij}$ and equating the derivatives to zero, we have

1. for $\lambda_{1j} > 0 \Rightarrow 2(a_{1j} - 1) - \mu = 0 \Rightarrow a_{1j} = 1 + \frac{\mu}{2} \quad (C.6)$

2. for $\lambda_{1j} < 0 \Rightarrow 2(a_{1j} - 1) + \mu = 0 \Rightarrow a_{1j} = 1 - \frac{\mu}{2} \quad (C.7)$

3. for $\lambda_{2j} > 0 \Rightarrow 2(a_{2j} - 1) - \mu = 0 \Rightarrow a_{2j} = 1 - \frac{\mu}{2} \quad (C.8)$
4. for $\lambda_{2j} < 0 \Rightarrow 2(a_{2j} - 1) - \mu = 0 \Rightarrow a_{2j} = 1 + \frac{\mu}{2}$ (C.9)

Plugging (C.6)-(C.9) to (C.4), we obtain

$$\sum_{\lambda_{i,j} > 0} a_{1j} - \sum_{\lambda_{i,j} < 0} a_{2j} + \sum_{\lambda_{i,j} < 0} a_{2j} = (1 + \frac{\mu}{2})n_{1p} - (1 - \frac{\mu}{2})n_{1n} - (1 - \frac{\mu}{2})n_{2p} + (1 + \frac{\mu}{2})n_{2n} = 0$$

(C.10)

where $n_{ip}$ and $n_{in}$ stand for the number of positive and negative eigenvalues for $K_i$ respectively. Rearranging (C.10), we have

$$\frac{\mu}{2} (n_{1p} + n_{1n} + n_{2p} + n_{2n}) = n_{1n} - n_{1p} + n_{2p} - n_{2n}.$$ (C.11)

Thus

$$\frac{\mu}{2} = \frac{n_{1n} - n_{1p} + n_{2p} - n_{2n}}{n_{1p} + n_{1n} + n_{2p} + n_{2n}}.$$ (C.12)

If $K_i$ is full rank, i.e., $n = n_{ip} + n_{in}$, (C.12) can be further simplified to

$$\frac{\mu}{2} = \frac{1}{2n} (n_{1n} - n_{1p} + n_{2p} - n_{2n}).$$ (C.13)

From (C.13) it is easy to see that $-1 \leq \frac{\mu}{2} \leq 1$. Note that $\frac{\mu}{2} = \pm 1$ implied that all the eigenvalues are either positive or negative. Therefore, $a_{ij} > 0$ for all $i$ and $j$. The positive definiteness of $A$, is retained throughout this process.

2). Common metric:

If a common metric $A$ is used, i.e., $A = A_i = A$, (C.1) becomes

$$\text{tr}((K_1 - K_2)A) = 0,$$ (C.14)

where

$$K_1 - K_2 = N \sum_{j=1}^{G} \lambda_j \phi_j \phi_j^T$$ (C.15)
\[ A = \sum_{j=1}^{n} \frac{a_j}{\lambda_j} \phi_j \phi_j^T \]  

By minimizing

\[ J = \sum_{j=1}^{n} (a_j - 1)^2 - \mu \left[ \sum_{\lambda_j > 0} a_j - \sum_{\lambda_j < 0} a_j \right] \]

we can obtain \( a_i \). Performing differentiation as before, we can easily show that

1. for \( \lambda_j > 0 \), \( a_j = 1 + \frac{\mu}{2} \)  
2. for \( \lambda_j < 0 \), \( a_j = 1 - \frac{\mu}{2} \).

Plugging (C.15), (C.16), (C.18) and (C.19) to (C.14), we have

\[ \sum_{\lambda_j > 0} a_j - \sum_{\lambda_j < 0} a_j = (1 + \frac{\mu}{2})n_p - (1 - \frac{\mu}{2})n_n = 0 \]

The above equation leads to

\[ \frac{\mu}{2} = \frac{n_n - n_p}{n_n + n_p} . \]

If \( K_1 - K_2 \) is full rank, i.e., \( n = n_n + n_p \), (C.21) can be further simplified to

\[ \frac{\mu}{2} = \frac{n_n - n_p}{n} . \]

From (C.23), it is easy to see that \(-1 \leq \frac{\mu}{2} \leq 1\), which ensures \( a_j > 0 \) for all \( j \).
Appendix D
The Proof of Theorem 2.1

Theorem 2.1:

Let $Q$ and $R$ both be symmetric $n \times n$ matrices, with $Q$ being positive definite. Then $QR$ and $R$ have the same number of positive and negative eigenvalues.

(Proof):

Since $Q$ is positive definite and symmetric, we can find a matrix $W$ that satisfies

$$Q = W^2$$

where $W$ is also positive definite and symmetric. We can diagonalize $W$ as

$$W = \Phi^T \Lambda \Phi$$

where $\Phi$ is orthonormal, and $\Lambda = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$, where $\lambda_i > 0$ for $i = 1, 2, \ldots, n$. Therefore,

$$Q = WW = \Phi^T \Lambda \Phi \Phi^T \Lambda \Phi = \Phi^T \Lambda^2 \Phi$$

Fact 1: $QR = WWR$ and $WRW$ have same eigenvalues.

(proof) Let $\mu$ be an eigenvalue of $QR$. The eigenvalues can be obtained by setting

$$|WWR - \mu I| = 0 .$$

Substituting $I$ with $W^{-1}$, (D4) can be rewritten as

$$|W(WR - \mu W^{-1})| = 0 .$$

Since $|W| \neq 0$, (D5) can be simplified to

$$|WR - \mu W^{-1}| = 0 .$$

Similarly, let $\mu'$ be an eigenvalue of $WRW$. The eigenvalues can be obtained by setting

$$|WRW - \mu' W| = 0 .$$
Again, substituting $I$ with $W^{-1}W$, (D.7) can be rewritten as
\[
|WR - \mu W^{-1}| = 0
\]  
(D.8)
and (D.8) can be further simplified to
\[
|WR - \mu W^{-1}| = 0
\]  
(D.9)
which is exactly the same as (D.6).

Thus we have shown that the eigenvalues of $WWR$ are the same as those of $WRW$.

Fact 2: $WRW = W^R RW$. This is obvious because $W = W^T$.

Fact 3: Since $R$ and $W^R RW$ are congruent and both are symmetric, $R$ and $W^R RW$ have the same number of positive and negative eigenvalues by virtue of the Sylvester's Law of Inertia [18].

Thus from Fact 1 through Fact 3, we conclude that $R$ and $QR$ have the same number of positive and negative eigenvalues.
Appendix E
The Proof of Theorem 4.1

Theorem 4.1:

For given nx1 vectors $X_i, i=1,2,\ldots,N$, let $\hat{M}$ and $\hat{\Sigma}$ be the sample mean and the sample covariance. If $N<n$, then

$$
(X_i - \hat{M})^T \hat{\Sigma}^{-1} (X_i - \hat{M}) = \frac{(N-1)^2}{N} \quad i=1,2,\ldots,N
$$

where $\hat{\Sigma}^{-1}$ is the pseudo-inverse of $\hat{\Sigma}$, obtained by

$$
\hat{\Sigma}^{-1} = \sum_{i=1}^{N-1} \frac{1}{\lambda_i} \phi_i \phi_i^T
$$

where $\lambda_i$ and $\phi_i$ are the nonzero eigenvalues and corresponding eigenvectors of $\hat{\Sigma}$.

(Proof):

Let $U=[X_1 X_2 \ldots X_N]$ be the sample matrix. Without loss of generality, we can assume that these samples to have the sample mean $\hat{M} = 0$ for notational simplicity. Therefore, the sample covariance matrix $\hat{\Sigma}$ can be obtained by

$$
\hat{\Sigma} = \frac{1}{N-1} U U^T
$$

We can find the eigenvectors and eigenvalues of $\hat{\Sigma}$ by

$$
\left(\frac{1}{N-1} U U^T\right) \Phi = \Phi \Lambda
$$

where $\Lambda_{(N-1)\times(N-1)}$ is the matrix of $N-1$ nonzero eigenvalues and $\Phi_{N\times(N-1)}$ is the matrix of the corresponding $(N-1)$ eigenvectors. Multiplying $U^T$ from the left side, we can convert (E.4) to

$$
(U^T U) (U^T \Phi) = (U^T \Phi) [(N-1) \Lambda].
$$

Since $U^T \Phi$ is not orthonormal, we can change the scale of the eigenvectors by
\[ \Psi = U^T \Phi (N-1) \Lambda^{-1/2} . \]  
(E.6)

It is easy to verify that \( \Psi^T \Psi = I \). From (E.6), \( U \) can be expressed as

\[ U = \sqrt{N-1} \Phi \Lambda^{1/2} \Psi^T . \]  
(E.7)

Therefore, by (E.7) and (E.4)

\[ U^T \Sigma^* U = (N-1) \Psi \Lambda^{1/2} \Phi \Lambda^{-1} \Phi^T \Phi \Lambda^{1/2} \Psi^T \]

\[ = (N-1) \Psi \Psi^T . \]  
(E.8)

Let \( 1_N \) be an \( N \times 1 \) column vector with every entry equal to 1. Since the sample mean \( \hat{\mu} \) of the sample matrix \( U \) is zero by assumption, we have

\[ \hat{\mu} = \frac{1}{N} U 1_N \]

\[ = \sqrt{N-1} \frac{\Phi \Lambda^{1/2} \Psi^T 1_N}{N} \]

\[ = \sqrt{N-1} \frac{\sum_{i=1}^{N-1} (\psi_i^T 1_N \sqrt{\lambda_i} \phi_i)}{N} \]

\[ = 0 \]  
(E.9)

where \( \psi_i \) and \( \phi_i \) are the column vectors of \( \Psi \) and \( \Phi \) respectively, and \( \lambda_i \) are the diagonal terms of \( \Phi \), \( i = 1, 2, \ldots, N-1 \). Since \( \phi_i \) are linearly independent vectors and \( \lambda_i \neq 0 \), we have

\[ \psi_i^T 1_N = 0 \quad i = 1, 2, \ldots, N-1 . \]  
(E.10)

From (E.9) it is easy to see that \( 1_N \) is the eigenvector for \( U^T U/(N-1) \) with eigenvalue=0. That is,

\[ \frac{1}{N-1} U^T U 1_N = 0 \]  
(E.11)

Let

\[ \psi_N = \frac{1_N}{\sqrt{N}} . \]  
(E.12)

Since \( \psi_N \) is an orthonormal eigenvector of \( U^T U/(N-1) \), the augmented matrix \( [I \psi_N] \), an \( N \times N \) matrix, contains a complete set of orthonormal eigenvectors. Therefore, this augmented matrix has the following properties:
\[ [\psi \quad \psi_N]^T[\psi \quad \psi_N] = I \quad \text{(E.13)} \]

and
\[ [\psi \quad \psi_N][\psi \quad \psi_N]^T = \psi\psi^T + \psi_N\psi_N^T = I. \quad \text{(E.14)} \]

From (E.14), we have
\[ YY^T = I - \psi_N^T \psi_N \]
\[ = I - \frac{1}{N} 1_N 1_N^T \quad \text{(E.15)} \]

Plugging (E.15) back into (E.8), we obtain
\[ U^* \hat{\Sigma} U = (N - 1)(I - \frac{1}{N} 1_N 1_N^T). \quad \text{(E.16)} \]

The distances, \( X_i^T \hat{\Sigma}^* X_i, i=1,2,\ldots,N \), are simply the diagonal terms of \( U^* \hat{\Sigma} U \). It is easy to verify that the diagonal terms of \( U^* \hat{\Sigma} U \) are all the same and equal to \((N-1)^2/N\). Therefore, we have
\[ X_i^T \hat{\Sigma}^* X_i = \frac{(N-1)^2}{N} \quad i=1,2,\ldots,N. \quad \text{(E.17)} \]