RICE UNIVERSITY

Issues Related to Data Mining with Self Organizing Maps

by

Abha Jain

A Thesis Submitted
IN PARTIAL FULFILLMENT OF THE
Requirements for the Degree
MASTER OF SCIENCE

APPROVED, THESIS COMMITTEE:

[Signature]
Dr. Erzsébet Merényi
Research Professor in Electrical and
Computer Engineering

[Signature]
Dr. John Clark, Chair
Professor in Electrical and Computer
Engineering

[Signature]
Dr. Devika Subramanian
Professor in Electrical and Computer
Engineering

[Signature]
Dr rer. nat. Thomas Villmann
Clinic for Psychotherapy, University of
Leipzig

Houston, Texas
April, 2004
INFORMATION TO USERS

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleed-through, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

UMI Microform 1419090
Copyright 2004 by ProQuest Information and Learning Company.
All rights reserved. This microform edition is protected against unauthorized copying under Title 17, United States Code.
ABSTRACT

Issues Related to Data Mining with Self Organizing Maps

by

Abha Jain

This thesis demonstrates that the clustering by Kohonen’s Self-Organizing Map algorithm (KSOM) can be significantly improved by magnification control. As a second contribution, the thesis proposes a fully automated technique for clustering the prototypes of the data (SOM weights). The motivation for this work comes from a serious need for effective, precise, and detailed knowledge discovery (clustering) for complex, high-dimensional data that are encountered in a variety of important applications such as remote sensing, medical imaging, etc. While many conventional clustering methods may fail to handle such data, modifications of KSOM show promise. We analyze the performance of one such advanced modification, the magnification control algorithm by Bauer, Der and Herrmann [13] to determine its scope. By magnification control, different clustering criteria can be optimized. We also demonstrate that negative magnification improves the detectability of rare classes in the data potentially leading to discovery of new classes.
Acknowledgments

I would like to thank my advisor Dr. Erzsébet Merényi and the members of the committee for their help and constructive suggestions.

I would also like to thank Mr. Philip Tracadas for his assistance with the softwares used in course of this research.

I am also very grateful to my family and friends for their unfailing and much needed support through out my Masters.
Contents

Abstract ii
Acknowledgments iii
List of Figures ix
List of Tables xxix

1 Background 1

1.1 The Self-Organizing Map .................. 1
1.2 Why SOM? ................................ 4
1.3 Magnification control ....................... 8
1.4 Focus of the present work .................. 11
1.5 Our approach ................................ 12
1.6 Cluster identification in the SOM .......... 12

2 Magnification control for 1-dimensional data 14

2.1 Description of 1-dimensional data sets .......... 14
2.1.1 Data Set I .................................. 14
2.1.2 Data Set II .................................. 15
2.2 Procedure for evaluating $\alpha_{achieved}$ .......... 15
2.3 Simulations on 1-dimensional data sets .......... 18
2.3.1 Magnification control on Data Set I (pdf: $p(v) = 2v$) .... 18
2.3.2 Magnification control on Data Set II (pdf: $p(v) = 3v^2$) ... 23
2.4 Comparison of Conscience algorithm and BDH with $\alpha_{desired} = 1.0$ .. 28
  2.4.1 Conscience algorithm ...................................... 28
  2.4.2 Conscience with zero neighborhood ......................... 30
  2.4.3 Conscience with a neighborhood of 1 around the winner ... 32
2.5 BDH using periodic SOM and periodic data ..................... 37
2.6 One- and two-step procedures for magnification control ....... 39
2.7 Conclusions on magnification control for 1-dimensional data ... 39

3 Magnification control for 2-dimensional data ................. 43
  3.1 Data independent in the two dimensions ...................... 44
    3.1.1 Data description ........................................ 45
    3.1.2 Simulation results ...................................... 45
  3.2 Weakly correlated data in two dimensions: $\rho_{v_1v_2} \ll 1$ ...... 47
    3.2.1 Data description ........................................ 48
    3.2.2 Simulation results ...................................... 48
  3.3 Strongly correlated data in two dimensions: $\rho_{v_1v_2} \approx 1$ .... 49
    3.3.1 Data description ........................................ 50
    3.3.2 Simulation results ...................................... 51
  3.4 Conclusions on magnification control for 2-dimensional data ... 53
4 Magnification control for higher dimensional data

4.1 Evaluation of BDH on noiseless, 6-dimensional, 5- and 20-class data sets

4.1.1 Description of the noiseless 5- and 20-class data sets

4.1.2 Simulations on the noiseless 5- and 20-class data sets

4.2 Evaluation of BDH on noisy 6-dimensional, 8- and 11-class data sets

4.2.1 Description of the noisy 8- and 11-class data sets

4.2.2 Procedure for evaluating $\alpha_{achieved}$

4.2.3 Simulations on the noisy 8- and 11-class data sets

4.2.4 Noisy 8-class data set

4.2.5 Noisy 11 class data set

4.3 Conclusions on magnification control for high-dimensional data

5 Magnification control for real, high-dimensional data (multi-spectral imagery)

5.1 Description of the Ocean City multi-spectral image

5.2 Simulation results

5.2.1 Class $V$

5.2.2 Class $a$

5.2.3 Class $C$

5.3 Conclusion
6 Automatic clustering of the SOM weights

6.1 Background .............................................. 89
6.2 Technique developed by us ......................... 93
6.3 Simulation results ..................................... 106
   6.3.1 Data Set I ........................................... 106
   6.3.2 Data Set II .......................................... 108
   6.3.3 Noisy 8-class data set ............................ 110
   6.3.4 Noisy 11-class data set ......................... 112

7 Conclusions ................................................. 114

A On the evaluation of the magnification exponent of an SOM

A.1 Introduction ............................................. 117
A.2 A histogram based method for evaluation of $\alpha$ ........... 118
A.3 Determination of bin width in a 1-dimensional case .......... 119
   A.3.1 Normal reference rule for 1-dimensional data .......... 120
A.4 The multivariate histogram ............................. 122
   A.4.1 Normal reference rule for multivariate data ......... 123
   A.4.2 A special case: $d = 2$ with nonzero correlation .... 124
A.5 Curse of dimensionality ................................. 124
A.6 Conclusion .............................................. 126
References
List of Figures

1.1 SOM - an array of PEs. Each input pattern is connected to all the Processing Elements (PEs). The weights associated with each PE have the same dimensionality as that of the input vectors. .............................. 2

1.2 Clustering using SOMs is a two-step process. First prototypes (representatives) of the data are formed in such a way that they are ordered in a topology preserving way on a two-dimensional grid. In the second step, these prototypes are clustered. Each input vector then belongs to the same cluster as its prototype. (Each square in the grid represents a PE which has a weight vector associated with it. The fence between two PEs represents the Euclidean distance between the weights associated with those PEs) ................................. 4

2.1 Left: Distribution of Data Set I. Right: Distribution of Data Set II. .............................. 15

2.2 Initial and final weights when BDH was applied on Data Set I with $\alpha_{desired} =$ 1.0. Left: Initial weights which were chosen randomly. Right: Final weights are ordered, indicating that the mapping is topology preserving. .............................. 18
2.3 Result of applying BDH on Data Set I with $\alpha_{desired} = 1$. Number of weights in each bin: theoretical ($n_i^{theoretical}$) for $\alpha = 1$, those obtained by BDH ($n_i$) with $\alpha_{desired} = 1$ and what ideally would be obtained by applying KSOM. In this case, $n_i^{theoretical}$ and $n_i$ are quite close. The value of $\alpha$ that gives the minimum value of the sum $\sum_{i=0}^{9} (n_i - n_i^{theoretical})^2$, is $\alpha_{achieved}$. Here, $\alpha_{achieved} = 0.982482$.

2.4 Initial and final weights when BDH was applied on Data Set I with $\alpha_{desired} = 0.3$. **Left:** Initial weights which were chosen randomly. **Right:** Final weights are ordered, therefore the mapping is topology preserving.

2.5 Result of applying BDH on Data Set I with $\alpha_{desired} = 0.3$. Number of weights in each bin: theoretical ($n_i^{theoretical}$) for $\alpha = 0.3$, those obtained by BDH ($n_i$) with $\alpha_{desired} = 0.3$ and what would ideally be obtained by applying KSOM. In this case, $n_i^{theoretical}$ and $n_i$ are quite close. The value of $\alpha$ that gives the minimum value of the sum $\sum_{i=0}^{9} (n_i - n_i^{theoretical})^2$, is $\alpha_{achieved}$. Here, $\alpha_{achieved} = 0.322523$.

2.6 Initial and final weights when BDH was applied on Data Set I with $\alpha_{desired} = -0.5$.

**Left:** Initial Weights which were chosen randomly. **Right:** Final weights are ordered, indicating that the mapping is topology preserving.
2.7 Result of applying BDH on Data Set I with $\alpha_{\text{desired}} = -0.5$. **Left:** Number of weights in each bin: theoretical ($n_i^{\text{theoretical}}$) for $\alpha = -0.5$, those obtained by BDH ($n_i$) with $\alpha_{\text{desired}} = -0.5$ and what ideally would be obtained by applying KSOM. In this case, $n_i^{\text{theoretical}}$ and $n_i$ are quite close. The value of $\alpha$ that gives the minimum value of the sum $\sum_{i=0}^{9}(n_i - n_i^{\text{theoretical}})^2$, is $\alpha_{\text{achieved}}$. Here, $\alpha_{\text{achieved}} = -0.122122$. **Right:** Number of weights in each bin: theoretical for $\alpha = 1$ and those obtained by BDH with $\alpha_{\text{desired}} = -0.5$. The two plots exhibit an inverse relationship. Since the input pdf is directly proportional to the weight density when $\alpha_{\text{desired}} = 1$, the inverse relation between the plots in the figure shows that the input pdf and the weight density obtained on applying BDH with $\alpha_{\text{desired}} < 0$ are inversely proportional.

2.8 Performance of BDH algorithm on Data Set I: Plot comparing $\alpha_{\text{achieved}}$ and $\alpha_{\text{desired}}$.

BDH performs best for $0 \leq \alpha_{\text{desired}} \leq 1.0$.

2.9 Initial and final weights when BDH was applied on Data Set II with $\alpha_{\text{desired}} = 1.0$.

**Left:** Initial Weights which were chosen randomly. **Right:** Final weights are ordered indicating that the mapping is topology preserving.
2.10 Result of applying BDH on Data Set II with $\alpha_{\text{desired}} = 1$. Number of weights in each bin: theoretical ($n_i^{\text{theoretical}}$) for $\alpha = 1$, those obtained by BDH ($n_i$) with $\alpha_{\text{desired}} = 1$ and what would ideally be obtained by applying KSOM. In this case, $n_i^{\text{theoretical}}$ and $n_i$ are quite close. The value of $\alpha$ that gives the minimum value of the sum $\sum_{i=0}^{9}(n_i - n_i^{\text{theoretical}})^2$, is $\alpha_{\text{achieved}}$. Here, $\alpha_{\text{achieved}} = 1.022523$. 

2.11 Initial and final weights when BDH was applied on Data Set II with $\alpha_{\text{desired}} = 0.4$.

**Left:** Initial Weights which were chosen randomly. **Right:** Final weights are ordered indicating that the mapping is topology preserving.

2.12 Result of applying BDH on Data Set II with $\alpha_{\text{desired}} = 0.4$. Number of weights in each bin: theoretical ($n_i^{\text{theoretical}}$) for $\alpha = 0.4$, those obtained by BDH ($n_i$) with $\alpha_{\text{desired}} = 0.4$ and what ideally would be obtained by applying KSOM. In this case, $n_i^{\text{theoretical}}$ and $n_i$ are quite close. The value of $\alpha$ that gives the minimum value of the sum $\sum_{i=0}^{9}(n_i - n_i^{\text{theoretical}})^2$, is $\alpha_{\text{achieved}}$. Here, $\alpha_{\text{achieved}} = 0.422523$.

2.13 Initial and final weights when BDH was applied on Data Set II with $\alpha_{\text{desired}} = -0.3$.

**Left:** Initial Weights which were chosen randomly. **Right:** Final weights are ordered, therefore the mapping is topology preserving.
2.14 Result of applying BDH on Data Set II with $\alpha_{\text{desired}} = -0.3$. **Left:** Number of weights in each bin: theoretical ($n_i^{\text{theoretical}}$) for $\alpha = -0.3$, those obtained by BDH ($n_i$) with $\alpha_{\text{desired}} = -0.3$ and what ideally would be obtained by applying KSOM. In this case, $n_i^{\text{theoretical}}$ and $n_i$ are quite close. The value of $\alpha$ that gives the minimum value of the sum $\sum_{i=0}^{9}(n_i - n_i^{\text{theoretical}})^2$, is $\alpha_{\text{achieved}}$. Here, $\alpha_{\text{achieved}} = -0.015616$. **Right:** Number of weights in each bin: theoretical for $\alpha = 1$ and those obtained by BDH with $\alpha_{\text{desired}} = -0.3$. The two plots exhibit an inverse relationship. Since the input pdf is directly proportional to the weight density when $\alpha_{\text{desired}} = 1$, the inverse relation between the plots in the figure shows that the input pdf and the weight density obtained on applying BDH with $\alpha_{\text{desired}} < 0$ are inversely proportional.

2.15 Performance of BDH algorithm on Data Set II: Plot comparing $\alpha_{\text{achieved}}$ and $\alpha_{\text{desired}}$. BDH performs best for $0.1 \leq \alpha_{\text{desired}} \leq 1.0$.

2.16 Final weights obtained for Data Set I after 1000000 iterations. **Left:** Those obtained by Conscience algorithm (with zero neighborhood) are not ordered. **Right:** Those obtained by BDH algorithm (with $\alpha_{\text{desired}} = 1$) are ordered.

2.17 Comparing the results after 100000 iterations for Data Set I. **Left:** $\alpha_{\text{achieved}} = 0.7773$ by Conscience algorithm (with zero neighborhood). **Right:** $\alpha_{\text{achieved}} = 0.9825$ by BDH (with $\alpha_{\text{desired}} = 1$).
2.18 Comparing the results after 1000000 iterations for Data Set I. **Left:** $\alpha_{achieved} = 0.9785$ by Conscience algorithm (with zero neighborhood). **Right:** $\alpha_{achieved} = 0.9995$ by BDH (with $\alpha_{desired} = 1$).

2.19 Final weights obtained for Data Set II after 1000000 iterations. **Left:** Those obtained by Conscience algorithm (with zero neighborhood) are not ordered. **Right:** Those obtained by BDH algorithm (with $\alpha_{desired} = 1$) are ordered.

2.20 Comparing the results after 100000 iterations for Data Set II. **Left:** $\alpha_{achieved} = 0.7282$ by Conscience algorithm (with zero neighborhood). **Right:** $\alpha_{achieved} = 1.0225$ by BDH (with $\alpha_{desired} = 1$).

2.21 Comparing the results after 1000000 iterations for Data Set II. **Left:** $\alpha_{achieved} = 0.9805$ by Conscience algorithm (with zero neighborhood). **Right:** $\alpha_{achieved} = 1.0275$ by BDH (with $\alpha_{desired} = 1$).

2.22 Final weights obtained for Data Set I after 1000000 iterations. **Left:** Those obtained by Conscience algorithm (with a neighborhood of 1 around the winner) are only locally ordered. **Right:** Those obtained by BDH algorithm (with $\alpha_{desired} = 1$) are ordered globally.

2.23 Comparing the results after 100000 iterations for Data Set I. **Left:** $\alpha_{achieved} = 0.9585$ by Conscience algorithm (with a neighborhood of 1 around the winner). **Right:** $\alpha_{achieved} = 0.9825$ by BDH (with $\alpha_{desired} = 1$).
2.24 Comparing the results after 1000000 iterations for Data Set I. **Left:** $\alpha_{\text{achieved}} = 0.9404$ by Conscience algorithm (with a neighborhood of 1 around the winner). 

**Right:** $\alpha_{\text{achieved}} = 0.9995$ by BDH (with $\alpha_{\text{desired}} = 1$). ........................................... 35

2.25 Final weights obtained for Data Set II after 100000 iterations. **Left:** Those obtained by Conscience algorithm (with a neighborhood of 1 around the winner) are only partially ordered. **Right:** Those obtained by BDH algorithm (with $\alpha_{\text{desired}} = 1$) are globally ordered. ........................................... 36

2.26 Comparing the results after 100000 iterations for Data Set II. **Left:** $\alpha_{\text{achieved}} = 0.9334$ by Conscience algorithm (with a neighborhood of 1 around the winner).

**Right:** $\alpha_{\text{achieved}} = 1.0225$ by BDH (with $\alpha_{\text{desired}} = 1$). ........................................... 36

2.27 Comparing the results 1000000 iterations for Data Set II. **Left:** $\alpha_{\text{achieved}} = 0.9555$ by Conscience algorithm (with a neighborhood of 1 around the winner). **Right:**

$\alpha_{\text{achieved}} = 1.0275$ by BDH (with $\alpha_{\text{desired}} = 1$). ........................................... 37

2.28 Making the input space periodic. ........................................... 40

2.29 By utilizing the technique for making the SOM and the input space periodic undesirable boundary effects can be removed. ........................................... 41

2.30 Comparing one- and two-step procedures for magnification control by plotting $\alpha_{\text{achieved}}$ and $\alpha_{\text{desired}}$. **Left:** For Data Set I. **Right:** For Data Set II. ............... 41

3.1 2-dimensional data samples $\mathbf{v} = (v_1, v_2)$, such that $p_\mathbf{v}(\mathbf{v}) = p_{v_1}(v_1)p_{v_2}(v_2)$, i.e., data is independent in the two dimensions. ........................................... 44
3.2 2-dimensional input data samples (small dots) and distribution of SOM weights (larger dots), resulting from the application of the BDH algorithm. Weights belonging to adjacent PEs in the SOM are connected. 2-dimensional input samples \( \mathbf{v} = (v_1, v_2) \) are such that \( v_1 \) and \( v_2 \) are independent. \textbf{Left:} BDH applied with \( \alpha_{\text{desired}} = 0.6, \ \alpha_{\text{achieved}} = 0.6075 \). \textbf{Right:} BDH applied with \( \alpha_{\text{desired}} = 0.3, \ \alpha_{\text{achieved}} = 0.4046 \).

3.3 BDH performance when 2-D input samples \( \mathbf{v} = (v_1, v_2) \) are such that \( v_1 \) and \( v_2 \) are independent. \( \alpha_{\text{achieved}} \) and \( \alpha_{\text{desired}} \) match approximately. The discrepancies are largely due to the fact that the theoretical results are asymptotic and we only have a finite number of PEs (100) and also because the process of evaluation of \( \alpha_{\text{achieved}} \) is not very accurate.

3.4 2-dimensional data samples \( \mathbf{v} = (v_1, v_2) \), such that \( \rho_{v_1v_2} \ll 1 \), i.e., data in the two dimensions is weakly correlated.

3.5 2-dimensional input data samples (small dots) and distribution of SOM weights (larger dots), resulting from application of the BDH algorithm with \( \alpha_{\text{desired}} = 1.0 \). Weights belonging to adjacent PEs in the SOM are connected. 2-dimensional input samples \( \mathbf{v} = (v_1, v_2) \) are such that \( \rho_{v_1v_2} \ll 1 \). This is a case of "forbidden data" where supporting theory does not guarantee success. \( \alpha_{K_{\text{SOM}}} \) was found to be 0.789289. \textbf{Left:} BDH applied with \( m = \frac{3}{4} \alpha_{\text{desired}} - 1, \ \alpha_{\text{achieved}} = 0.992492 \). \textbf{Right:} BDH applied with \( m = \frac{1}{\alpha_{K_{\text{SOM}}}} \alpha_{\text{desired}} - 1, \ \alpha_{\text{achieved}} = 0.985485 \).
3.6 2-dimensional input data samples (small dots) and distribution of SOM weights (larger dots), resulting from application of the BDH algorithm with $\alpha_{\text{desired}} = 0.4$.

Weights belonging to adjacent PEs in the SOM are connected. 2-dimensional input samples $\mathbf{v} = (v_1, v_2)$ are such that $\rho_{v_1v_2} \ll 1$. This is a case of “forbidden data” where supporting theory does not guarantee success. $\alpha_{K_{\text{SOM}}}$ was found to be 0.789289. **Left:** BDH applied with $m = \frac{3}{2} \alpha_{\text{desired}} - 1$, $\alpha_{\text{achieved}} = 0.644745$.

**Right:** BDH applied with $m = \frac{1}{2} \alpha_{\text{desired}} - 1$, $\alpha_{\text{achieved}} = 0.552653$.

3.7 BDH performance when 2-D input samples $\mathbf{v} = (v_1, v_2)$ are such that 2-dimensional input samples $\mathbf{v} = (v_1, v_2)$ are such that $\rho_{v_1v_2} \ll 1$. This is a case of “forbidden data” where supporting theory does not guarantee success. $\alpha_{\text{achieved}}$ and $\alpha_{\text{desired}}$ are almost equal at $\alpha_{\text{desired}} = 1$ and the two values differ increasingly (but in a predictable manner) as $\alpha_{\text{desired}}$ decreases. This is a stronger result than available from theory, as the theory only guaranteed success if and only if $v_1$ and $v_2$ were independent. The match between $\alpha_{\text{achieved}}$ and $\alpha_{\text{desired}}$ is better when using experimentally evaluated $\alpha_{K_{\text{SOM}}}$ instead of $2/3$.

3.8 2-dimensional data samples $\mathbf{v} = (v_1, v_2)$, such that $\rho_{v_1v_2} \approx 1$, i.e., data in the two dimensions is strongly correlated.
3.9 2-dimensional input data samples (small dots) and distribution of SOM weights (larger dots), resulting from application of the BDH algorithm with $\alpha_{\text{desired}} = 1.0$.

Weights belonging to adjacent PEs in the SOM are connected. 2-dimensional input samples $v = (v_1, v_2)$ are such that $\rho_{v_1,v_2} \approx 1$. This is a case of “forbidden data” where supporting theory does not guarantee success. $\alpha_{K_{\text{SOM}}}$ was found to be 0.905405. **Left:** BDH applied with $m = \frac{3}{2} \alpha_{\text{desired}} - 1$, $\alpha_{\text{achieved}} = 1.262763$.

**Right:** BDH applied with $m = \frac{1}{\alpha_{K_{\text{SOM}}}} \alpha_{\text{desired}} - 1$, $\alpha_{\text{achieved}} = 0.972472$. . . . 51

3.10 2-dimensional input data samples (small dots) and distribution of SOM weights (larger dots), resulting from application of the BDH algorithm with $\alpha_{\text{desired}} = 0.4$.

Weights belonging to adjacent PEs in the SOM are connected. 2-dimensional input samples $v = (v_1, v_2)$ are such that $\rho_{v_1,v_2} \approx 1$. This is a case of “forbidden data” where supporting theory does not guarantee success. $\alpha_{K_{\text{SOM}}}$ was found to be 0.905405. **Left:** BDH applied with $m = \frac{3}{2} \alpha_{\text{desired}} - 1$, $\alpha_{\text{achieved}} = 0.701802$.

**Right:** BDH applied with $m = \frac{1}{\alpha_{K_{\text{SOM}}}} \alpha_{\text{desired}} - 1$, $\alpha_{\text{achieved}} = 0.591692$. . . . 52
3.11 BDH performance when 2-D input samples \( v = (v_1, v_2) \) are such that 2-dimensional input samples \( v = (v_1, v_2) \) are such that \( \rho_{v_1v_2} \ll 1 \). This is a case of "forbidden data" where supporting theory does not guarantee success. Here, even though \( \alpha_{\text{achieved}} \) by the map differs from \( \alpha_{\text{desired}} \), there is a clearly observable trend that the values of \( \alpha_{\text{achieved}} \) are systematically decreasing, following the desired values with a more or less constant shift. This is again a stronger result than the theory provides. The match between \( \alpha_{\text{achieved}} \) and \( \alpha_{\text{desired}} \) is better when using experimentally evaluated \( \alpha_{K_{\text{SOM}}} \) instead of 2/3.

4.1 Description of the noiseless 5- and 20-class data sets. Signatures of the classes and their distribution over the subareas of the image. **Left:** 5-class data set. **Right:** 20-class data set.

4.2 Results of clustering of the 5-class data set. **Left:** Using KSOM. Top) Weight vectors in the 10 \( \times \) 10 SOM. Only 1 PE represents the rare class U. Bottom Left) Clusters identified in the map. The darker the fence between two PEs, the smaller the difference between the corresponding weights. Bottom Right) This figure shows which class each weight vector is closest to, which complements the information on the left. **Right:** Using BDH with \( \alpha_{\text{desired}} = -0.8 \). Top) Weight vectors in the 10 \( \times \) 10 SOM. The rare class U is now represented by 10 PEs! Bottom Left) Clusters identified in the SOM. Bottom Right) This figure shows which class each weight vector is closest to, which complements the information on the left.
4.3 Results of clustering of the 20-class data set. **Left:** Using KSOM. Top) Weight vectors in the $10 \times 10$ map. Only 1 PE is occupied by each of the rare classes $R$ and $Q$. Bottom Left) Clusters identified in the SOM. The lighter the color of the fence between two PEs, the larger the difference between the corresponding weights. Bottom Right) This figure shows which class each weight vector is closest to, which complements the information on the left. **Right:** Using BDH with $\alpha_{desired} = -0.8$.

Top) Weight vectors in the $10 \times 10$ SOM. 4 and 7 PEs now represent the rare classes $R$ and $Q$ respectively. Bottom Left) Clusters identified in the SOM. Bottom Right) This figure shows which class each weight vector is closest to, which complements the information on the left.

4.4 Description of the noisy 8- and 11-class data sets. Mean signatures of the classes and their distribution over the subareas of the image. **Left:** 8-class data set. **Right:** 11-class data set.

4.5 SOMs resulting from the application of Conscience and BDH with $\alpha_{desired} = 1$ on noisy 8-class data set. The weight vectors of the PEs are plotted. The PE belonging to a particular class has the same color as that used to represent the class in Figure 4.4 (Left). **Left:** Result of applying BDH with $\alpha_{desired} = 1$, $\alpha_{achieved} = 0.995$. **Right:** Result of applying Conscience, $\alpha_{achieved} = 0.905$. Performance of BDH with $\alpha_{desired} = 1$ is better than that of Conscience.
4.6 Number of weights belonging to each class in the SOMs resulting from Conscience and BDH with $\alpha_{\text{desired}} = 1$ on noisy 8-class data set. $\text{Error}_{\text{BDH}} = 9.6954$ and $\text{Error}_{\text{Conscience}} = 11.2250$. The values of $\alpha_{\text{achieved}}$ were 0.995 and 0.905 for BDH and Conscience, respectively. So the performance of BDH with $\alpha_{\text{desired}} = 1$ is slightly better than that of Conscience. For both algorithms, the number of weights belonging to each class are close to but not exactly equal to ideal. This discrepancy is caused by the fact that the theoretical results are asymptotic and we only have a finite number of PEs. Integer arithmetic also leads to deviation from the ideal numbers.

4.7 SOMs resulting from the application of KSOM and BDH with $\alpha_{\text{desired}} = 0.6$ on the noisy 8-class data set. The weight vectors of the PEs are plotted. The PE belonging to a particular class has the same color as that used to represent the class in Figure 4.4 (Left). **Left:** Result of applying KSOM, $\alpha_{\text{achieved}} = 0.9167$. **Right:** Result of applying BDH with $\alpha_{\text{desired}} = 0.6$, $\alpha_{\text{achieved}} = 0.915$.

4.8 Number of weights belonging to each class in the SOMs resulting from KSOM and BDH with $\alpha_{\text{desired}} = 0.6$ on the noisy 8-class data set. $\alpha_{\text{achieved}}(\text{BDH with } \alpha_{\text{desired}} = 0.6) = 0.915$ and $\alpha_{\text{achieved}}(\text{KSOM}) = 0.9167$. So we observe that the value of $\alpha_{\text{achieved}} \approx 1$ for KSOM, as well as for BDH with $\alpha_{\text{desired}} = 0.6$. 
4.9 Result of applying BDH with $\alpha_{\text{desired}} = -0.5$. **Left:** SOM, the weight vectors of the PEs are plotted. The PE belonging to a particular class has the same color as that used to represent the class in Figure 4.4 (Left). The effect of negative magnification is observed by the fact that here the small classes in the data set occupy much larger regions of the SOM as compared to the number of weights belonging to them in the case of Conscience, KSOM or BDH with $\alpha_{\text{achieved}} \approx 1$ (the discrepancy for class H is due to the fact that the SOM has a finite size and this class did not get enough space to spread out).

4.10 $\alpha_{\text{achieved}}$ (obtained by the two variants of BDH) vs $\alpha_{\text{desired}}$ for the noisy 8-class data set. We can see that (i) BDH results in maps with $\alpha_{\text{achieved}} \approx 1$ when applied with $\alpha_{\text{desired}}$ anywhere between $\sim 0.6$ and $\sim 1.1$. (ii) When applied with $\alpha_{\text{desired}} < 0$, negative magnification is achieved. The smaller classes in the input data are magnified in the SOM. (iii) There is no definite trend in the performance of BDH when used with values of $\alpha_{\text{desired}}$ outside of the regions discussed in (i) and (ii). These are significant findings as maps with $\alpha_{\text{achieved}} \approx 1$ and $\alpha_{\text{desired}} < 0$ have important practical applications.
4.11 SOMs resulting from the application of Conscience and BDH with $\alpha_{desired} = 1$ on the noisy 11-class data set. The weight vectors of the PEs are plotted. The PE belonging to a particular class has the same color as that used to represent the class in Figure 4.4 (Right). **Left:** Result of applying BDH with $\alpha_{desired} = 1$, $\alpha_{achieved} = 1.005$. **Right:** Result of applying Conscience, $\alpha_{achieved} = 0.945$. Performance of BDH with $\alpha_{desired} = 1$ is somewhat better than that of Conscience.

4.12 Number of weights belonging to each class in the SOMs resulting from Conscience and BDH with $\alpha_{desired} = 1$ on the noisy 11-class data set. $Error_{BDH} = 8.7605$ and $Error_{Conscience} = 14.4914$. The values of $\alpha_{achieved}$ were 1.005 and 0.945 for BDH and Conscience respectively. So the performance of BDH with $\alpha_{desired} = 1$ is slightly better than that of Conscience. One thing to note is that even in the ideal case, the rare classes have very little or no representation in the SOM. For both algorithms, the number of weights belonging to each class are close to but not exactly equal to ideal. This discrepancy arises because the theoretical results are asymptotic and we only have a finite number of PEs. Integer arithmetic also leads to deviation from the ideal numbers.
4.13 SOMs resulting from the application of KSOM and BDH with $\alpha_{desired} = 0.6$ on the noisy 11-class data set. The weight vectors of the PEs are plotted. The PE belonging to a particular class has the same color as that used to represent the class in Figure 4.4 (Right). **Left:** Result of applying KSOM, $\alpha_{achieved} = 0.9917$. **Right:** Result of applying BDH with $\alpha_{desired} = 0.6$, $\alpha_{achieved} = 0.915$. The three rare classes do find some representation in the SOM here. However, the number of weight vectors belonging to these classes are still very small and so the rare classes might go undetected.

4.14 Number of weights belonging to each class in the SOMs resulting from KSOM and BDH with $\alpha_{desired} = 0.6$ on the noisy 11-class data set. $\alpha_{achieved}(BDH \text{ with } \alpha_{desired} = 0.6) = 0.915$ and $\alpha_{achieved}(KSOM) = 0.9917$. These values are consistent with the observation for the 8-class data set that the value of $\alpha_{achieved} \approx 1$ for KSOM as well as for BDH with $\alpha_{desired} = 0.6$. The three rare classes do find some representation in the SOM here. However, the number of weight vectors belonging to these classes are still very small and so the rare classes might go undetected.
4.15 Result of applying BDH with $\alpha_{\text{desired}} = -0.5$. **Left:** SOM, the weight vectors of the PEs are plotted. The PE belonging to a particular class has the same color as that used to represent the class in Figure 4.4 (Right). The rare classes R, Y and T occupy much larger regions of the SOM as compared to the number of weights belonging to them in the case of Conscience, KSOM or BDH with $\alpha_{\text{achieved}} \approx 1$. The classes which were rare in the input set have been enhanced in the SOM, thereby improving their detectability.

4.16 $\alpha_{\text{achieved}}$ (obtained by both the variants of BDH) vs $\alpha_{\text{desired}}$ for the noisy 11-class data set. We can see that (i) BDH results in maps with $\alpha_{\text{achieved}} \approx 1$ when applied with $\alpha_{\text{desired}}$ anywhere between $\sim 0.6$ and $\sim 1.1$. (ii) When applied with $\alpha_{\text{desired}} < 0$, even though the values of $\alpha_{\text{achieved}}$ are not negative, yet the number of weights belonging to each of the rare classes are many more as compared to the corresponding numbers obtained with other algorithms (numbers listed in Table 4.3). The rare classes in the input data are enhanced in the SOM. So, negative magnification is achieved. (iii) There is no definitive trend in the performance of BDH when used with values of $\alpha_{\text{desired}}$ outside of the regions discussed in (i) and (ii). These are important findings as maps with $\alpha_{\text{achieved}} \approx 1$ and $< 0$ have important practical applications. These observations are exactly the same as those obtained from the experiments on the noisy 8-class data set.

5.1 **Concept of a spectral image.** Figure from [14].
5.2 Comparison of supervised classification and BDH clustering with $\alpha_{\text{desired}} < 0$ \textbf{Left:}

An earlier supervised classification. A gray spot (the color of the background, 'bg') outlined by a small black rectangle within the framed top right quadrant is an unclassified region. \textbf{Right:} Result of SOM clustering using BDH magnification control with $\alpha_{\text{desired}} = -0.8$ on the top right quadrant of the image. The agreement between the supervised class map and this cluster map is striking, which inspires confidence in the clustering. Secondly, the spot that remained unclassified in the supervised map is now filled exactly and with a color (greenish-yellow - class a) that is different from all previous class colors and its spectral signature is distinct (Figure 5.3). We discovered a new class. Moreover, this cluster only occurs at this location, and nowhere else: we discovered a small rare class! Other small classes V and C are marked by ovals and dashed rectangles in the two images.  

5.3 Mean spectral signatures of the classes shown in Figure 5.2. Those of the small classes C, V and a have been plotted in red. a, the newly discovered class has a spectral signature that is distinct from all the other clusters.  

5.4 Comparisons of the SOMs developed by BDH vs Conscience learning. \textbf{Left:} The SOM obtained by the application of BDH (with $\alpha_{\text{desired}} < 0$) on $256 \times 256$ subimage of the entire Ocean City image. \textbf{Right:} SOM obtained by Conscience learning using the entire Ocean City image. As discussed in the text, the rare clusters are magnified in the BDH SOM in comparison to the Conscience SOM.
6.1 A simple example dendrogram.

6.2 Partitioning Illustration. The parent cluster $C$ contains two subclusters $A$ and $B$ having a total of 8 leaves ($n_c = 8$) and four "loose" leaves ($n_l = 4$). If the condition in Equation 6.4 is satisfied, $C$ is a cluster. Otherwise, $C$ will not be kept as a cluster and $A$ and $B$ will be examined to see if they can be kept as clusters.

6.3 **Left:** The (100) weight vectors resulting from applying KSOM on Data Set I plotted in the 2-dimensional input space. Only those weights which had non-zero number of inputs mapped to them were used for clustering. **Right:** Weight vectors and number of inputs mapped to each PE (the maximum here is 26) in the SOM obtained by applying KSOM to Data Set I.

6.4 The proposed clustering technique separated the weights corresponding to Data Set I into 4 different clusters that are the same as the expected clusters.

6.5 **Left:** The 100 weight vectors resulting from applying KSOM on Data Set II plotted in the 2-dimensional input space. Only those weights which had non-zero number of inputs mapped to them were used for clustering. **Right:** Weight vectors and number of inputs mapped to each PE (the maximum here is 31) in the SOM obtained by applying KSOM to Data Set II.

6.6 The proposed clustering technique separated the weights corresponding to Data Set II into 5 different clusters which is what we expect by looking at the weights in the input space. A singleton cluster has been detected.
6.7 **Left:** Weights obtained by applying BDH with $\alpha_{desired} = 1$ on the 6-dimensional, noisy 8-class data set. A $20 \times 20$ SOM was used. **Right:** Height of the bar represents the number of inputs mapped to the corresponding PE in the $20 \times 20$ SOM.

6.8 **Clustering of the weights of the SOM (corresponding to BDH with $\alpha_{desired} = 1$ on the 6-dimensional, noisy 8-class data set) by the proposed technique. Different clusters are represented by different colors. Weight vectors are plotted and PEs with no inputs mapped are marked as in Figure 6.7. **Left:** Using $K = 1$ resulted in 15 clusters. **Right:** Using $K = 4$ resulted in 8 clusters.

6.9 **Left:** Weights obtained by applying BDH with $\alpha_{desired} = -0.8$ on the 6-dimensional, noisy 11-class data set. A $20 \times 20$ SOM was used. **Right:** Height of the bar represents the number of inputs mapped to the corresponding PE in the $20 \times 20$ SOM.

6.10 **Clustering of the weights of the SOM (corresponding to BDH with $\alpha_{desired} = -0.8$ on the 6-dimensional, noisy 11-class data set) by the proposed technique. Different clusters are represented by different colors. Weight vectors are plotted and PEs with no inputs mapped are marked as in Figure 6.9. **Left:** Using $K = 1$, resulted in 17 clusters. **Right:** Using $K = 4$ resulted in 11 clusters.
List of Tables

2.1 Comparing BDH with $\alpha_{desired} = 1$ with Conscience with zero neighborhood .................................................. 34
2.2 Comparing BDH with $\alpha_{desired} = 1$ with Conscience with 1 neighborhood 37
4.1 Comparing BDH with $\alpha_{desired} = 1$ and Conscience ................................. 66
4.2 Comparing BDH with $\alpha_{desired} = 1$ with Conscience ................................. 73
4.3 Number of weights belonging to the rare classes in the SOMs obtained by various algorithms .................................................. 77
6.1 Within-cluster distances $S(Q_k)$ and between-cluster distances $d(Q_k, Q_l)$; $x_i, x_{i'} \in Q_k, i \neq i', x_j \in Q_l, k \neq l$. $N_k$ is the number of samples in cluster $Q_k$ and $c_k = 1/N_k \sum_{x_i \in Q_k} x_i$ .................................................. 93
A.1 Example of asymptotically optimal bin widths and errors for $f = \mathcal{N}(0, I_d)$ .................................................. 125
A.2 Equivalent sample sizes across dimensions for the multivariate normal density, based on Epanechnikov’s criterion .................................................. 126
Chapter 1
Background

Advanced modifications of the neural network algorithm called the Kohonen’s Self-Organizing Map (SOM) [15] [16] show great promise in effective, precise and detailed knowledge discovery (clustering) for complex, high-dimensional data as opposed to many conventional techniques that may fail to handle such data. In this chapter, an introduction to the Self-Organizing Map algorithm and its properties is provided, and the advantages of using SOMs for clustering complex, high-dimensional data as compared to some other existing techniques are discussed. A explanation of Magnification control, the particular aspect of SOMs that has been investigated during the course of this thesis follows next. The organization of the thesis is discussed at the end.

1.1 The Self-Organizing Map

A Self-Organizing Map is a (usually) two-dimensional grid, where each node in the grid is a Processing Element or PE (also called neuron). PEs are indexed by 1, 2,..., N. If the input data is d-dimensional, a d-dimensional vector, called the weight vector, is associated with each PE (Figure 1.1). During learning, the weights adapt and each becomes a prototype of a set of similar inputs by the algorithm described in Equations 1.1 and 1.2. Let \( \mathbf{v} = (v_1, v_2, \ldots, v_d) \in M \subseteq \mathbb{R}^d \) be the input pattern
Self-Organizing Map - PE Space

Every node $i$ in the map is associated with a $d$-dimensional weight vector $w_i$.

Figure 1.1  SOM - an array of PEs. Each input pattern is connected to all the Processing Elements (PEs). The weights associated with each PE have the same dimensionality as that of the input vectors.

present in the manifold $M$ and $w_j = (w_{j1}, w_{j2}, \ldots, w_{jd})$ $j = 1, \ldots, N$ be synaptic weight vectors associated with PE $j$. This is an iterative algorithm. The following three steps are carried out at every iteration.

1. Competition
   A random sample $v \in M$ is drawn and a winning neuron $i(v)$ is selected.
   \[ i(v) = \arg \min_j ||v - w_j||, \quad j = 1, \ldots, N \quad (1.1) \]

2. Cooperation
   The winning neuron activates neurons in its neighborhood, according to a neighborhood function $h_{j, i(v)}(t)$, where $t$ is the iteration number.

3. Synaptic adaptation
   \[ w_j(t + 1) = w_j(t) + a(t)h_{j, i(v)}(t)(v - w_j(t)) \quad (1.2) \]
where $a(t)$ is the learning parameter that is slowly decreased as iterations proceed.

The SOM is a vector quantization process that distributes the weight vectors such that the input distribution is represented in an optimal way. Each weight vector is the prototype of a set of input vectors. The vectors in the input space that are closest to a given weight constitute its receptive field. For finite data sets a modified reconstruction error is minimized. At the same time, the weights of the SOM are organized into a two-dimensional order such that the PEs with similar weights are closer to each other in the grid than those with more dissimilar weights. This is the topology preserving property of SOMs. In this sense, the SOM is a similarity graph, and a clustering diagram. Clustering with an SOM is a two-level process: first, a large set of prototypes - much larger than the expected number of clusters is found using the SOM algorithm (by using the iterative process discussed in equations 1.8 and 1.5). On the next level, these prototypes are clustered (Figure 1.2). Each input vector then belongs to the same cluster as its prototype (the weight it is closest to in terms of Euclidean distance measure). The primary benefit of the two-level approach is the reduction of the computational cost. This and other advantages of using SOMs for clustering will be discussed next. The original “basic” SOM described by Equations 1.1 and 1.2 was invented by Teuvo Kohonen in the late 1970s [15]. We will refer to it by the acronym KSOM.
Figure 1.2  Clustering using SOMs is a two-step process. First prototypes (representatives) of the data are formed in such a way that they are ordered in a topology preserving way on a two-dimensional grid. In the second step, these prototypes are clustered. Each input vector then belongs to the same cluster as its prototype. (Each square in the grid represents a PE which has a weight vector associated with it. The fence between two PEs represents the Euclidean distance between the weights associated with those PEs)

1.2 Why SOM?

Clustering refers to grouping together similar data vectors of a set of data vectors. Clustering of high-dimensional, complex, unknown data is a challenging task. Existing clustering techniques are often not useful for several reasons. Many times, the number of classes in the data set is unknown. Also, often when a data set is analyzed, there is very little prior knowledge available about it. An example is the case when hyperspectral images (these are remotely sensed images with a high-dimensional
spectral signature associated with each pixel) of other planets are analyzed. In such a case it is possible that we come across some data type that has been unknown so far. High dimensionality also makes the data very difficult to handle. Typically, it is not easy to reduce the dimensionality because the data may not lie on a linear manifold. So, dimensionality reduction using usual linear techniques such as PCA often result in poor representation of the input data. There maybe difficulties even if we have labelled samples and are doing supervised classification because enough training data might not be available for each class. For the Bayesian classifier for example, the mean and the covariance matrix of each class needs to be determined. If the data is $d$-dimensional, at least $d + 1$ samples of each class are needed such that the matrix is non-singular. $10 \times d$ samples are required for a reasonable estimate of the covariance matrix. In the case of hyperspectral images, the dimensionality of the spectral signatures associated with each image pixel can be as high as 500! In such a case, enough (labelled) training samples are usually not available.

In case of prototype based partitional clustering, data points are partitioned into a specified number of clusters usually optimizing some criterion. In K-means for example, it is the within cluster sum of squares. This kind of clustering has the disadvantages that it is sensitive to initialization and susceptible to local minima and it is also sensitive to noise and outliers. SOMs have a benefit over these methods in terms of noise reduction: the prototypes in this case are local averages of the data
and therefore less sensitive to random variations than the original data. Moreover, it is difficult to determine the number of clusters, which is needed for using prototype based partitional clustering. SOM clustering does not require prior determination of the number of clusters.

In general, clustering is also a computationally intensive task. As an example, the number of ways of sorting $N$ vectors into $k$ groups is a Stirling number of the second kind which is approximately $k^N/k!$. So searching the best partition of 100 vectors in 5 clusters requires consideration of more than $10^{67}$ partitions. The problem is compounded when $k$ is unknown: the number of possibilities becomes a sum of Stirling numbers [17]. Using a two-level approach as in SOMs reduces the computational cost as the clustering will be done on a much smaller set of vectors.

Using SOMs for clustering is advantageous as it enables visualization of high-dimensional clusters in two or three dimensions and unlike some other techniques it doesn't require prior assumption about the number of classes in the data, splitting, merging criteria etc.

We summarize the primary advantages of using SOMs for clustering in the following list:

**Reduction in Computational Complexity** : Since the clustering is now carried out on a relatively small number of prototypes, the computational complexity is reduced greatly.
Noise Reduction: The prototypes are local averages of the data and, therefore, less sensitive to random variations than the original data. Outliers are less of a problem.

Clusters and Similarity Patterns can be visualized: As the prototypes are ordered topologically (in a neighborhood preserving way) in the SOM, the SOM is a similarity map and a clustering diagram. By representing the differences between weights of neighboring PEs of the grid in some way we can visualize the clusters.

No need of making any prior assumptions: Unlike some other clustering algorithms, this technique does not require any prior knowledge about the data set to be clustered (other algorithms might require the number of classes in the data for example).

Information Compression: Information compression occurs while preserving topological and metric relationship of primary data items.

Some applications for which Self-Organizing maps have been used successfully are listed below:

- Clustering of remotely sensed hyperspectral images of a region for obtaining spatial maps of the distribution of different materials present in that place [18].
Also for discovering new materials for example, discovery of new classes of soil on Mars [19].

- Interpreting patterns of gene expression [20].

- Analysis of electrical signals from the brain to understand how the brain functions or how learning takes place etc [21].

- Development of visual document maps for organizing, visualizing, searching, categorizing and filtering of textual and image data. This has enabled content-addressable search. [22, 23].

- Portrayal of complex correlations in statistical data [24].

1.3 Magnification control

One theoretically interesting and powerful data analysis aspect of SOMs is the so-called *map magnification*. The following power law relates the density of weights in the input space $Q(w)$ to the pdf $P(w)$ of the input samples,

$$Q(w) = cP(w)^\alpha$$  \hspace{1cm} (1.3)

where $\alpha$ is the *magnification exponent* and $c$ is a constant [13].

Some values of $\alpha$ are associated with particular quantization or information theoretical properties [13]. A map with $\alpha = 1$ maximizes information theoretic entropy. $\alpha = 1/3$ for 1-dimensional data corresponds to the minimum mean squared error
quantization case. $\alpha < 0$ enables better categorization by enlarging response areas for low-frequency inputs. This is potentially useful for making discoveries as it would enhance the detectability of rare classes. Kohonen’s SOM algorithm (KSOM) [15] achieves $\alpha = 2/3$ (under certain conditions) [25]. This value of $\alpha$ is optimal in neither minimum distortion nor maximum entropy sense. A SOM variant called Conscience algorithm [26] can induce $\alpha = 1$, but not any other value. For 1-dimensional data, and in the special case of a 1-dimensional SOM with a fixed rectangular neighborhood of $n$ PEs around the winner, $\alpha$ can be described as a function of $n$ [27]. For 1-dimensional data, there exists another result that describes $\alpha$ as a function of the normalized second moment of the neighborhood function [28]. However, neither of these results is extendible to higher dimensions. An algorithm that enables explicit control of the magnification was developed by Bauer, Der and Hermann [13] by using adaptively adjusted local learning rates, where the adaptability is node dependent. We will refer to this algorithm by the acronym BDH. It modifies the learning rate in KSOM by forcing the local adaptabilities to depend on the input density $P(w_j)$ at the position of the receptive field center $w_j$ associated with PE $j$:

$$\epsilon_j = \epsilon_0 P(w_j)^m$$

(1.4) where $m$ is the free parameter that allows controlling $\alpha$. Since $P(w_j)$ is unknown, the above value is obtained on the basis of the information already acquired by the
SOM and exploiting the following relation:

\[ P(w_j) \propto Q(w_j)P(j) \]  \hspace{1cm} (1.5)

where, \( Q(w_j) \) is the weight density in \( M \) and \( P(j) \) is the probability of the neuron \( j \) being the best matching unit. The RHS of Equation (1.5) can be approximated as

\[ Q(w_j) \propto \frac{1}{V_j} \]  \hspace{1cm} (1.6)

where \( V_j \) is the volume of the Voronoi polygon corresponding to \( w_j \). This volume in turn is proportional to the \( d^{th} \) power of the mean distance between the input signal \( v \) and the winning weight vector \( w_j \), where \( d \) is the dimensionality of the input space.

Also,

\[ P(j) \propto \frac{1}{\Delta t_j} \]  \hspace{1cm} (1.7)

where, \( \Delta t_j \) is proportional to the difference between the present iteration index \( t \) and the last time neuron \( j \) won the competition. This results in

\[ \epsilon_j(t) = \epsilon_0(t) \left[ \frac{1}{\Delta t_j} \left( \frac{1}{|v - w_j|^d} \right) \right]^m \]  \hspace{1cm} (1.8)

as the modified learning rate. This learning rate is the same for all neurons at a particular iteration. The relation between \( m \) and the desired \( \alpha \) is given by

\[ \alpha = \frac{2}{3}(1 + m) \]  \hspace{1cm} (1.9)

Therefore, depending upon the value of the desired \( \alpha (\alpha_{desired}) \), \( m = \frac{3}{2} \alpha_{desired} - 1 \) has to be used in the BDH algorithm.
1.4 Focus of the present work

Theoretical validity of BDH is severely limited. The available theory guarantees that BDH will successfully induce the desired value of $\alpha$ only for:

- 1-D input data
- 2-D data, $\mathbf{v} = (v_1, v_2)$, if and only if

$$p_{\mathbf{v}}(\mathbf{v}) = p_{v_1}(v_1)p_{v_2}(v_2) \text{ (i.e., the data are independent in the two dimensions.)}$$

We will refer to the data sets with the above properties as "allowed" data and the rest of them as "forbidden" data. Theoretical analysis of BDH has been possible only for "allowed" data. In this thesis, we investigated the BDH algorithm for both "allowed" and "forbidden" data and determined its performance and utility.

The limitations in the theoretical analysis of BDH come from the theoretical difficulties of the original Kohonen’s SOM algorithm. In [29], it was proven that the KSOM algorithm cannot be derived as a stochastic gradient descent algorithm on any energy (potential) function. This makes it difficult to theoretically analyze KSOM and other SOM algorithms based on it. Other theoretically more treatable, learning dynamics includes the modified SOM-algorithm [30] which was derived as gradient descent on an energy function, but up until now it has not been possible to derive the magnification law for it. [31] allows theoretical treatment of the learning algorithm but at the cost of loosing the convenient fixed lattice structure.
1.5 Our approach

Because of the difficulty in theoretically analyzing BDH, it has not been investigated before for “forbidden” data. Ours is the first approach in this context [32] [33]. In order to utilize the desirable properties of magnification control for data analysis and clustering, we investigate the performance of BDH for “forbidden” data. In the absence of tools for theoretical analysis, we carried out extensive simulations on both “allowed” and “forbidden” data. Our approach has been to use a certain value of $\alpha$ ($\alpha_{desired}$) in the BDH algorithm, evaluate the value of $\alpha$ actually achieved ($\alpha_{achieved}$) (whenever it is possible) and compare the two to gauge the performance of BDH. In cases when it was not possible to evaluate the value of $\alpha_{achieved}$, the performance was assessed indirectly: by observing the developed map. By carrying our careful, controlled experiments, we have tried to chart BDH performance for “forbidden” data and hope that this not only helps in better clustering but also provides some feedback to theory of BDH performance for all kinds of data.

1.6 Cluster identification in the SOM

After the SOM has converged, clusters in the SOM need to be identified. Many approaches including different visualization methods, semi-automatic techniques etc. are used in grouping the weights of the SOM into clusters. However, there does not exist any standard omni-potent scheme that does this perfectly for all cases. We developed a technique to identify the clusters in the SOM in a fully automated way.
In Chapter 6, we describe this technique and present results of applying it on some test data sets.

In the next chapters we will discuss various simulations and the conclusions we have been able to draw about the performance and utility of the BDH algorithm. Experiments with 1-, 2- and higher dimensional synthetic data and conclusions about the performance of the BDH algorithm on these types of data sets are discussed in chapters 2, 3 and 4. In Chapter 5, the BDH algorithm is applied to a real multispectral image. A fully automated technique for identification of clusters in the SOM is described in Chapter 6. Finally, we summarize our findings in Chapter 7.
Chapter 2
Magnification control for 1-dimensional data

In this chapter, we discuss the experiments on 1-dimensional data which we carried out as a first step in understanding magnification control.

We used different values of $\alpha_{desired}$ in the BDH algorithm, evaluated the corresponding value of $\alpha_{achieved}$ of the map and compared the two to assess the performance of the algorithm for 1-dimensional data. We also discuss some experiments we carried out to compare the BDH algorithm with $\alpha_{desired} = 1$ and the Conscience algorithm [26], a heuristic algorithm that results in maps with $\alpha_{achieved} = 1$. This chapter also includes investigations on two slightly different versions of the BDH algorithm for 1-dimensional data.

2.1 Description of 1-dimensional data sets

Simulations were carried out on two 1-dimensional data sets of known distributions.

2.1.1 Data Set I

This data set consisted of 10000 samples of 1-dimensional data with pdf $p(v) = 2v, v \in [0, 1]$ (Figure 2.1 (Left)).
2.1.2 Data Set II

This data set consisted of 10000 samples of 1-dimensional data with pdf \( p(v) = 3v^2, v \in [0,1] \) (Figure 2.1 (Right)).

![Figure 2.1 Left: Distribution of Data Set I. Right: Distribution of Data Set II.](image)

2.2 Procedure for evaluating \( \alpha_{achieved} \)

In this section, we describe the process of evaluation of \( \alpha_{achieved} \) by the map for 1-dimensional SOM and 1-dimensional data given the pdf of the data. The following power law relates the density of weights in the input space \( Q(v) \) to the pdf \( p(v) \) of the input samples,

\[
Q(v) = p(v)^\alpha \times \text{constant} \tag{2.1}
\]

\( Q(v) \) being the density of weights in the input space, is the number of reference vectors in a small volume \( dv \) of the input space. Let us first analyze the case of distribution \( p(v) = 2v, v \in [0,1] \) and find the constant in Equation 2.1. Integrating both sides of
Equation 2.1,
\[ \int_{\text{entire input space}} Q(v)dv = \text{constant} \int_0^1 p(v)^\alpha dv \] (2.2)

When \( p(v) \) is varied from 0 to 1, equivalently, \( Q(v) \) is integrated over the entire input space which results in the total number of weight vectors, i.e. the total number of PEs \( N \) in the SOM:
\[ \int_{\text{entire input space}} Q(v)dv = N \] (2.3)

The right hand side of Equation (2.2) for the distribution \( p(v) = 2v \) is
\[ \text{constant} \int_0^1 p(v)^\alpha dv = \text{constant} \int_0^1 (2v)^\alpha dv = \frac{\text{constant}}{\alpha + 1} \frac{2^\alpha}{\alpha + 1} - v^{\alpha+1}\bigg|_0^1 \] (2.4)

Using Equations 2.3 and 2.4 we get the value of the constant as
\[ \text{constant} = \frac{N(\alpha + 1)}{2^\alpha} \] (2.5)

So the relation between \( Q(v) \) and \( P(v) \) is
\[ Q(v) = N\left(\frac{\alpha + 1}{2^\alpha}\right)P(v)^\alpha \] (2.6)

Now, let us divide the input range into bins of length \( \Delta = 0.1 \). Then, to cover the range from 0 to 1, we shall have 10 such intervals. The \( i^{th} \) interval will range from \( i\Delta \) to \( (i+1)\Delta \). Using Equation 2.6, we can calculate the theoretical number of weight vectors, \( n_i^{\text{theoretical}} \) in the \( i^{th} \) bin. Integrating the left hand side of Equation 2.6
over the $i^{th}$ bin gives the value is $n_i^{\text{theoretical}}$. By integrating the right hand side of Equation 2.6, over the $i^{th}$ bin, we get

$$\frac{N(\alpha + 1)}{2^\alpha} \frac{2^\alpha}{\alpha + 1} v^{\alpha+1} \left| \frac{(i+1)^{\Delta}}{i^{\Delta}} \right| (2.7)$$

and so

$$n_i^{\text{theoretical}} = N [((i + 1)^{\Delta})^{(\alpha+1)} - (i^{\Delta})^{(\alpha+1)}] (2.8)$$

So, now we have the theoretical value of the number of reference vectors in the $i^{th}$ bin if the magnification exponent is $\alpha$. After the BDH algorithm is applied and the map is formed, we can find out the actual number of weight vectors lying in the different bins. Say $n_i$ is the actual number of weight vectors lying in the $i^{th}$ bin. Then, the $\alpha$ achieved by the map ($\alpha_{\text{achieved}}$) is the value that minimizes the sum of squared differences $\Sigma_{i=0}^9 (n_i - n_i^{\text{theoretical}})^2$ between the theoretical and measured average values of the number of weights in each bin. For the distribution, $p(v) = 3v^2$, we proceed in a similar manner and obtain the following value for $n_i^{\text{theoretical}}$

$$n_i^{\text{theoretical}} = N [((i + 1)^{\Delta})^{(2\alpha+1)} - (i^{\Delta})^{(2\alpha+1)}] (2.9)$$

The value of $\alpha_{\text{achieved}}$ by the map is found out in the same way as for the first distribution. In [13], the BDH results were discussed for some values of $\alpha_{\text{desired}}$ between $-2/3$ and $4/3$ for Data Set I. We carried out simulations with both distributions, $p(v) = 2v$ and $p(v) = 3v^2$. 
2.3 Simulations on 1-dimensional data sets

In this section, we demonstrate the performance of the BDH algorithm on the two 1-dimensional data sets. In order to assess the performance, we use different values of $\alpha_{desired}$ in the algorithm, evaluate the corresponding value of $\alpha_{achieved}$ by the map and compared the two.

2.3.1 Magnification control on Data Set I (pdf: $p(v) = 2v$)

For Data Set I, we carried out experiments using $-0.5 \leq \alpha_{desired} \leq 4/3$ in the BDH algorithm. The number of PEs, $N$, in the 1-dimensional SOM is 100. We depict some of these simulations next.

Case I: Inducing $\alpha_{desired} = 1.0$

Figures 2.2 and 2.3 depict the results of applying BDH with $\alpha_{desired} = 1.0$ on Data Set I. From Figure 2.2, it can be seen that the final weights are ordered, hence,

![Figure 2.2](image)

**Figure 2.2** Initial and final weights when BDH was applied on Data Set I with $\alpha_{desired} = 1.0$. **Left:** Initial weights which were chosen randomly. **Right:** Final weights are ordered, indicating that the mapping is topology preserving.
there is global organization of the weights which shows that the mapping is topology preserving. The value of \( \alpha_{achieved} \) is 0.982482, which is quite close to \( \alpha_{desired} = 1 \). The weight density closely follows the input pdf as \( \alpha_{achieved} \approx 1 \).

![Diagram](image)

**Figure 2.3** Result of applying BDH on Data Set I with \( \alpha_{desired} = 1 \). Number of weights in each bin: theoretical \( n_i^{theoretical} \) for \( \alpha = 1 \), those obtained by BDH \( n_i \) with \( \alpha_{desired} = 1 \) and what ideally would be obtained by applying KSOM. In this case, \( n_i^{theoretical} \) and \( n_i \) are quite close. The value of \( \alpha \) that gives the minimum value of the sum \( \Sigma_{i=0}^{9}(n_i - n_i^{theoretical})^2 \), is \( \alpha_{achieved} \). Here, \( \alpha_{achieved} = 0.982482 \).

**Case II: Inducing** \( \alpha_{desired} = 0.3 \)

Figures 2.4 and 2.5 depict the results of applying BDH with \( \alpha_{desired} = 0.3 \) on Data Set I. From Figure 2.4, it can be seen that the final weights are ordered, hence, there is global organization of the weights, so once again, we see that the mapping is topology preserving. The value of \( \alpha_{achieved} \) is 0.322523, which is quite close to \( \alpha_{desired} = 0.3 \).
Figure 2.4  Initial and final weights when BDH was applied on Data Set I with $\alpha_{desired} = 0.3$.  
**Left:** Initial weights which were chosen randomly.  **Right:** Final weights are ordered, therefore the mapping is topology preserving.

Figure 2.5  Result of applying BDH on Data Set I with $\alpha_{desired} = 0.3$.  Number of weights in each bin: theoretical ($n_i^{\text{theoretical}}$) for $\alpha = 0.3$, those obtained by BDH ($n_i$) with $\alpha_{desired} = 0.3$ and what would ideally be obtained by applying KSOM.  In this case, $n_i^{\text{theoretical}}$ and $n_i$ are quite close.  The value of $\alpha$ that gives the minimum value of the sum $\sum_{i=0}^{9}(n_i - n_i^{\text{theoretical}})^2$, is $\alpha_{achieved} = 0.322523$.

**Case III: Inducing $\alpha_{desired} = -0.5$**

This is an interesting case when a negative value of the $\alpha_{desired}$ is used in the BDH algorithm.  A negative magnification exponent means that the density of the weights
is in inverse proportion to the input pdf. Wherever the pdf of the input samples is higher, lesser number of weights will occur in that region and vice-versa. Figures 2.6 and 2.7 depict the results of applying BDH with $\alpha_{desired} = -0.5$ on Data Set I. In this case, when BDH is used with $\alpha_{desired} = -0.5$, the value of $\alpha_{achieved} = -0.122122$ which is not very close to $\alpha_{desired}$ but is a negative value. Figure 2.7 (Left) shows how the theoretical and experimental results differ in terms of number of weights in each bin. Figure 2.7 (Right) shows that we achieve a map with an inverse relation between the weight density and the input pdf, which is the aim of negative magnification. Also, once again it can be seen from Figure 2.6, that the final weights are ordered indicating that the mapping is topology preserving.

Figure 2.8 summarizes the results of applying BDH on Data Set I with different values of $\alpha_{desired}$.

So for Data Set I, the region of good performance of BDH is $0 \leq \alpha_{desired} \leq 1.0$.
Figure 2.7  Result of applying BDH on Data Set I with $\alpha_{desired} = -0.5$. **Left:** Number of weights in each bin: theoretical ($n_i^{theoretical}$) for $\alpha = -0.5$, those obtained by BDH ($n_i$) with $\alpha_{desired} = -0.5$ and what ideally would be obtained by applying KSOM. In this case, $n_i^{theoretical}$ and $n_i$ are quite close. The value of $\alpha$ that gives the minimum value of the sum $\sum_{i=0}^{9} (n_i - n_i^{theoretical})^2$, is $\alpha_{achieved}$. Here, $\alpha_{achieved} = -0.122122$. **Right:** Number of weights in each bin: theoretical for $\alpha = 1$ and those obtained by BDH with $\alpha_{desired} = -0.5$. The two plots exhibit an inverse relationship. Since the input pdf is directly proportional to the weight density when $\alpha_{desired} = 1$, the inverse relation between the plots in the figure shows that the input pdf and the weight density obtained on applying BDH with $\alpha_{desired} < 0$ are inversely proportional.

Figure 2.8  Performance of BDH algorithm on Data Set I: Plot comparing $\alpha_{achieved}$ and $\alpha_{desired}$. BDH performs best for $0 \leq \alpha_{desired} \leq 1.0$
(Figure 2.8). For $\alpha_{desired} > 1$ or $< 0$, BDH does not perform well.

2.3.2 Magnification control on Data Set II (pdf: $p(v) = 3v^2$)

For Data Set II we carried out experiments by using $-0.4 \leq \alpha_{desired} \leq 4/3$ in the BDH algorithm. The number of PEs, $N$, in the 1-dimensional SOM is 100. We show some of these cases next.

**Case I:** **Inducing** $\alpha_{desired} = 1.0$

Figures 2.9 and 2.10 depict the results of applying BDH with $\alpha_{desired} = 1$ on Data Set II. From Figure 2.9, it can be seen that the final weights are ordered, hence,

![Initial and final weights when BDH was applied on Data Set II with $\alpha_{desired} = 1.0$. Left: Initial Weights which were chosen randomly. Right: Final weights are ordered indicating that the mapping is topology preserving.](image)

there is global organization of the weights which shows that the mapping is topology preserving. The value of $\alpha_{achieved}$ is 1.022523, which is quite close to $\alpha_{desired} = 1$. The weight density closely follows the input pdf as $\alpha_{achieved} \approx 1$. 

```
Figure 2.10 Result of applying BDH on Data Set II with $\alpha_{desired} = 1$. Number of weights in each bin: theoretical ($n_i^{\text{theoretical}}$) for $\alpha = 1$, those obtained by BDH ($n_i$) with $\alpha_{desired} = 1$ and what would ideally be obtained by applying KSOM. In this case, $n_i^{\text{theoretical}}$ and $n_i$ are quite close. The value of $\alpha$ that gives the minimum value of the sum $\sum_{i=0}^{p}(n_i - n_i^{\text{theoretical}})^2$, is $\alpha_{achieved}$. Here, $\alpha_{achieved} = 1.022523$.

Case II: Inducing $\alpha_{desired} = 0.4$

Figures 2.11 and 2.12 depict the results of applying BDH with $\alpha_{desired} = 0.4$ on Data Set II. From Figure 2.11, it can be seen that the final weights are ordered, hence, there is global organization of the weights so once again, we see that the mapping is topology preserving. The value of $\alpha_{achieved}$ is 0.422523, which is quite close to $\alpha_{desired} = 0.4$.

Case III: Inducing $\alpha_{desired} = -0.3$

A negative magnification exponent refers to an inverse relationship between the pdfs of the data and the weights, i.e., wherever the pdf of the input samples is
Figure 2.11  Initial and final weights when BDH was applied on Data Set II with $\alpha_{\text{desired}} = 0.4$. **Left:** Initial Weights which were chosen randomly. **Right:** Final weights are ordered indicating that the mapping is topology preserving.

Figure 2.12  Result of applying BDH on Data Set II with $\alpha_{\text{desired}} = 0.4$. Number of weights in each bin: theoretical ($n_i^{\text{theoretical}}$) for $\alpha = 0.4$, those obtained by BDH ($n_i$) with $\alpha_{\text{desired}} = 0.4$ and what ideally would be obtained by applying KSOM. In this case, $n_i^{\text{theoretical}}$ and $n_i$ are quite close. The value of $\alpha$ that gives the minimum value of the sum $\sum_{i=0}^{9} (n_i - n_i^{\text{theoretical}})^2$, is $\alpha_{\text{achieved}}$. Here, $\alpha_{\text{achieved}} = 0.422523$.

higher, lesser number of weights will occur in that region and vice-versa. Figures 2.13 and 2.14 depict the results of applying BDH with $\alpha_{\text{desired}} = -0.3$ on Data Set II. In this case, when BDH is used with $\alpha_{\text{desired}} = -0.3$, the value of $\alpha_{\text{achieved}} = -0.015616$
Figure 2.13  Initial and final weights when BDH was applied on Data Set II with $\alpha_{\text{desired}} = -0.3$. **Left:** Initial Weights which were chosen randomly. **Right:** Final weights are ordered, therefore the mapping is topology preserving.

Figure 2.14  Result of applying BDH on Data Set II with $\alpha_{\text{desired}} = -0.3$. **Left:** Number of weights in each bin: theoretical ($n_i^{\text{theoretical}}$) for $\alpha = -0.3$, those obtained by BDH ($n_i$) with $\alpha_{\text{desired}} = -0.3$ and what ideally would be obtained by applying KSOM. In this case, $n_i^{\text{theoretical}}$ and $n_i$ are quite close. The value of $\alpha$ that gives the minimum value of the sum $\sum_{i=0}^{9}(n_i - n_i^{\text{theoretical}})^2$ is $\alpha_{\text{achieved}}$. Here, $\alpha_{\text{achieved}} = -0.015616$. **Right:** Number of weights in each bin: theoretical for $\alpha = 1$ and those obtained by BDH with $\alpha_{\text{desired}} = -0.3$. The two plots exhibit an inverse relationship. Since the input pdf is directly proportional to the weight density when $\alpha_{\text{desired}} = 1$, the inverse relation between the plots in the figure shows that the input pdf and the weight density obtained on applying BDH with $\alpha_{\text{desired}} < 0$ are inversely proportional.

which is not very close to $\alpha_{\text{desired}}$. Figure 2.7 (Left) shows how the theoretical and experimental results differ in terms of number of weights in each bin. However, 2.7
Figure 2.15  Performance of BDH algorithm on Data Set II: Plot comparing $\alpha_{\text{achieved}}$ and $\alpha_{\text{desired}}$. BDH performs best for $0.1 \leq \alpha_{\text{desired}} \leq 1.0$

(Right) shows that we achieve a map with an inverse relation between the weight density and the input pdf, which is the aim of negative magnification. Also, once again it can be seen from Figure 2.13, that the final weights are ordered indicating that the mapping is topology preserving.

Figure 2.15 summarizes the results of applying BDH on Data Set II with different values of $\alpha_{\text{desired}}$.

So for Data Set II, the region of best performance of BDH is $0.1 \leq \alpha_{\text{desired}} \leq 1.0$ (Figure 2.15). For $\alpha_{\text{desired}} > 1$ or $\leq 0$, BDH does not perform well.
2.4 Comparison of Conscience algorithm and BDH with $\alpha_{desired} = 1.0$

The Conscience algorithm [26] is a heuristic algorithm that results in maps with $\alpha_{achieved} = 1$. Such a map also exhibits equiprobabilistic mapping i.e., every PE in the SOM has an equal probability of winning. The BDH algorithm with $\alpha_{desired} = 1$ also promises to result in maps with $\alpha_{achieved} = 1$. Conscience is a heuristic algorithm, whereas, BDH is a more principled approach for magnification control. We compared the two algorithms to determine which is a better choice for obtaining an equiprobabilistic map.

2.4.1 Conscience algorithm

In an equiprobabilistic map i.e., every PE in the SOM has an equal probability of winning. Winner determination in this SOM algorithm uses the Conscience mechanism that adjusts the distances to encourage PEs that aren't winning with an average frequency and, to discourage PEs that are winning with an above average frequency. If $D_j$ is the Euclidean distance between the input and the weight of the $j^{th}$ PE for all PEs $j = 1, \ldots, N$, then the adjusted distance $D'_j$ is the distance minus a bias. The bias, denoted by $B_j$, is computed as follows

$$B_j = \gamma \left( \frac{1}{N} - F_j \right)$$  \hspace{1cm} (2.10)

where $F_j$ is the frequency with which the PE $j$ has historically won. At initialization, $F_j = \frac{1}{N}$ and so initially, $B_j = 0$. After $B_j$ and $D'_j$ are computed, the adjusted distance
$D_j$ is computed as

$$D'_j = D_j - B_j$$

(2.11)

The PE with the minimum adjusted distance is the winner. The frequency $F_j$ is adjusted as follows

For the winning PE, $F_{j \text{ new}} = F_{j \text{ old}} + \beta(1 - F_{j \text{ old}})$

For all other PEs, $F_{j \text{ new}} = F_{j \text{ old}} + \beta(0 - F_{j \text{ old}})$

Once the PE with the smallest adjusted distance is determined the winner PE's weight is adjusted to move closer to the input value. This is the case which we will refer to as Conscience with zero neighborhood. Another version of Conscience is when the weight of the winner PE as well the weights of two other PEs, one on each side of it are adjusted. We will refer to this as Conscience with a neighborhood of 1 around the winner. We carried out experiments to compare these to BDH with $\alpha_{\text{desired}} = 1$.

A map with $\alpha = 1$ is the most faithful representation of the input data. As mentioned earlier, such a mapping is equiprobabilistic, that is, all the PEs in such a map have equal probability of winning. This can be measured by the map entropy given by

$$I = \sum_{i=1}^{N} P(j) \log_2 P(j)$$

(2.12)

where $P(j)$ is the probability that the $j^{\text{th}}$ PE wins the competition and is given by

$$P(j) = \frac{I_j}{I_N}$$

(2.13)
where $I_j$ is the number of inputs samples mapped to the $j^{th}$ PE (an input sample $v$ is mapped to the $j^{th}$ PE if $j = \arg \min_k ||w_k - v||$) and $I_N$ is the total number of input samples. Ideally, in an equiprobabilistic map, all PEs should have an equal probability of winning, i.e., $P(j) = \frac{1}{N}$, and in that case $I$ would have a maximum value of $\log_2 N$. So, the higher the value of the map entropy, the more equiprobabilistic the mapping is. The number of input samples mapped to each PE in the SOM constitutes the “density information” available from the SOM.

2.4.2 Conscience with zero neighborhood

In this section, we compare the performance of BDH with $\alpha_{desired} = 1$ with that of Conscience with a zero neighborhood [26].

Experiments on Data Set I

From Figure 2.16, it can be seen that the final weights obtained by applying BDH with $\alpha_{desired} = 1$ are ordered (i.e., the mapping is topology preserving) unlike those obtained from Conscience.

Figures 2.17 and 2.18 depict that more number of iterations are required for convergence of the Conscience algorithm than for that of the BDH algorithm.

Experiments on Data Set II

From Figure 2.19, once again it can be seen that the final weights obtained by applying BDH with $\alpha_{desired} = 1$ are ordered (i.e., mapping is topology preserving)
Figure 2.16  Final weights obtained for Data Set I after 1000000 iterations. **Left:** Those obtained by Conscience algorithm (with zero neighborhood) are not ordered. **Right:** Those obtained by BDH algorithm (with $\alpha_{desired} = 1$) are ordered.

Figure 2.17  Comparing the results after 100000 iterations for Data Set I. **Left:** $\alpha_{achieved} = 0.7773$ by Conscience algorithm (with zero neighborhood). **Right:** $\alpha_{achieved} = 0.9825$ by BDH (with $\alpha_{desired} = 1$).

While those obtained by Conscience are not.

Figures 2.20 and 2.21 depict that more number of iterations are required for convergence of the Conscience algorithm than for BDH algorithm.

Table 2.1 summarizes the comparison between Conscience with zero neighborhood and BDH with $\alpha_{desired} = 1$. We can see that the entropy of the map obtained by BDH is higher than that of the map obtained by Conscience.
Figure 2.18  Comparing the results after 1000000 iterations for Data Set I. Left: $\alpha_{\text{achieved}} = 0.9785$ by Conscience algorithm (with zero neighborhood). Right: $\alpha_{\text{achieved}} = 0.9995$ by BDH (with $\alpha_{\text{desired}} = 1$).

Figure 2.19  Final weights obtained for Data Set II after 1000000 iterations. Left: Those obtained by Conscience algorithm (with zero neighborhood) are not ordered. Right: Those obtained by BDH algorithm (with $\alpha_{\text{desired}} = 1$) are ordered.

2.4.3 Conscience with a neighborhood of 1 around the winner

In this section, we compare the performance of BDH with $\alpha_{\text{desired}} = 1$ with that of Conscience with a neighborhood of 1 around the winner.
Figure 2.20  Comparing the results after 100000 iterations for Data Set II. Left: $\alpha_{\text{achieved}} = 0.7282$ by Conscience algorithm (with zero neighborhood). Right: Right: $\alpha_{\text{achieved}} = 1.0225$ by BDH (with $\alpha_{\text{desired}} = 1$).

Figure 2.21  Comparing the results after 1000000 iterations for Data Set II. Left: $\alpha_{\text{achieved}} = 0.9805$ by Conscience algorithm (with zero neighborhood). Right: $\alpha_{\text{achieved}} = 1.0275$ by BDH (with $\alpha_{\text{desired}} = 1$).

Experiments on Data Set I

From Figure 2.22, it can be seen that the final weights obtained by applying BDH with $\alpha_{\text{desired}} = 1$ are ordered (i.e., the mapping is topology preserving) unlike those obtained by Conscience.

Figures 2.23 and 2.24 depict that more number of iterations are required for the
Table 2.1  Comparing BDH with $\alpha_{desired} = 1$ with Conscience with zero neighborhood

<table>
<thead>
<tr>
<th>Iteration number</th>
<th>Data Set</th>
<th>Algorithm</th>
<th>$\alpha_{achieved}$</th>
<th>Entropy of the map</th>
</tr>
</thead>
<tbody>
<tr>
<td>100000</td>
<td>I</td>
<td>Conscience</td>
<td>0.7773</td>
<td>6.3650</td>
</tr>
<tr>
<td>100000</td>
<td>I</td>
<td>BDH (with $\alpha_{desired} = 1$)</td>
<td>0.9825</td>
<td>6.6254</td>
</tr>
<tr>
<td>1000000</td>
<td>I</td>
<td>Conscience</td>
<td>0.9785</td>
<td>6.3198</td>
</tr>
<tr>
<td>1000000</td>
<td>I</td>
<td>BDH (with $\alpha_{desired} = 1$)</td>
<td>0.9995</td>
<td>6.6250</td>
</tr>
<tr>
<td>1000000</td>
<td>II</td>
<td>Conscience</td>
<td>0.7282</td>
<td>6.3190</td>
</tr>
<tr>
<td>1000000</td>
<td>II</td>
<td>BDH (with $\alpha_{desired} = 1$)</td>
<td>1.0225</td>
<td>6.6224</td>
</tr>
<tr>
<td>1000000</td>
<td>II</td>
<td>Conscience</td>
<td>0.9805</td>
<td>6.1215</td>
</tr>
<tr>
<td>1000000</td>
<td>II</td>
<td>BDH (with $\alpha_{desired} = 1$)</td>
<td>1.0275</td>
<td>6.6225</td>
</tr>
</tbody>
</table>

Figure 2.22  Final weights obtained for Data Set I after 1000000 iterations. **Left:** Those obtained by Conscience algorithm (with a neighborhood of 1 around the winner) are only locally ordered. **Right:** Those obtained by BDH algorithm (with $\alpha_{desired} = 1$) are ordered globally.

convergence of the Conscience algorithm than those for BDH algorithm.
Figure 2.23  Comparing the results after 100000 iterations for Data Set I. **Left:** $\alpha_{achieved} = 0.9585$ by Conscience algorithm (with a neighborhood of 1 around the winner). **Right:** $\alpha_{achieved} = 0.9825$ by BDH (with $\alpha_{desired} = 1$).

Figure 2.24  Comparing the results after 100000 iterations for Data Set I. **Left:** $\alpha_{achieved} = 0.9404$ by Conscience algorithm (with a neighborhood of 1 around the winner). **Right:** $\alpha_{achieved} = 0.9995$ by BDH (with $\alpha_{desired} = 1$).

**Experiments on Data Set II**

From Figure 2.25, it can be seen that the final weights obtained by applying BDH with $\alpha_{desired} = 1$ are ordered globally (i.e., mapping is topology preserving) where as those obtained from Conscience are only locally ordered.

Figures 2.26 and 2.27 depict that more number of iterations are required for the
Figure 2.25  Final weights obtained for Data Set II after 1000000 iterations. **Left:** Those obtained by Conscience algorithm (with a neighborhood of 1 around the winner) are only partially ordered. **Right:** Those obtained by BDH algorithm (with $\alpha_{\text{desired}} = 1$) are globally ordered.

convergence of the Conscience algorithm than that of BDH algorithm.

Figure 2.26  Comparing the results after 100000 iterations for Data Set II. **Left:** $\alpha_{\text{achieved}} = 0.9334$ by Conscience algorithm (with a neighborhood of 1 around the winner). **Right:** $\alpha_{\text{achieved}} = 1.0225$ by BDH (with $\alpha_{\text{desired}} = 1$).

Table 2.2 summarizes the comparison between Conscience with zero neighborhood and BDH with $\alpha_{\text{desired}} = 1$. We can see that the entropy of the map obtained by BDH is higher than that of the map obtained by Conscience.
Figure 2.27  Comparing the results 100000 iterations for Data Set II. **Left:** $\alpha_{\text{achieved}} = 0.9555$ by Conscience algorithm (with a neighborhood of 1 around the winner). **Right:** $\alpha_{\text{achieved}} = 1.0275$ by BDH (with $\alpha_{\text{desired}} = 1$).

Table 2.2  Comparing BDH with $\alpha_{\text{desired}} = 1$ with Conscience with 1 neighborhood

<table>
<thead>
<tr>
<th>Iteration number</th>
<th>Data Set</th>
<th>Algorithm</th>
<th>$\alpha_{\text{achieved}}$</th>
<th>Entropy of the map</th>
</tr>
</thead>
<tbody>
<tr>
<td>100000</td>
<td>I</td>
<td>Conscience</td>
<td>0.9585</td>
<td>4.7069</td>
</tr>
<tr>
<td>100000</td>
<td>I</td>
<td>BDH (with $\alpha_{\text{desired}} = 1$)</td>
<td>0.9825</td>
<td>6.6254</td>
</tr>
<tr>
<td>1000000</td>
<td>I</td>
<td>Conscience</td>
<td>0.9404</td>
<td>4.7799</td>
</tr>
<tr>
<td>1000000</td>
<td>I</td>
<td>BDH (with $\alpha_{\text{desired}} = 1$)</td>
<td>0.9995</td>
<td>6.6250</td>
</tr>
<tr>
<td>100000</td>
<td>II</td>
<td>Conscience</td>
<td>0.9334</td>
<td>4.8825</td>
</tr>
<tr>
<td>100000</td>
<td>II</td>
<td>BDH (with $\alpha_{\text{desired}} = 1$)</td>
<td>1.0225</td>
<td>6.6224</td>
</tr>
<tr>
<td>1000000</td>
<td>II</td>
<td>Conscience</td>
<td>0.9555</td>
<td>4.5722</td>
</tr>
<tr>
<td>1000000</td>
<td>II</td>
<td>BDH (with $\alpha_{\text{desired}} = 1$)</td>
<td>1.0275</td>
<td>6.6225</td>
</tr>
</tbody>
</table>

2.5  BDH using periodic SOM and periodic data

Undesirable boundary effects can be observed in some cases like Figures 2.2 (Right), 2.9 (Right) and 2.11 (Right) where the final weights are not stretched from
0 to 1 as they should be. When an input is given to the Self-Organizing Map, a PE is decided as the winner and the weight of this winner PE and those of PEs around it are modified to move closer to the input by a value which is a function of the distance of the PE from the winner PE (Equation 1.2). When a PE at the ends of the 1-dimensional SOM is the winner, the weights of PEs on only one side of it learn the corresponding input. When a PE not close to the ends is the winner, weights of PEs on both sides of it are changed to move closer to the input. So the input values which are towards the higher and lower limits of the input range (which when input to the SOM have PEs on the ends of the SOM to be the winners) may not get proper representation in the SOM. Utilizing a technique of making the SOM and data periodic corrects this problem. In this technique, with input \( v \), winner \( i(v) \) is

**Case I:** If \( 0.2 < v < 0.8 \),

\[
i(v) = \arg \min_j ||v - w_j||, \quad j = 1, \ldots, N
\]

**Case II:** If \( v \leq 0.2 \),

\[
i_1(v) = \arg \min_j ||v - w_j||, \quad j = 1, \ldots, N
\]
\[
i_2(v) = \arg \min_j ||(1 + v) - w_j||, \quad j = 1, \ldots, N
\]

if \( ||v - w_{11}|| < ||(1 + v) - w_{21}|| \), \( i(v) = i_1(v) \)

else \( i(v) = i_2(v) \)

**Case III:** If \( v \geq 0.8 \),

\[
i_1(v) = \arg \min_j ||v - w_j||, \quad j = 1, \ldots, N
\]
\[ i_2(v) = \arg \min_j || (1 - v) - w_j ||, \quad j = 1, \ldots, N \]

\[
\text{if } ||v - w_1|| < ||(1 - v) - w_2||, \quad i(v) = i_1(v) \\
\text{else } i(v) = i_2(v)
\]

This way of winner determination takes care of making the SOM periodic. Figure 2.28 describes how the input space is made periodic. We were able to remove the undesirable boundary effects by making use of this technique of making the SOM and the input space periodic (Figures 2.29 (Left) and (Right)).

### 2.6 One- and two-step procedures for magnification control

The two-step procedure for magnification control is to use \( \alpha_{\text{desired}} = 1 \) for some iterations and then use the actual \( \alpha_{\text{desired}} \) in the BDH algorithm. Here we were investigating if the performance of the BDH algorithm could be improved by applying BDH with \( \alpha_{\text{desired}} \) on an equiprobabilistic map (obtained by using BDH with \( \alpha_{\text{desired}} = 1 \) for some iterations in the beginning). Figure 2.30 shows the values of \( \alpha_{\text{achieved}} \) for the one- and two-step procedures for the two data sets. We can see that the two-step procedure performs almost the same as the one-step procedure.

### 2.7 Conclusions on magnification control for 1-dimensional data

The main conclusions from experiments on 1-dimensional data are the following:

**Performance of the BDH algorithm** From the comparison of the values of \( \alpha_{\text{desired}} \) and \( \alpha_{\text{achieved}} \) for Data Sets I and II, the following conclusions were drawn:
To Ensure that the Input Space is Periodic

Case I \( v > 0.5 \)
(The winner will have a value close to \( v \))

We will be trying to ensure that the all weights on the left and right of the input (in a periodic sense) are moved closer to the input.

Updating the weights of the PEs
- If the value of the weight lies between \((v - 0.5)\) and 1,
  it is updated using \( v \) as the input.
- Else (if it lies in the range between 0 and \( v - 0.5 \)),
  the updating is done with the input being \(-(1 - \phi)\)
  
  If weight < 0
  weight = weight + 1

(This way the input will be periodic)

Case II \( v < 0.5 \)
(The winner will have a value close to \( v \))

We will be trying to ensure that the all weights on the left and right of the input (in a periodic sense) are moved closer to the input.

Updating the weights of the PEs
- If the value of the weight lies between 0 and \((v + 0.5)\),
  it is updated using \( v \) as the input.
- Else (if it lies in the range between \( v + 0.5 \) and 1),
  the updating is done with the input being \((1 - \phi)\)
  
  If weight > 1
  weight = 1 - weight

(This way the input will be periodic)

Figure 2.28 Making the input space periodic.
By utilizing the technique for making the SOM and the input space periodic undesirable boundary effects can be removed.

Comparing one- and two-step procedures for magnification control by plotting $\alpha_{achieved}$ and $\alpha_{desired}$. **Left:** For Data Set I. **Right:** For Data Set II.

- Plots comparing $\alpha_{desired}$ and $\alpha_{achieved}$ show that BDH performs best in the range $0.1 \leq \alpha_{desired} \leq 1.0$
- The mapping with BDH is topology preserving. This is shown by the fact that the weights are ordered.
- It is possible to induce $\alpha < 0$ although the achieved value may differ considerably from the desired value.
- The algorithm doesn’t work well for $\alpha_{desired} > 1$. 
BDH with $\alpha_{achieved} = 1$ and Conscience Algorithm The performance of BDH with $\alpha_{desired} = 1$ is better than that of Conscience algorithm in terms of the following:

- Global ordering of weights is achieved with the BDH algorithm (with $\alpha_{achieved} = 1$) whereas the best Conscience can do is local ordering when used with neighborhood = 1, when used with a neighborhood of zero, the weights are not ordered at all.

- Less number of iterations are required for convergence with BDH as compared to Conscience.

- The map entropy is closer to the maximum for the map obtained by BDH (with $\alpha_{achieved} = 1$) as compared to that of the map obtained by Conscience.

BDH with periodic SOM and data Undesirable boundary effects were observed in some cases, which were minimized by utilizing a technique of making the SOM and data periodic.

One and two-step procedures For the 1-D data case, the two-step procedure (using $\alpha_{desired} = 1$ in the algorithm for the first few iterations and then using the actual $\alpha_{desired}$) performs almost the same as the one-step procedure.
Chapter 3
Magnification control for 2-dimensional data

Available theory guarantees successful performance of the BDH algorithm only for 1-dimensional, and for a special case of 2-dimensional data. After confirming that the BDH algorithm performs well on 1-dimensional data (Chapter 2), we investigated its performance on “forbidden data”. In this chapter, we discuss the simulations we carried out to chart the performance of BDH algorithm on “allowed” and “forbidden” 2-dimensional data. We analyzed the following three cases:

1. Data in two dimensions independent, i.e. \( p(v) = p(v_1)p(v_2) \). This is theoretically “allowed” data (Chapter 1).

2. Weakly correlated data in two dimensions, i.e., small correlation coefficient, \( \rho_{v_1v_2} \ll 1 \). This comes under the category of “forbidden” data (Chapter 1).

3. Highly correlated data in two dimensions, i.e. large correlation coefficient, \( \rho_{v_1v_2} \approx 1 \). This too is theoretically “forbidden” data (Chapter 1), and more so than 2.

We generated several 2-dimensional data sets with pdf's satisfying one of the above properties and carried out simulations with different values of \( \alpha_{desired} \) in the BDH algorithm. The histogram based method used for \( \alpha \) evaluation for 1-dimensional data (Chapter 2) was extended to 2-dimensions to evaluate the value of \( \alpha_{achieved} \) by
Figure 3.1 2-dimensional data samples $v = (v_1, v_2)$, such that $p_v(v) = p_{v_1}(v_1)p_{v_2}(v_2)$, i.e., data is independent in the two dimensions.

the map in the case of 2-dimensional data (detailed in Appendix A). To assess the performance of the algorithm, $\alpha_{desired}$ and $\alpha_{achieved}$ were compared for several values of $\alpha_{desired}$. Three 2-dimensional data sets and BDH simulations on them are discussed next.

3.1 Data independent in the two dimensions

In this case, the pdf of the two-dimensional data samples $v$ satisfies the following condition: $p_v(v) = p_{v_1}(v_1)p_{v_2}(v_2)$. As discussed in Chapter 1, BDH algorithm is theoretically justified to work in this case. A 2-dimensional data set with a pdf having the above property was generated and simulations were carried out with different values of $\alpha_{desired}$ in the BDH algorithm.
3.1.1 Data description

This data set, consisting of 1000 2-dimensional vectors was generated according to the following pdf (Figure 3.1):

\[ p_{v_1}(v_1) = 2v_1, \quad v_1 \in [0, 1] \quad \text{and} \quad p_{v_2}(v_2) = 2v_2, \quad v_2 \in [0, 1] \]

\[ p_v(v) = 2v_1v_2, \quad v_1, v_2 \in [0, 1] \quad (3.1) \]

3.1.2 Simulation results

Figures 3.2 depicts the results of BDH with \( \alpha_{\text{desired}} = 0.6 \) and 0.3 respectively. Weights belonging to adjacent PEs in the SOM are connected. Since the weights of PEs adjacent in the SOM are nearby in the input space and vice-versa, the mapping is topology preserving. Figure 3.3 demonstrates the performance of BDH on this data set. \( \alpha_{\text{achieved}} \) and \( \alpha_{\text{desired}} \) match approximately. However, even though this data set is

![Figure 3.2](image)

**Figure 3.2** 2-dimensional input data samples (small dots) and distribution of SOM weights (larger dots), resulting from the application of the BDH algorithm. Weights belonging to adjacent PEs in the SOM are connected. 2-dimensional input samples \( v = (v_1, v_2) \) are such that \( v_1 \) and \( v_2 \) are independent. **Left:** BDH applied with \( \alpha_{\text{desired}} = 0.6, \alpha_{\text{achieved}} = 0.6075. **Right:** BDH applied with \( \alpha_{\text{desired}} = 0.3, \alpha_{\text{achieved}} = 0.4046.**
classified under “allowed data”, we still see some discrepancies between the values of \( \alpha_{\text{achieved}} \) and \( \alpha_{\text{desired}} \). These are largely due to the fact that the process of evaluation of \( \alpha_{\text{achieved}} \) is not accurate and also because the theoretical results are asymptotic and we only have a finite number of PEs (100).

![Image of a graph comparing \( \alpha_{\text{achieved}} \) with \( \alpha_{\text{desired}} \)](image)

**Figure 3.3** BDH performance when 2-D input samples \( v = (v_1, v_2) \) are such that \( v_1 \) and \( v_2 \) are independent. \( \alpha_{\text{achieved}} \) and \( \alpha_{\text{desired}} \) match approximately. The discrepancies are largely due to the fact that the theoretical results are asymptotic and we only have a finite number of PEs (100) and also because the process of evaluation of \( \alpha_{\text{achieved}} \) is not very accurate.

In Chapter 1, we discussed that the relation between the parameter \( m \) used in the BDH algorithm and \( \alpha_{\text{desired}} \) is given by

\[
\alpha_{\text{desired}} = \frac{2}{3} (1 + m)
\]  

(3.2)

This relation was given out in [13] based on the analysis in [25] where the theoretical value of \( \alpha_{\text{achieved}} \) for KSOM was found to be 2/3*. For “forbidden” data, we carried

\*\( \alpha_{\text{achieved}} = 2/3 \) for KSOM holds only if either the data is 1-dimensional or 2-dimensional with the property that \( p_v(v) = p_{v_1}(v_1)p_{v_2}(v_2) \).
out experiments on two variants of the BDH algorithm. According to Equation 3.2, we used \( m = \frac{3}{2} \alpha_{\text{desired}} - 1 \) in one variant and in the other, we used \( m = \frac{1}{\alpha_{\text{KSOM}}} \alpha_{\text{desired}} - 1 \), where \( \alpha_{\text{KSOM}} \) is the value of \( \alpha_{\text{achieved}} \) evaluated for the map generated by the application of KSOM on the data set. \( m = \frac{1}{\alpha_{\text{KSOM}}} \alpha_{\text{desired}} - 1 \) for the second version of BDH was obtained by replacing \( 2/3 \) by \( \alpha_{\text{KSOM}} \) in Equation 3.2. We compared the values of \( \alpha_{\text{achieved}} \) obtained by the two variants of BDH with the same \( \alpha_{\text{desired}} \). Next, we discuss these experiments on two “forbidden” 2-dimensional data sets.

### 3.2 Weakly correlated data in two dimensions: \( \rho_{v_1v_2} \ll 1 \)

In this case, the data consists of 2-dimensional samples, \( v = (v_1, v_2) \) such that \( \rho_{v_1v_2} \ll 1 \). This is “forbidden data” where supporting theory does not guarantee success.
3.2.1 Data description

This data set consists of 1000 2-dimensional data vectors, \( v = (v_1, v_2) \) of two kinds in equal numbers. One is such that \( v_2 = v_1 + n \) and the other is \( v_2 = -v_1 + n \), where \( n = \mathcal{N}(0, 0.0625) \). \( v_1 \) and \( v_2 \) are weakly correlated with the correlation coefficient \( \rho_{v_1v_2} = 0.0044 \) (Figure 3.4).

3.2.2 Simulation results

Figure 3.5 depicts the results of BDH with \( \alpha_{\text{desired}} = 1.0 \) for the two versions of BDH (using \( m = \frac{3}{2} \alpha_{\text{desired}} - 1 \) and \( m = \frac{1}{\alpha_{\text{KSOM}}} \alpha_{\text{desired}} - 1 \) respectively). Figure 3.6 corresponds to \( \alpha_{\text{desired}} = 0.4 \). For this data set, \( \alpha_{\text{KSOM}} \) was found to be 0.789289. In all the cases, weights of PEs adjacent in the SOM are nearby in the input space and vice-versa, so the mapping is topology preserving.

![Data samples and weights for \( \alpha_{\text{desired}} = 1 \)](image)

Figure 3.5 2-dimensional input data samples (small dots) and distribution of SOM weights (larger dots), resulting from application of the BDH algorithm with \( \alpha_{\text{desired}} = 1.0 \). Weights belonging to adjacent PEs in the SOM are connected. 2-dimensional input samples \( v = (v_1, v_2) \) are such that \( \rho_{v_1v_2} \ll 1 \). This is a case of “forbidden data” where supporting theory does not guarantee success. \( \alpha_{\text{KSOM}} \) was found to be 0.789289. Left: BDH applied with \( m = \frac{3}{2} \alpha_{\text{desired}} - 1 \), \( \alpha_{\text{achieved}} = 0.992492 \). Right: BDH applied with \( m = \frac{1}{\alpha_{\text{KSOM}}} \alpha_{\text{desired}} - 1 \), \( \alpha_{\text{achieved}} = 0.985485 \).
Figure 3.6 2-dimensional input data samples (small dots) and distribution of SOM weights (larger dots), resulting from application of the BDH algorithm with $\alpha_{\text{desired}} = 0.4$. Weights belonging to adjacent PEs in the SOM are connected. 2-dimensional input samples $v = (v_1, v_2)$ are such that $\rho_{v_1v_2} \ll 1$. This is a case of "forbidden data" where supporting theory does not guarantee success. $\alpha_{KSOM}$ was found to be 0.789289. **Left:** BDH applied with $m = \frac{2}{3} \alpha_{\text{desired}} - 1$, $\alpha_{\text{achieved}} = 0.644745$. **Right:** BDH applied with $m = \frac{1}{\alpha_{KSOM} \alpha_{\text{desired}} - 1}$, $\alpha_{\text{achieved}} = 0.552653$.

Figure 3.7 depicts the performance of the BDH algorithm on this data set. It can be seen that $\alpha_{\text{achieved}}$ and $\alpha_{\text{desired}}$ are almost equal at $\alpha_{\text{desired}} = 1$ and the two values differ increasingly (but in a predictable manner) as $\alpha_{\text{desired}}$ decreases. This is a stronger result than available from theory, as the theory only guaranteed success if and only if $v_1$ and $v_2$ were independent. Also, the match between $\alpha_{\text{achieved}}$ and $\alpha_{\text{desired}}$ is better when using experimentally evaluated $\alpha_{KSOM}$ instead of $2/3$. This is an interesting result, but not of great practical value because using the value $\alpha_{KSOM}$ in BDH for magnification control involves evaluation of $\alpha_{\text{achieved}}$ by the map resulting from the application of KSOM, which is not an easy task in general (Appendix A).

### 3.3 Strongly correlated data in two dimensions: $\rho_{v_1v_2} \approx 1$

This case, when the correlation coefficient $\rho_{v_1v_2} \approx 1$, is also "forbidden data".
Figure 3.7  BDH performance when 2-D input samples $v = (v_1, v_2)$ are such that 2-dimensional input samples $v = (v_1, v_2)$ are such that $\rho_{v_1v_2} \ll 1$. This is a case of “forbidden data” where supporting theory does not guarantee success. $\alpha_{\text{achieved}}$ and $\alpha_{\text{desired}}$ are almost equal at $\alpha_{\text{desired}} = 1$ and the two values differ increasingly (but in a predictable manner) as $\alpha_{\text{desired}}$ decreases. This is a stronger result than available from theory, as the theory only guaranteed success if and only if $v_1$ and $v_2$ were independent. The match between $\alpha_{\text{achieved}}$ and $\alpha_{\text{desired}}$ is better when using experimentally evaluated $\alpha_{KSOM}$ instead of $2/3$.

3.3.1 Data description

Figure 3.8  2-dimensional data samples $v = (v_1, v_2)$, such that $\rho_{v_1v_2} \approx 1$, i.e., data in the two dimensions is strongly correlated.
This data set consists of 1000 2-dimensional samples, \( v = (v_2, v_1) \), such that \( v_2 = v_1 + n \), where \( n = \mathcal{N}(0,0.25) \). The correlation coefficient is \( \rho_{v_1v_2} = 0.9026 \) (Figure 3.8).

### 3.3.2 Simulation results

The left and right panels of Figure 3.9 depict the results of BDH with \( \alpha_{\text{desired}} = 1.0 \) for the two versions of BDH (using \( m = \frac{3}{2} \alpha_{\text{desired}} - 1 \) and \( m = \frac{1}{\alpha_{\text{K SOM}}} \alpha_{\text{desired}} - 1 \), respectively). Figure 3.10 shows the same for \( \alpha_{\text{desired}} = 0.4 \). For this data set, \( \alpha_{\text{K SOM}} \) was found to be 0.905405. In all the cases, weights of PEs adjacent in the SOM are nearby in the input space and vice-versa, so the mapping is topology preserving.

![Figure 3.9](image)

**Figure 3.9** 2-dimensional input data samples (small dots) and distribution of SOM weights (larger dots), resulting from application of the BDH algorithm with \( \alpha_{\text{desired}} = 1.0 \). Weights belonging to adjacent PEs in the SOM are connected. 2-dimensional input samples \( v = (v_1, v_2) \) are such that \( \rho_{v_1v_2} \approx 1 \). This is a case of “forbidden data” where supporting theory does not guarantee success. \( \alpha_{\text{K SOM}} \) was found to be 0.905405. **Left:** BDH applied with \( m = \frac{3}{2} \alpha_{\text{desired}} - 1 \), \( \alpha_{\text{achieved}} = 1.262763 \). **Right:** BDH applied with \( m = \frac{1}{\alpha_{\text{K SOM}}} \alpha_{\text{desired}} - 1 \), \( \alpha_{\text{achieved}} = 0.972472 \).

Figure 3.11 depicts the performance of BDH on this data set. In this strongly
Figure 3.10  2-dimensional input data samples (small dots) and distribution of SOM weights (larger dots), resulting from application of the BDH algorithm with $\alpha_{desired} = 0.4$. Weights belonging to adjacent PEs in the SOM are connected. 2-dimensional input samples $\mathbf{v} = (v_1, v_2)$ are such that $\rho_{v_1v_2} \approx 1$. This is a case of “forbidden data” where supporting theory does not guarantee success. $\alpha_{K_{SOM}}$ was found to be 0.905405. **Left**: BDH applied with $m = \frac{2}{3} \alpha_{desired} - 1$, $\alpha_{achieved} = 0.701802$. **Right**: BDH applied with $m = \frac{1}{\alpha_{K_{SOM}}} \alpha_{desired} - 1$, $\alpha_{achieved} = 0.591692$.

correlated case, even though $\alpha_{achieved}$ by the map differs from $\alpha_{desired}$, there is a clearly observable trend that the values of $\alpha_{achieved}$ are systematically decreasing, following the desired values with a more or less constant shift. This is again a stronger result than the theory provides and encourages further investigation of BDH for real data, including higher dimensional data. The match between $\alpha_{achieved}$ and $\alpha_{desired}$ is once again better when using experimentally evaluated $\alpha_{K_{SOM}}$ instead of $2/3$. This is interesting, but as stated before, not of great practical value because using the value $\alpha_{K_{SOM}}$ in BDH for magnification control requires evaluation of $\alpha_{achieved}$ by the map resulting from the application of KSOM, which can be difficult in general (Appendix A).

The results of BDH on “forbidden” 2-dimensional data are stronger than what the theory provides and encourage further investigation of BDH for real data, including
Figure 3.11  BDH performance when 2-D input samples $v = (v_1, v_2)$ are such that 2-dimensional input samples $v = (v_1, v_2)$ are such that $\rho_{v_1v_2} \ll 1$. This is a case of “forbidden data” where supporting theory does not guarantee success. Here, even though $\alpha_{\text{achieved}}$ by the map differs from $\alpha_{\text{desired}}$, there is a clearly observable trend that the values of $\alpha_{\text{achieved}}$ are systematically decreasing, following the desired values with a more or less constant shift. This is again a stronger result than the theory provides. The match between $\alpha_{\text{achieved}}$ and $\alpha_{\text{desired}}$ is better when using experimentally evaluated $\alpha_{K\text{SOM}}$ instead of $2/3$.

higher dimensional data. In the next two chapters, some interesting results of applying BDH to higher dimensional data are discussed.

3.4 Conclusions on magnification control for 2-dimensional data

Main conclusions from experiments on 2-dimensional data:

- If the 2-dimensional data samples are independent in the two dimensions, application of BDH algorithm produces values of $\alpha_{\text{achieved}}$ that are close to the corresponding values of $\alpha_{\text{desired}}$.

- If data in the two dimensions are weakly correlated i.e., $\rho \ll 1$, $\alpha_{\text{achieved}}$ and
\( \alpha_{\text{desired}} \) are almost equal at \( \alpha = 1 \) and the two values differ in a predictably increasingly manner as \( \alpha \) decreases. This is a stronger result than available from theory, as the theory only guaranteed success if and only if \( v_1 \) and \( v_2 \) were independent.

- If data in the two dimensions are highly correlated i.e., \( \rho \approx 1 \), even though \( \alpha_{\text{achieved}} \) by the map differs from the \( \alpha_{\text{desired}} \), there is a clear trend that the values of \( \alpha_{\text{achieved}} \) are systematically decreasing, following the desired values with a more or less constant shift. This is again a stronger result than the theory provides and encourages further investigation of BDH for real data, including higher dimensional data.

The above mentioned trends in magnification control were observed for \( 0.4 \leq \alpha_{\text{desired}} \leq 1 \), in all cases.

- In the BDH algorithm, the factor 'm' is used as a parameter. We carried out simulations on two variants of BDH: one with \( m = \frac{3}{2} \alpha_{\text{desired}} - 1 \) and the other with \( m = \frac{1}{\alpha_{KSOM}} \alpha_{\text{desired}} - 1 \), where \( \alpha_{KSOM} \) is the value of \( \alpha_{\text{achieved}} \) evaluated for the map obtained by the application of KSOM on the data set. Performance of the BDH algorithm, i.e., the match between the values of \( \alpha_{\text{achieved}} \) and \( \alpha_{\text{desired}} \) is better when evaluated \( \alpha_{KSOM} \) is used in BDH rather than \( 2/3 \).

- Mapping obtained by the BDH algorithm was topology preserving.
Chapter 4
Magnification control for higher dimensional data

Encouraged by the results obtained for the performance of BDH on 2-dimensional "forbidden" data, we carried out experiments on high-dimensional data. In this chapter, we discuss those simulations and identify two useful regions of the magnification factor, for applicability of the BDH algorithm for high-dimensional (and hence "forbidden") data.

4.1 Evaluation of BDH on noiseless, 6-dimensional, 5- and 20-class data sets

Our methodology to examine the performance of the BDH algorithm on any 1-and 2-dimensional data sets was to use a certain value of $\alpha_{desired}$ in the BDH algorithm, evaluate the corresponding value of $\alpha_{achieved}$ by the map and compare the two. However, evaluation of $\alpha_{achieved}$ is not an easy task in general, especially if the input pdf is unknown (as is most commonly the case). The evaluation of $\alpha_{achieved}$ involves the estimation of the pdf of the data and the pdf of the weights. So far, we were using a histogram based method for $\alpha$ evaluation. This method becomes inapplicable for high-dimensional data as the number of samples required for pdf estimation increases exponentially with dimensionality (Appendix A). So for general, higher dimensional cases, we evaluate of the performance of the algorithm indirectly: by observing the resulting map. In case the data is of known characteristics it may be possible to
evaluate the value of $\alpha_{\text{achieved}}$ by the map. For assessing the performance of BDH in case of higher dimensional data, we generated several 6-dimensional data sets. BDH was applied on these and conclusions were drawn from the results obtained. In this section we discuss the simulations carried out on these two noiseless, data sets, one with 5, and other with 20 known classes.

4.1.1 Description of the noiseless 5- and 20-class data sets

The 5-class data set

This data set is a $128 \times 128$ pixel image where each pixel has a 6-dimensional vector associated with it. It has 5 classes whose 6-dimensional signatures are shown in Figure 4.1 (Left). Class $U$ is a rare class with only 1 data point of this kind. The rest of the classes have 4096 or 4095 data points each. In this data set, correlation coefficients between the different dimensions range from $0.004(\rho_{v2v3})$ to $0.9924(\rho_{v3v6})$, which renders this a “forbidden” case for application of the BDH.

The 20-class data set

This data set is similar to the 5-class image except it has 20 classes, as shown in Figure 4.1 (Right). Two of the classes, marked $R$ and $Q$ are relatively rare, with only 1 and 16 data points, respectively. Correlation coefficients between the different dimensions range from $0.0081(\rho_{v3v4})$ to $0.5641(\rho_{v4v2})$, so this too is a “forbidden” case for the BDH according to the available theory.

These two data sets are noiseless - all the data vectors belonging to a particular
class are exactly the same.

4.1.2 Simulations on the noiseless 5- and 20-class data sets

Finding rare classes in a data set is a challenging task. Input classes with rare occurrence find little or no representation in the map when KSOM is used. Application of BDH with $\alpha_{desired} < 0$ would result in negative magnification: classes which are
rare in the input will be magnified in the map, i.e., the number of PEs representing
the rare classes will be higher. This is a promising technique for detection of rare
classes. In this section, we examine the application of BDH with $\alpha_{desired} < 0$ on the
5- and 20-class data sets. These maps will be compared to those obtained by KSOM.

5-class data set

When KSOM is used on this data set, the rare class is represented by only one PE
(Figure 4.2 (Left)). BDH with $\alpha_{desired} = -0.8$, magnifies the rare class in the map:
it is represented by 10 PEs (Figure 4.2 (Right)).

![Figure 4.2](image)

**Figure 4.2** Results of clustering of the 5-class data set. **Left:** Using KSOM. Top) Weight
vectors in the $10 \times 10$ SOM. Only 1 PE represents the rare class U. Bottom Left) Clusters identified in
the map. The darker the fence between two PEs, the smaller the difference between the corresponding
weights. Bottom Right) This figure shows which class each weight vector is closest to, which
complements the information on the left. **Right:** Using BDH with $\alpha_{desired} = -0.8$. Top) Weight
vectors in the $10 \times 10$ SOM. The rare class U is now represented by 10 PEs! Bottom Left) Clusters
identified in the SOM. Bottom Right) This figure shows which class each weight vector is closest to,
which complements the information on the left.
20-class data set

![Image of clustering results](image)

**Figure 4.3** Results of clustering of the 20-class data set. **Left:** Using KSOM. Top) Weight vectors in the 10 × 10 map. Only 1 PE is occupied by each of the rare classes R and Q. Bottom Left) clusters identified in the SOM. The lighter the color of the fence between two PEs, the larger the difference between the corresponding weights. Bottom Right) This figure shows which class each weight vector is closest to, which complements the information on the left. **Right:** Using BDH with $\alpha_{desired} = -0.8$. Top) Weight vectors in the 10 × 10 SOM. 4 and 7 PEs now represent the rare classes R and Q respectively. Bottom Left) Clusters identified in the SOM. Bottom Right) This figure shows which class each weight vector is closest to, which complements the information on the left.

Clustering this data set using KSOM is depicted in Figure 4.3 (Left). The rare classes are detectible but each one is poorly represented, by a single PE only. Also, the lack of strong fences separating them from surrounding PEs makes them less discernible (the ‘fences’ represent the Euclidean distance between the weights of adjacent PEs. Black-to-white corresponds to low-to-high differences). BDH with $\alpha_{desired} = -0.8$ magnifies the rare classes in the map. Figure 4.3 (Right) shows that class R is now represented by 4 PEs as opposed to 1 in the map formed by KSOM, and
class $Q$ by approximately 7 PEs. So once again, the rare classes have been magnified and their separation has also been made more discernable by BDH with $\alpha_{desired} < 0$.

To conclude:

- Classes that are rare in the input space, and find little or no representation in the map formed by the use of KSOM, are magnified in the map formed by BDH with $\alpha_{desired} < 0$, thereby making their detection possible. So, this is a promising technique for the detection of rare classes.

The above results on the noiseless 5- and 20-class data sets are reported in our recent paper [32].

4.2 Evaluation of BDH on noisy 6-dimensional, 8- and 11-class data sets

The noisy 8-class and 11-class data sets are two somewhat specialized (but still fairly general) higher dimensional synthetic data sets for which it was possible to calculate the value of $\alpha_{achieved}$ by the map. The performance of the BDH algorithm was analyzed by evaluating the value of $\alpha_{achieved}$ and comparing it with $\alpha_{desired}$.

4.2.1 Description of the noisy 8- and 11-class data sets

These two data sets are similar to the 5- and 20-class data sets in that they are also $128 \times 128$ pixel images where each pixel has a 6-dimensional vector associated with it. These are, however, noisy data sets. Each class consists of vectors which are
Figure 4.4  Description of the noisy 8- and 11-class data sets. Mean signatures of the classes and their distribution over the subareas of the image. **Left:** 8-class data set. **Right:** 11-class data set.

Gaussian distributed with the same variance around a mean vector. On the average about 10% of Gaussian noise was added to create the spectral variations within each class.
8-class data set

The mean signatures of the various classes and their distribution over the subareas of the image are shown in Figure 4.4 (Left). Classes A and B each cover 4096 pixels, classes C and O have 2048 pixels, and classes D, H, I, M have 1024 pixels. Gaussian noise about 10% on average, was added to create realistic variations within the spectral classes. The pairwise correlations of the various dimensions of this data set are typically strong, most between 0.3 – 0.8 and only two are less than 0.05, so this too is a “forbidden” case for the BDH algorithm according to the available theory.

11-class data set

This data set is similar to the 8-class image except it has 11 classes, as shown in Figure 4.4 (Right). Three of the classes marked R, Y and T are relatively rare, with only 16, 64 and 128 data vectors respectively. Once again, Gaussian noise about 10% on average, was added to create realistic variations within the spectral classes. Correlation coefficients between the different dimensions range from 0.0584(\(\rho_{uvuv}\)) to 0.887(\(\rho_{uvuv}\)), making this also is a “forbidden” case for BDH application.

4.2.2 Procedure for evaluating \(\alpha_{achieved}\)

We can see that the noisy 8-class and 11-class data sets are fairly general in the sense that they are high-dimensional and have quite high pairwise correlation values between different dimensions. However, because of the distribution of the data
samples (Gaussian with known variance around a known mean), they are specialized
each class, \([n_{\text{achieved,1}}, n_{\text{achieved,2}}, \ldots, n_{\text{achieved,N}}]\)
where \(N\) denotes the number of classes. The weights belonging to a particular class
are those that are closest to the mean vector of that class than to that of any other
class. Using the known distribution of the data and by using the power law relating
weight density and input distribution (Equation 1.3), we calculated the theoreti-
cal number of weights in each class for \(M\) different values of \(\alpha\) \((n_{\text{theoretical,1}}(\alpha = \alpha_i), n_{\text{theoretical,2}}(\alpha = \alpha_i), \ldots, n_{\text{theoretical,N}}(\alpha = \alpha_i))\), \(i = 1, 2, \ldots, M\) and check which
set of numbers best match the achieved numbers to determine the value of \(\alpha_{\text{achieved}}\),
i.e.,

\[
\alpha_{\text{achieved}} = \arg \min_i \sum_{j=1}^N (n_{\text{theoretical,cj}}(\alpha = \alpha_i) - n_{\text{achieved,cj}})^2
\]  
(4.1)

The evaluated value of \(\alpha_{\text{achieved}}\) is only approximate because the obtained theoretical
number of weights is approximate. Even though the numbers may not be exact, we
still get a fair idea of the magnification and hence of the performance of the BDH
algorithm.

4.2.3 Simulations on the noisy 8- and 11-class data sets

We applied BDH with several values of \(\alpha_{\text{desired}}\) between \(-0.8\) and \(1.1\) and evaluated
the corresponding values of \(\alpha_{\text{achieved}}\) by the map. A curve comparing the two sets of
values depicts the overall performance of BDH. We also compared the performance of
BDH algorithm with the Conscience algorithm and the KSOM algorithm to determine which self-organizing map algorithm has the best performance for given (various) $\alpha_{desired}$ values.

We plotted the achieved and calculated theoretical number of weights for the corresponding $\alpha_{desired}$, for each class as another visualization to assess the magnification achieved.

A $20 \times 20$ SOM was used for these simulations.

4.2.4 Noisy 8-class data set

Conscience and BDH with $\alpha_{desired} = 1$

![Diagram](image)

Figure 4.5 SOMs resulting from the application of Conscience and BDH with $\alpha_{desired} = 1$ on noisy 8-class data set. The weight vectors of the PEs are plotted. The PE belonging to a particular class has the same color as that used to represent the class in Figure 4.4 (Left). **Left:** Result of applying BDH with $\alpha_{desired} = 1$, $\alpha_{achieved} = 0.995$. **Right:** Result of applying Conscience, $\alpha_{achieved} = 0.905$. Performance of BDH with $\alpha_{desired} = 1$ is better than that of Conscience.

The Conscience algorithm [26] was designed to achieve $\alpha = 1$. Here, we compare the performance of BDH with $\alpha_{desired} = 1$ with Conscience. The results are depicted
Figure 4.6  Number of weights belonging to each class in the SOMs resulting from Conscience and BDH with $\alpha_{desired} = 1$ on noisy 8-class data set. $Error_{BDH} = 9.6954$ and $Error_{Conscience} = 11.2250$. The values of $\alpha_{achieved}$ were 0.995 and 0.905 for BDH and Conscience, respectively. So the performance of BDH with $\alpha_{desired} = 1$ is slightly better than that of Conscience. For both algorithms, the number of weights belonging to each class are close to but not exactly equal to ideal. This discrepancy is caused by the fact that the theoretical results are asymptotic and we only have a finite number of PEs. Integer arithmetic also leads to deviation from the ideal numbers.

in Figure 4.5. The spatial sizes of the various classes in the 8-class data set are in a 4 : 2 : 1 ratio (Section 4.2.1). Since $\alpha = 1$ means that the distribution of the weights is proportional to the input distribution, the number of weights belonging to each class should ideally also be in a 4 : 2 : 1 ratio. This further means that since the number of weights in the SOM is 400, the number of weights each class should ideally have is 100 for A and B, 50 for C and O and 25 for each of D, H, M and O. In our results for both Conscience and BDH with $\alpha_{desired} = 1$, the values are close to but not exactly equal to ideal. This discrepancy arises because the theoretical results are asymptotic and
we only have a finite number of PEs. Integer arithmetic also leads to deviation from the ideal numbers. In Figure 4.6, the ideal number of weights for each class have been plotted along with the numbers obtained from the SOMs resulting from the application of Conscience and BDH algorithms. The BDH algorithm results better match the ideal values (Table 4.1). So the performance of BDH with $\alpha_{\text{desired}} = 1$ is

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Error</th>
<th>$\alpha_{\text{achieved}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>BDH ($\alpha_{\text{desired}} = 1$)</td>
<td>9.6954</td>
<td>0.995</td>
</tr>
<tr>
<td>Conscience</td>
<td>11.2250</td>
<td>0.905</td>
</tr>
</tbody>
</table>

Table 4.1  Comparing BDH with $\alpha_{\text{desired}} = 1$ and Conscience

slightly better than that of Conscience. This is encouraging as this means that even for the “forbidden” data case it is possible to successfully induce $\alpha = 1$. Also, in this case using BDH with $\alpha_{\text{desired}} = 1$ is more advantageous than Conscience because not only the performance is better in terms of $\alpha_{\text{achieved}}$, but also because we know that global ordering is obtained with BDH while using Conscience can result in only a local ordering in the SOM. Evidence of less than perfect global ordering is obvious in Figure 4.5: one PE with yellow (Class D) and one with green class (Class C) designation remained disordered among other classes.
**Figure 4.7** SOMs resulting from the application of KSOM and BDH with $\alpha_{\text{desired}} = 0.6$ on the noisy 8-class data set. The weight vectors of the PEs are plotted. The PE belonging to a particular class has the same color as that used to represent the class in Figure 4.4 (Left). **Left:** Result of applying KSOM, $\alpha_{\text{achieved}} = 0.9167$. **Right:** Result of applying BDH with $\alpha_{\text{desired}} = 0.6$, $\alpha_{\text{achieved}} = 0.915$.

**KSOM and BDH with $\alpha_{\text{desired}} = 0.6$**

The results of applying BDH with $\alpha_{\text{desired}} = 0.6$ are depicted in Figure 4.7. The value of $\alpha_{\text{achieved}} = 0.915$. In [25], the theoretical value of $\alpha_{\text{achieved}}$ for KSOM, was found to be $2/3 = 0.667$ (under certain conditions only, which does not include a 6-dimensional data set like this one). Weight distributions over classes, as a result of applying KSOM and BDH on the noisy 8-class data set are depicted in Figure 4.8. The value of $\alpha_{\text{achieved}} = \alpha_{\text{KSOM}} = 0.9167$. So we observe that the value of $\alpha_{\text{achieved}} \approx 1$ for KSOM, as well as for BDH with $\alpha_{\text{desired}} = 0.6$. 
Figure 4.8  Number of weights belonging to each class in the SOMs resulting from KSOM and BDH with $\alpha_{\text{desired}} = 0.6$ on the noisy 8-class data set. $\alpha_{\text{achieved}}(BDH \text{ with } \alpha_{\text{desired}} = 0.6) = 0.915$ and $\alpha_{\text{achieved}}(KSOM) = 0.9167$. So we observe that the value of $\alpha_{\text{achieved}} \approx 1$ for KSOM, as well as for BDH with $\alpha_{\text{desired}} = 0.6$.

BDH with $\alpha_{\text{desired}} = -0.5$

Classes that are rare in the input set are enhanced in the SOM by the use of negative magnification. In this data set, there are no rare classes, but the effect of negative magnification is observed by the fact that the small classes in the data set occupy larger regions of the SOM (Figure 4.9 (Left)) as compared to the number of weights belonging to them in the case of Conscience, KSOM or BDH with $\alpha_{\text{achieved}} \approx 1$ (Figures 4.5, 4.7 (Left)). In Figure 4.9 (Right), the number of weights belonging to each class are plotted and compared to the corresponding numbers obtained with other algorithms. We can see that with negative magnification, the smaller classes
Figure 4.9  Result of applying BDH with $\alpha_{\text{desired}} = -0.5$. **Left**: SOM, the weight vectors of the PEs are plotted. The PE belonging to a particular class has the same color as that used to represent the class in Figure 4.4 (Left). The effect of negative magnification is observed by the fact that here the small classes in the data set occupy much larger regions of the SOM as compared to the number of weights belonging to them in the case of Conscience, KSOM or BDH with $\alpha_{\text{achieved}} \approx 1$ (the discrepancy for class $H$ is due to the fact that the SOM has a finite size and this class did not get enough space to spread out).

have more number of weights belonging to them as compared to the number of weights belonging to them in the case of Conscience, KSOM, or BDH with $\alpha_{\text{achieved}} \approx 1$ (the discrepancy for class $H$ is due to the fact that the SOM has a finite size and this class did not get enough space to spread out). In this case, even though the value of $\alpha_{\text{achieved}}$ is not exactly equal to $\alpha_{\text{desired}}$, we have achieved what is desired of negative magnification: reverse relationship between the input $pdf$ $(P(w))$ and weight density $(Q(w))$. This is desirable for detecting rare classes in the input data. The available theory for BDH does not guarantee anything for “forbidden” data, but we have been able to show that BDH with $\alpha_{\text{desired}} < 0$ can be used for achieving negative magnification. This is a very important result for practical applications.
Figure 4.10 $\alpha_{\text{achieved}}$ (obtained by the two variants of BDH) vs $\alpha_{\text{desired}}$ for the noisy 8-class data set. We can see that (i) BDH results in maps with $\alpha_{\text{achieved}} \approx 1$ when applied with $\alpha_{\text{desired}}$ anywhere between $\sim 0.6$ and $\sim 1.1$. (ii) When applied with $\alpha_{\text{desired}} < 0$, negative magnification is achieved. The smaller classes in the input data are magnified in the SOM. (iii) There is no definite trend in the performance of BDH when used with values of $\alpha_{\text{desired}}$ outside of the regions discussed in (i) and (ii). These are significant findings as maps with $\alpha_{\text{achieved}} \approx 1$ and $< 0$ have important practical applications.

The results detailed so far (in this chapter) were those obtained by the version of BDH where we used $m = \frac{3}{2} \alpha_{\text{desired}} - 1$ in the implementation. We also carried out experiments using the other variant of BDH in which $m = \frac{1}{\alpha_{\text{KSOM}}} \alpha_{\text{desired}} - 1$ (where $\alpha_{\text{KSOM}}$ is the value of $\alpha_{\text{achieved}}$ evaluated for the SOM generated by the application of KSOM) was used. Figure 4.10 shows $\alpha_{\text{achieved}}$ (obtained by both the variants of BDH) vs $\alpha_{\text{desired}}$ for the noisy 8-class data set. From this plot, we can draw the following conclusions about the performance of BDH algorithm on “forbidden” data:

- BDH results in maps with $\alpha_{\text{achieved}} \approx 1$ when applied with $\alpha_{\text{desired}}$ anywhere
between $\sim 0.6$ and $\sim 1.1$.

- When applied with $\alpha_{desired} < 0$, negative magnification is achieved. The smaller classes in the input data are magnified in the SOM.

- There is no definitive trend in the performance of BDH when used with values of $\alpha_{desired}$ exclusive of the regions discussed in the above two points.

Maps with $\alpha_{achieved} \approx 1$ are equiprobabilistic maps which represent the input space most faithfully. Rare classes in the input data are enhanced in the SOM when negative magnification is achieved ($\alpha_{achieved} < 0$). So, these are two values ($\alpha_{achieved} \approx 1, < 0$) of the magnification factor that have important practical applications. We have shown that BDH can provide SOMs with $\alpha_{achieved} \approx 1$ and $\alpha_{achieved} < 0$, so it has great potential for practical applications, even though the available theory does not guarantee anything about the performance of the BDH algorithm for "forbidden" data.

4.2.5 Noisy 11 class data set
Conscience and BDH with $\alpha_{desired} = 1$

For this data set too we compare the performance of the BDH algorithm with $\alpha_{desired} = 1$ the with Conscience algorithm [26] that was designed to achieve $\alpha = 1$. The results are depicted in Figure 4.11. Except for the three very small classes, the other 8 classes in this data set are in an approximate ratio of $4 : 2 : 1$ (Section 4.2.1).
Figure 4.11  SOMs resulting from the application of Conscience and BDH with $\alpha_{\text{desired}} = 1$ on the noisy 11-class data set. The weight vectors of the PEs are plotted. The PE belonging to a particular class has the same color as that used to represent the class in Figure 4.4 (Right). Left: Result of applying BDH with $\alpha_{\text{desired}} = 1$, $\alpha_{\text{achieved}} = 1.005$. Right: Result of applying Conscience, $\alpha_{\text{achieved}} = 0.945$. Performance of BDH with $\alpha_{\text{desired}} = 1$ is somewhat better than that of Conscience.

We calculated the ideal number of weights belonging to each class in an SOM with weight density proportional to the input density ($\alpha_{\text{achieved}} = 1$) and compared them with the numbers obtained for SOMs resulting from application of Conscience and BDH (with $\alpha_{\text{desired}} = 1$) to assess the performance of these algorithms (Figure 4.12). The experimentally obtained values for the two algorithms are close to but not exactly equal to ideal. This discrepancy arises because the theoretical results are asymptotic and we only have a finite number of PEs. Integer arithmetic also leads to deviation from the ideal numbers. BDH results better match the ideal values (Table 4.2, Figure 4.12). So the performance of BDH with $\alpha_{\text{desired}} = 1$ is slightly better than that of Conscience. This is encouraging as this means that even for the "forbidden" data case it is possible to successfully induce $\alpha = 1$. Also, as we have seen in this case,
Figure 4.12  Number of weights belonging to each class in the SOMs resulting from Conscience and BDH with \( \alpha_{desired} = 1 \) on the noisy 11-class data set. Error\(_{BDH} = 8.7605 \) and Error\(_{Conscience} = 14.4914 \). The values of \( \alpha_{achieved} \) were 1.005 and 0.945 for BDH and Conscience respectively. So the performance of BDH with \( \alpha_{desired} = 1 \) is slightly better than that of Conscience. One thing to note is that even in the ideal case, the rare classes have very little or no representation in the SOM. For both algorithms, the number of weights belonging to each class are close to but not exactly equal to ideal. This discrepancy arises because the theoretical results are asymptotic and we only have a finite number of PEs. Integer arithmetic also leads to deviation from the ideal numbers.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>( Error = \sqrt{\sum_{j=1}^{11} (n_{theoretical_{ij}}(\alpha = 1) - n_{achieved_{ij}})^2} )</th>
<th>( \alpha_{achieved} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>BDH (( \alpha_{desired} = 1 ))</td>
<td>8.7605</td>
<td>1.005</td>
</tr>
<tr>
<td>Conscience</td>
<td>14.4914</td>
<td>0.945</td>
</tr>
</tbody>
</table>

using BDH with \( \alpha_{desired} = 1 \) is more advantageous than Conscience, because not only the performance is better in terms of \( \alpha_{achieved} \), but also because we know that global
ordering is obtained with BDH unlike in Conscience which results in only a local ordering in the SOM. One thing to note is that even ideally, the rare classes have very little or no representation in the SOM. So, even though an SOM with \( \alpha_{\text{achieved}} = 1 \) gives a faithful picture of the input data set in terms of having number of weights belonging to a particular class proportional to the number of inputs belonging to that class, it has a drawback that rare classes (which in practical applications may be very interesting) may have little or no representation. Detection of rare and/or interesting classes may therefore not be possible in such a map. We will see how this issue is taken care of in a map with negative magnification (Section 4.2.5).

**KSOM and BDH with \( \alpha_{\text{desired}} = 0.6 \)**

![Figure 4.13 SOMs resulting from the application of KSOM and BDH with \( \alpha_{\text{desired}} = 0.6 \) on the noisy 11-class data set. The weight vectors of the PEs are plotted. The PE belonging to a particular class has the same color as that used to represent the class in Figure 4.4 (Right). **Left**: Result of applying KSOM, \( \alpha_{\text{achieved}} = 0.9017 \). **Right**: Result of applying BDH with \( \alpha_{\text{desired}} = 0.6 \), \( \alpha_{\text{achieved}} = 0.915 \). The three rare classes do find some representation in the SOM here. However, the number of weight vectors belonging to these classes are still very small and so the rare classes might go undetected.](image)
Figure 4.14  Number of weights belonging to each class in the SOMs resulting from KSOM and BDH with $\alpha_{\text{desired}} = 0.6$ on the noisy 11-class data set. $\alpha_{\text{achieved}}(BDH \, \text{with} \, \alpha_{\text{desired}} = 0.6) = 0.915$ and $\alpha_{\text{achieved}}(KSOM) = 0.9917$. These values are consistent with the observation for the 8-class data set that the value of $\alpha_{\text{achieved}} \approx 1$ for KSOM as well as for BDH with $\alpha_{\text{desired}} = 0.6$. The three rare classes do find some representation in the SOM here. However, the number of weight vectors belonging to these classes are still very small and so the rare classes might go undetected.

The results of applying BDH with $\alpha_{\text{desired}} = 0.6$ are depicted in Figure 4.13 (Left). The value of $\alpha_{\text{achieved}} = 0.915$. In [25], the theoretical value of $\alpha_{\text{achieved}}$ for KSOM, was found to be $2/3 = 0.667$ (under certain conditions only, which does not include a 6-dimensional data set like this one). Number of weights belonging to different classes as a result of applying KSOM on the noisy 11-class data set are depicted in Figure 4.14. The value of $\alpha_{\text{achieved}} = \alpha_{KSOM} = 0.9917$. These values are consistent with the observation for the 8-class data set that the value of $\alpha_{\text{achieved}} \approx 1$ for KSOM as well as for BDH with $\alpha_{\text{desired}} = 0.6$. The three rare classes do find some
representation in the SOMs here. However, the number of weight vectors belonging to these classes are still very small and so the rare classes might go undetected. The number of weights belonging to the smallest class $R$ are 1 and 2 in the SOMs resulting from the application of KSOM and BDH (with $\alpha_{\text{desired}} = 0.6$) respectively. There are 16 input vectors in class $R$. Had the rare class been rarer (yet important in a practical application), it might not have found any representation in the SOM using KSOM or BDH with $\alpha_{\text{desired}} = 0.6$.

**BDH with $\alpha_{\text{desired}} = -0.5$**

**Figure 4.15** Result of applying BDH with $\alpha_{\text{desired}} = -0.5$. **Left:** SOM, the weight vectors of the PEs are plotted. The PE belonging to a particular class has the same color as that used to represent the class in Figure 4.4 (Right). The rare classes $R$, $Y$ and $T$ occupy much larger regions of the SOM as compared to the number of weights belonging to them in the case of Conscience, KSOM or BDH with $\alpha_{\text{achieved}} \approx 1$. The classes which were rare in the input set have been enhanced in the SOM, thereby improving their detectability.

Classes which are rare in the input set are enhanced in the SOM by the use of negative magnification. In this data set, there are three rare classes, $R$, $Y$ and $T$,
Table 4.3  Number of weights belonging to the rare classes in the SOMs obtained by various algorithms

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>( n_{achieved_{R}} )</th>
<th>( n_{achieved_{Y}} )</th>
<th>( n_{achieved_{T}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>BDH ((\alpha_{desired} = 1))</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Conscience</td>
<td>0</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>BDH ((\alpha_{desired} = 0.6))</td>
<td>2</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>KSOM</td>
<td>1</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>BDH ((\alpha_{desired} = -0.5))</td>
<td>13</td>
<td>12</td>
<td>22</td>
</tr>
</tbody>
</table>

The effect of negative magnification is observed by the fact that these occupy much larger regions of the SOM (Figure 4.15 (Left)) as compared to the number of weights belonging to them in the case of Conscience, KSOM or BDH with \( \alpha_{achieved} \approx 1 \) (Figures 4.11, 4.13 (Left)). From Figure 4.15 (Right), we can see that with negative magnification, the number of weights belonging to each of the rare classes are many more as compared to the corresponding numbers obtained with other algorithms (numbers listed in Table 4.3).

So the rare classes in the input have been enhanced in the SOM thereby increasing their detectability. Naturally the representation of the non-rare classes in such a map is somewhat distorted. So, the SOM obtained with \( \alpha_{achieved} < 0 \) and the one obtained with \( \alpha_{achieved} \approx 1 \) together provide a more complete picture of the clustering of a data set.
The available theory for BDH does not guarantee anything for “forbidden” data, but we have been able to show that BDH with $\alpha_{\text{desired}} < 0$ can be used for improving the detectability of rare classes. This finding has important practical applications as oftentimes in real data the interesting classes, those that might lead to discoveries are present in very few numbers. Enhancing them using negative magnification makes their detection possible.

![Diagram](image)

**Figure 4.16** $\alpha_{\text{achieved}}$ (obtained by both the variants of BDH) vs $\alpha_{\text{desired}}$ for the noisy 11-class data set. We can see that (i) BDH results in maps with $\alpha_{\text{achieved}} \approx 1$ when applied with $\alpha_{\text{desired}}$ anywhere between $\sim 0.6$ and $\sim 1.1$. (ii) When applied with $\alpha_{\text{desired}} < 0$, even though the values of $\alpha_{\text{achieved}}$ are not negative, yet the number of weights belonging to each of the rare classes are many more as compared to the corresponding numbers obtained with other algorithms (numbers listed in Table 4.3). The rare classes in the input data are enhanced in the SOM. So, negative magnification is achieved. (iii) There is no definitive trend in the performance of BDH when used with values of $\alpha_{\text{desired}}$ outside of the regions discussed in (i) and (ii). These are important findings as maps with $\alpha_{\text{achieved}} \approx 1$ and $< 0$ have important practical applications. These observations are exactly the same as those obtained from the experiments on the noisy 8-class data set.
For this data set too, we carried out experiments with both versions of the BDH algorithm. So far we discussed the ones with the first one (i.e., using \( m = \frac{3}{2} \alpha_{desired} - 1 \) in the implementation). Figure 4.16 shows \( \alpha_{achieved} \) (obtained by both the variants of BDH) vs \( \alpha_{desired} \) for the noisy 11-class data set. From this plot, we can draw the following conclusions about the performance of BDH algorithm on "forbidden" data which are exactly same as those obtained from the experiments on the noisy 8-class data set:

- BDH results in maps with \( \alpha_{achieved} \approx 1 \) when applied with \( \alpha_{desired} \) anywhere between \( \sim 0.6 \) and \( \sim 1.1 \).

- When applied with \( \alpha_{desired} < 0 \), negative magnification is achieved. The rare classes in the input data are enhanced in the SOM improving their detectability.

- There is no definitive trend in the performance of BDH when used with values of \( \alpha_{desired} \) outside of the regions discussed in the above two points.

Maps with \( \alpha_{achieved} \approx 1 \) are equiprobabilistic maps that represent the input space most faithfully. Rare classes in the input data are enhanced in the SOM when negative magnification is achieved (\( \alpha_{achieved} < 0 \)). Naturally the representation of the non-rare classes in such a map is somewhat distorted. So, the SOM obtained with \( \alpha_{achieved} < 0 \) and the one obtained with \( \alpha_{achieved} \approx 1 \) together provide a more complete picture of the clustering of a data set.
4.3 Conclusions on magnification control for high-dimensional data

Main conclusions from the experiments on high-dimensional data:

- BDH results in maps with $\alpha_{\text{achieved}} \approx 1$ when applied with $\alpha_{\text{desired}}$ anywhere between $\sim 0.6$ and $\sim 1.1$.

- When applied with $\alpha_{\text{desired}} < 0$, negative magnification is achieved. The rare classes in the input data are enhanced in the SOM improving their detectability.

- From our current simulations we don’t see a definitive trend in the performance of BDH when used with values of $\alpha_{\text{desired}}$ outside of the regions discussed in the above two points.

- We also carried out simulations with the two-step procedure for inducing certain $\alpha_{\text{desired}}$ on the two noisy data sets. From our experiments, no consistent trend in improvement over the one-step procedure was observed.

- Results of the experiments on the two noisy data sets using calculated $\alpha_{KSO M}$ instead of $\alpha_{KSO M} = 2/3$ were not much different. Both sets of results identified the same two regions of BDH applicability.

Therefore, from our current analysis we can identify two regions of the values of $\alpha_{\text{desired}}$ where the BDH algorithm can be made use of for practical applications.
1. If $0.6 \lesssim \alpha_{desired} \lesssim 1.1$, $\alpha_{achieved} \approx 1$. This means that by applying BDH algorithm with $0.6 \lesssim \alpha_{desired} \lesssim 1.1$, we can achieve an equiprobabilistic mapping of the input space (i.e., $\alpha_{achieved} \approx 1$). Such an SOM is the best (most faithful) representation of the input space.

2. If $\alpha_{desired} < 0$, negative magnification is achieved. Rare classes, that may otherwise find little or no representation in the map on using KSOM or Conscience algorithm, are enhanced in the SOM by using BDH with $\alpha_{desired} < 0$ thereby improving their detectability.
Chapter 5
Magnification control for real, high-dimensional data (multi-spectral imagery)

In the previous chapter we saw that SOMs with negative magnification are achieved when BDH is applied with $\alpha_{desired} < 0$. Rare classes of the input data set are enhanced in such an SOM, thereby improving their detectability. We also observed that these rare classes may not find any representation in the equiprobabilistic SOM resulting from Conscience algorithm. Detection of rare classes is a valuable thing for real data sets as this can lead to discovery of small, interesting groups. Encouraged with the observations on the synthetic (but fairly general) data sets in Chapter 4, we carried out simulations on a real high-dimensional data set - the multi-spectral image of Ocean City. In this chapter we discuss the results of applying negative magnification and compare them with those obtained by applying Conscience. We published these results in a recent paper [33].

5.1 Description of the Ocean City multi-spectral image

The Ocean City data set is a remotely sensed spectral image of Ocean City, Maryland*. A spectral image consists of $n$ co-registered image bands, each of which is taken at a different wavelength. Therefore, an $n$-dimensional vector is associated

---

*The image was provided by Dr. Beatrix Csathó, Byrd Polar Institute, The Ohio State University.
with each pixel of such an image (Figure 5.1). This vector referred is a spectral signature that carries compositional information about the material in the respective pixel. Spectral images are powerful information sources and are used in many areas of scientific research, business, industry, defense systems, etc. There is a serious need for effective, precise and detailed knowledge discovery (clustering) for such complex, high-dimensional data. Detection and discovery of rare classes is an important aspect of spectral image data analysis. We used a $256 \times 256$ pixel subset of the Ocean City image for our experiments on real data. From the original 12 spectral bands we selected 8 highest quality bands and therefore the input vectors are 8-dimensional. High pairwise correlation values, lying mostly between 0.5 and 0.95, and high-dimensionality render this a “forbidden” data set.

Figure 5.1  Concept of a spectral image. Figure from [14].
5.2 Simulation results

We applied BDH with $\alpha_{\text{desired}} = -0.8$ on the $256 \times 256$ subset of the Ocean City spectral image. Conscience was applied on the entire data set - the $512 \times 512$

Figure 5.2  Comparison of supervised classification and BDH clustering with $\alpha_{\text{desired}} < 0$. Left: An earlier supervised classification. A gray spot (the color of the background, 'bg') outlined by a small black rectangle within the framed top right quadrant is an unclassified region. Right: Result of SOM clustering using BDH magnification control with $\alpha_{\text{desired}} = -0.8$ on the top right quadrant of the image. The agreement between the supervised class map and this cluster map is striking, which inspires confidence in the clustering. Secondly, the spot that remained unclassified in the supervised map is now filled exactly and with a color (greenish-yellow - class a) that is different from all previous class colors and its spectral signature is distinct (Figure 5.3). We discovered a new class. Moreover, this cluster only occurs at this location, and nowhere else: we discovered a small rare class! Other small classes V and C are marked by ovals and dashed rectangles in the two images.

Ocean City image. A $40 \times 40$ SOM was used was used in both cases. Because of the high-dimensionality and unknown pdf of this data set, it was not possible to calculate the value of $\alpha_{\text{achieved}}$. Instead, we compared the areas occupied by the known rare classes in the SOMs obtained by BDH and Conscience to assess the effect
of negative magnification. In addition, we also looked for previously unidentified clusters. Figure 5.2 (Left) shows the supervised class map for the Ocean City image.

![Graphs showing mean spectral signatures of classes](image)

**Figure 5.3** Mean spectral signatures of the classes shown in Figure 5.2. Those of the small classes C, V, and a have been plotted in red. a, the newly discovered class has a spectral signature that is distinct from all the other clusters.

On the right is the $256 \times 256$ spectral image subset (upper right quadrangle of the entire $512 \times 512$ image) clustered by BDH with negative $\alpha_{desired}$. This is an image of an urban region of Ocean City, so one can make out the roads (purple color), rows of houses (this is an ariel image, so what we see of the houses are the roofs)
(yellow, white, pink and similar colored rectangular patches on the sides of the road),
waterways (greenish blue and blue colored) etc. For the purpose of comparison, we
will look at three small classes, $V$ (pale aqua), $C$ (white) and $a$ (yellowish green) in
detail. The mean signatures of the various classes are plotted in Figure 5.3. The two
SOMs have been compared in Figure 5.4.

![SOM obtained by BDH (with $\alpha_{dest} = -0.8$)](image)

![SOM obtained by Conscience](image)

**Figure 5.4** Comparisons of the SOMs developed by BDH vs Conscience learning. **Left:** The
SOM obtained by the application of BDH (with $\alpha_{dest} < 0$) on 256 × 256 subimage of the entire
Ocean City image. **Right:** SOM obtained by Conscience learning using the entire Ocean City
image. As discussed in the text, the rare clusters are magnified in the BDH SOM in comparison to
the Conscience SOM.

### 5.2.1 Class $V$

This small cluster was known at the time of an earlier supervised classification,
but was more definitely outlined by BDH clustering. In the images in Figure 5.2, $V$
(pale aqua) has been outlined by ovals. Figure 5.4 compares the representation of $V$ in the SOMs obtained by BDH and Conscience. $V$ is represented by 2 PEs in the Conscience SOM as compared to 6 PEs in the BDH SOM, even though the Conscience SOM was learned with 4 times as many data points, including a few more occurrences of the class $V$ in the large image outside the upper quadrangle (Figure 5.2). Also, there are sharper fences separating $V$ in the BDH SOM as compared to Conscience SOM, thereby improving its detectability.

5.2.2 Class $a$

For comparing BDH and Conscience, we also looked for previously unidentified clusters and discovered a previously unclassified cluster $a$ outlined by a solid rectangle in the images in Figure 5.2. In the supervised class map on the left, it is colored by the background color, ‘bg’, indicating that this region is an unclassified one. On the BDH cluster map this spot is completely filled with a single (greenish-yellow) color which is different from that of all other clusters. It can also be seen from Figure 5.2, that this cluster occurs only at this small spot, $a$ is a small rare class! Its spectral signature is distinct from that of all other clusters (Figure 5.3). Figure 5.4 shows the SOM view of this discovery. The greenish-yellow cluster was not clearly delineated in Conscience SOM and was “discovered” only because we looked for it based on the BDH discovery. This rare class covers only 3 PEs in the Conscience SOM in contrast to 7 PEs in the BDH SOM where it is also contoured by better developed fences.
5.2.3 Class $C$

Small cluster $C$ (white) also corresponds to a previously known class from the supervised class map. This white class is indicated by dashed squares in the images in Figure 5.2. In Figure 5.4, class $C$ is marked in the SOMs obtained by BDH (Left) and Conscience (Right). This class occupies 4 PEs in both SOMs, in spite that within the 1/4 subimage used for BDH clustering the white class occurs only in a small dashed square at the upper right corner, while there are many more white class pixels in the entire image used for the Conscience SOM training (most notably the long vertical rectangle in the lower right image corner which is outlined by a dashed rectangle).

5.3 Conclusion

Our observations about the three rare classes demonstrate that the rare classes in the data were magnified in the SOM obtained by BDH with $\alpha_{desired} < 0$, in comparison to Conscience SOM. So once again, BDH performed negative magnification on “forbidden” data strongly indicating its potential for detecting and discovering rare, interesting classes in complex, high-dimensional data.
Chapter 6
Automatic clustering of the SOM weights

After the SOM has converged, clusters in the SOM need to be identified. Several approaches including different visualization methods, semi-automatic techniques etc. are used for grouping the weights of the SOM into clusters. We developed a technique to identify the clusters in the SOM in a fully automated way. In this chapter, we first discuss some existing approaches for identifying the clusters in the SOM, after that we describe the automatic technique we developed and present results of applying it on some test data sets.

6.1 Background

One of the most common approaches in identifying clusters in the SOM is by using gray level expression to display the distances between the weights of the neighboring units. Cluster detection is then usually done using human eye [34] [35] [36]. In [34], a “U-matrix” (unified distance matrix) representation of the Self-Organizing Map visualizes the distances between the weights of the adjacent PEs by displaying them with different colorings between the PEs. A dark coloring between the PEs corresponds to a large distance and thus a gap between the weight values in the input space. A light coloring between the PEs signifies that the weight vectors are close to each other in the input space. Light areas can be thought as clusters and dark areas
as cluster separators. If the SOM produces well-structured clusters, the U-matrix visualization technique can be used to examine the contours of the clusters. However, this qualitative approach may not be good enough because there is a possibility of a misinterpretation of the gray level expression and also because clusters may not always be well separated.

If the boundaries of the cluster are not clear, statistical techniques can be applied to derive the cluster boundaries. If the number of clusters in the data set are known, a statistical cluster analysis algorithm like the k-means algorithm can be used. If the number of clusters in unknown, which is mostly the case, it needs to be estimated. In [37], an approximation technique was used to find the most ‘stable’ clustering result found from the adaptive k-means algorithm with dynamic initialization (a-dK) method. The most ‘stable’ result referred to the most frequent number of clusters returned by the a-dK method over a range of ascending values. After the number of clusters was found, this information was used to determine the clusters of similar data items. However, a cautionary note in this paper says that this value is only an approximation and should not be taken as an absolute truth.

In [38], segmentation of the U-matrix [34] was used as the method for determining the clusters. A mathematical morphology segmentation method watershed has been used to segment the U-matrix image after filtering it (filtering is required since the U-matrix image is a noisy image). This technique using watershed segmentation looks
quite promising. The drawback is that we want more than just segmentation. When very high dimensional data (~ 200) is mapped onto a 2-dimensional SOM, topology preservation may not be ideal. In that case, it is possible that weights corresponding to the same cluster are associated with PEs that are not nearby in the SOM grid. In such a situation, segmentation would not be a useful strategy to identify the clusters. So, an algorithm that can identify the clusters based on the weight values associated with the PEs and without using any information based on their relative locations in the SOM grid is needed. Another drawback in the watershed scheme is the lack of utilization of the density information (the number of input samples mapped to each PE).

Another approach ([39]) is to identify the clusters in a developed SOM based solely on the density information. They conjectured that as a result of lateral competition, the grouping or clustering relations in the original pattern set will be preserved by SOM mapping and will be shown on the SOM density map. Patterns that are closer to each other in the original space will ‘crowd’ their representatives in some place on the map, and because the weights of the PEs between two or more representative crowded areas are influenced by both of the adjacent clusters, they will tend to respond to none of them. On the SOM density map, this will be reflected as ‘plateaus’ separated by ‘valleys’ on the map. The plateaus correspond to the clusters in the data set, and the valleys correspond to cluster boundaries. This method, however, may not
work well enough with real data because oftentimes, clusters are not tight enough to separate well. Moreover, the signatures belonging to different clusters may be very similar and even the PEs on the boundaries of the two clusters may have sufficient hits, thereby blurring the boundaries. In such a case, weight values will need to be used in conjunction with the density information. [40] describes a dendrogram based technique that utilizes the weight values alone for identifying clusters in the SOM. Several different within- and between-cluster distance measures were used. This method is advantageous as it does not assume topology preserving mapping, does not require the knowledge or estimation of the number of clusters and uses weight values for identifying clusters in the SOM. The drawbacks are that the distance measures used have a bias towards grouping weights into spherical clusters while in real data clusters can be of any shape. Also some assumptions about the data distribution that were used in the algorithm are not always justified. This technique does not allow the existence of singleton clusters. Density information contained in the SOM was not used at all. Both pieces of information available from the map, weight values and density information, should be combined to identify clusters. We propose a technique that is based on the same skeletal structure as that in [40], the phases are the same but the criteria used at every step are different. Our technique makes use of both the weight values and the density information available from the map, does not require the knowledge or estimation of the number of clusters, detects clusters of all shapes
(has no bias towards spherical shaped clusters) and does not assume that the SOM has topology preserving mapping.

### 6.2 Technique developed by us

This is a fully-automated, dendrogram based algorithm. The inputs will be clustered to the same group as the PE to which they are mapped. Of all the weights, those with non-zero number of inputs mapped to them will be clustered because our final goal the clustering of the inputs and only weights with non-zero number of inputs mapped to them aid in this procedure. To provide some flexibility to the user a particular constant can be controlled, but that is optional. Distance measures used in our technique and in [40] are listed in table 6.1.

<table>
<thead>
<tr>
<th>Within-cluster distance</th>
<th>$S(Q_k)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>max nearest neighbor distance</td>
<td>$S_{nn(max)}(Q_k) = \max_i \min_i'{</td>
</tr>
<tr>
<td>nearest neighbor distance</td>
<td>$S_{nn}(Q_k) = \frac{\sum_i \min_i'{</td>
</tr>
<tr>
<td>centroid distance</td>
<td>$S_c(Q_k) = \frac{\sum_i</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Between-cluster distance</th>
<th>$d(Q_k, Q_l)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>single linkage</td>
<td>$d_s(Q_k, Q_l) = \min_{i,j}{</td>
</tr>
<tr>
<td>centroid linkage</td>
<td>$d_{ce}(Q_k, Q_l) =</td>
</tr>
</tbody>
</table>
As mentioned earlier, our technique is based on the same skeletal structure as that in [40], the phases are the same but the criteria used at every phase are different. Next, we will discuss the various phases of our technique. For each of the phases, we will also discuss the procedure used in [40] for that phase and compare it with our approach. The proposed technique consists of four phases:

**Phase I: Building the dendrogram** The first step in the clustering process is building a dendrogram. Initially, each weight vector is a single element cluster. At every step, we find pairwise distances between all the clusters and then merge the two clusters that are closest to each other. This is done until there is only one cluster left. The distance measure we used in this step is the single link distance [40] described in Table 6.1. In [40], the distance measure used in this phase was the centroid linkage, i.e., the centroidal distance between the clusters. This between-cluster distance measure, centroid linkage, is justified only if the clusters are spherical, which is mostly not the case. The procedure for building the dendrogram:

1. Initialization: assign each weight vector to its own cluster.
2. Compute pairwise distances between all clusters.
3. Merge the two clusters that are closest to each other.
4. Return to step 2 until there is only one cluster left.
So, in this step, weight vectors are merged together to form a clustering dendrogram (tree) that finally consists of a single cluster comprising all the weight vectors.

**An example dendrogram (result of Phase I)**

<table>
<thead>
<tr>
<th>level</th>
<th>level +1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

- Leaf nodes (weights)
- Nodes of the dendrogram

Total number of weights to be clustered, $N = 5$

**Figure 6.1** A simple example dendrogram.

**Phase II: Determining interesting merges** Every node in the dendrogram represents a cluster where two other clusters merge (Figure 6.1). In this step, we label each node as an “interesting” (label = 1) or “uninteresting” (label = 0) merge. Let us consider a node (cluster) $C$ where two clusters $A$ and $B$ merge. If $C$ is not an “interesting” merge, clusters $A$ and $B$ are “interesting” clusters. Now we are to decide whether $C$ is an “interesting” merge or not. Our criterion for deciding this is based on the following property of clustering:

*Any point in a cluster is closer to some point in the same cluster than to any point in any other cluster and vice-versa.*
For \( k, k' = \arg \max_i \min_j ||a_i - a_j|| \), \( a_i, a_j \in A, i = 1, 2, \ldots, A_n \) (where \( A_n \) is the number of weight vectors in \( A \)),

\[
\text{if } ||a_k - a_{k'}|| > ||a_i - b_j|| \text{ for some } b_j \in B
\]

then \( ||a_i - a_{k'}|| > ||a_i - b_j|| \) \( (6.1) \)

Equation 6.1 means that the point \( a_i \in A \) is closer to a point in \( B \) than to any point in \( A \). This contradictory to our condition of clustering, so, if \( a_i \in A \), then \( b_j \) should belong to \( A \) too, and so should all the other points in \( B \). In other words, the merge of \( A \) and \( B \) is “interesting”. This condition can be summarized in terms of within- and between-cluster distances as

\[
\text{if } \left( \frac{\max(\mathcal{S}_{nn}(A), \mathcal{S}_{nn}(B))}{d_s(A, B)} > 1 \right)
\]

then \( C \) is an “interesting” merge. \( (6.2) \)

By the end of this phase, we have labelled each node in the tree as as “interesting” or “uninteresting” merge. But we still do not have the final clusters. The final partitioning will be carried out in the next phase where we will also modify the criterion for determining interesting merges to enable the partitioning to work properly.

The criterion used in [40] for deciding whether a node \( C \) is an “interesting”
merge or not was:

\[
\begin{align*}
& \text{if } d_s(A, B) < S_{nn}(A) + S_{nn}(B) \text{ or} \\
& \text{if } d_{ee}(A, B) < S_c(A) + S_c(B) \\
\end{align*}
\]

(6.3)

then \( C \) is an "interesting" merge.

The above criterion is heuristic in nature. The clusters derived by using the above criterion will not conform to the fundamental property of clustering, namely \textit{Any point in a cluster is closer to some point in the same cluster than to any point in any other cluster and vice-versa.}

The following assumptions were made in [40]:

- All singleton clusters were ignored to evaluate the criterion in Equation 6.3.

- The distances between samples and their nearest neighbors were assumed to be gamma distributed without justification.

- In order to make the between-cluster distance \( d_s(Q_k, Q_l) \) more robust, the mean of four shortest distances between the samples was used instead of only the shortest distance.

No theoretical or empirical justification was provided for the above mentioned assumptions.

\textbf{Phase III: Partitioning} In this step, the data clusters will be decided. We start from the top of the dendrogram and visit every node and see if it is an interesting
merge. If it is, this node is a potential cluster. Such a potential cluster (parent cluster) typically consists of subclusters (a subcluster is a node which is an "interesting" merge and so has label = 1) and loose leaves (Figure 6.2). The decision to be made at this stage is whether the parent cluster is to be kept as a cluster or not. If this parent cluster is not kept as a cluster, it is dissociated into subclusters and leaves (which are singleton clusters). The subclusters are then examined to see if they will be kept as clusters. This process is carried out recursively.

Let

- $n_c$ be the number of leaves belonging to the subclusters.
- $n_l$ be the number of loose leaves.
- $n_{c,\text{inputs}}$ be the total number of inputs mapped to the weights belonging to the subclusters.
- $n_{l,\text{inputs}}$ be the total number of inputs mapped to the loose leaves.

The parent cluster is kept as a final cluster if the following criterion is met

$$\frac{n_{c,\text{inputs}}}{(n_{c,\text{inputs}} + n_{l,\text{inputs}})} < 1/2 \quad (6.4)$$

An example in Figure 6.2 illustrates this. The parent cluster $C$ contains of two subclusters $A$ and $B$. If the condition in Equation 6.4 is satisfied, $C$ is a cluster. Otherwise, $C$ will not be kept as a cluster and $A$ and $B$ will be examined to see if they can be kept as clusters.
In [40], the parent cluster is kept as a final cluster if the following criterion is met

$$\text{If } \frac{n_c}{(n_c + n_l)} < 1/2$$  \hspace{1cm} (6.5)

So whereas our criterion in Equation 6.4 uses the density information available from the SOM to carry out the final partitioning, the criterion used in [40], does not make use of the density information. The density information is an important piece of information available from the SOM and not making use of that to carry out the grouping of weights will give only sub-optimal results.

In Equation 6.3, we explained the criterion used in [40], for deciding whether a node is an “interesting” merge or not. It was not geared towards making the partition phase work properly. Next, we discuss the modification we made to
our criterion for labelling a node as an "interesting" merge or not (Equation 6.2 from Phase II), to make partition work well.

If there are too many "interesting" merges and not enough loose leaves in the lower regions of the tree (higher values of level in Figure 6.1), the potential clusters at the upper regions of the tree will not be kept as clusters and because of the recursive nature of the process, there is a possibility of ending up with too many small clusters. So, in order for the partitioning to function properly, it is required that there are more "interesting" merges in the upper regions of the tree (lower values of level) and less interesting "interesting" merges in the lower regions of the tree (higher values of level). This can be ensured by introducing a suitable threshold in the condition for determining interesting merges

\[
\text{if } \left( \frac{\max(S_{nn}(max)(A), S_{nn}(max)(B))}{d_s(A, B)} > \text{threshold} \right) \text{ then } C \text{ is an "interesting" merge.} \tag{6.6}
\]

If the threshold in Equation 6.6 is made high, the possibility of the merge being an "interesting" merge becomes less. Since we want it to be less possible for higher values of level and more possible for smaller values of level, we make the threshold proportional to \((level + 1)\). So,

\[
\text{threshold} = \text{constant} \ast (level + 1).
\]

Now, the maximum possible value of \((level + 1)\) where we will be making a
decision whether it is an “interesting” merge or not is \((N - 1)\), where \(N\) is the total number of weight vectors being clustered. In order to make the threshold 1 at this level, the constant should have a value of \(\frac{1}{(N-1)}\). This way we can ensure that the possibility of a node being an “interesting” merge is increased as we go up the tree (i.e., with lower values of level) and yet the even the lowest nodes in tree have a fair chance of being labelled as “interesting” merges. So the condition for determining interesting merges is

\[
\text{if } \left( \frac{\max(S_{nn(max)}(A), S_{nn(max)}(B))}{d_s(A, B)} > \frac{1}{N-1} \right) \Rightarrow (\text{level} + 1) \\
\text{then } C \text{ is an “interesting” merge.} 
\]

(6.7)

**Phase IV: Merging mergeable clusters** In this phase, the clusters obtained in the partitioning step are further examined to see if any of them are mergeable and if so they are merged.

In [40], in this last phase, all singleton clusters were merged with their respective nearest non-singleton clusters (the distance measure used is centroid linkage \(d_{ce}\)). This is not justified because it prevents the existence of singleton clusters though there might be some in the data. No attempt was made for merging two non-singleton clusters which might actually be mergeable. Also, once again, this kind of an approach with the distance measure being \(d_{ce}\) is only justified if the clusters are spherical.
Our merging procedure does not assume the non-existence of singleton clusters. We also developed a criterion for checking if two non-singleton clusters are mergeable. Also, our procedure does not require the assumption that the clusters be spherical.

Let us denote the clusters at this stage by $Q_i$, $i = 1, 2, \ldots, N_f$. Where $N_f$ is the total number of clusters at this point. The merging process takes place as follows:

Go through the list of clusters and for every cluster $Q_i$,

until there exists no cluster $Q_j$ mergeable to $Q_i$

\[
\begin{align*}
&\text{find } Q_j \text{ mergeable to } Q_i \\
&\text{merge } Q_j \text{ to } Q_i
\end{align*}
\]

(1)

The task marked as (1), i.e., finding a cluster $Q_j$ mergeable to $Q_i$, is carried out as follows:

**Case I:** if $Q_i$ is not a singleton cluster

**Part I:**
if $Q_i$ is not a singleton cluster
{
    go through the list of clusters and for every cluster $Q_j$ ($j \neq i$)
    {
        if $Q_j$ is a non-singleton cluster
            {
                if $d_s(Q_i, Q_j) < \max(S_{nn\text{max}}(Q_i), S_{nn\text{max}}(Q_j))$
                    $Q_j$ is mergeable to $Q_i$. Return $Q_j$ to be merged with $Q_i$ at (1).
                else if($Q_j$ is a singleton cluster)
                    {
                        Let $n_w(Q_i)$ be the total number of weight vectors in the
                        non-singleton cluster $Q_i$.
                        if $d_s(Q_i, Q_j) \leq (1 + \frac{1}{n_w(Q_i)})S_{nn\text{max}}(Q_i)$
                        {
                            Return $Q_j$ to be merged with $Q_i$ at 1. (2)
                        }
                    }
            }
    }
}

Justification for (2): Here, we are trying to check if a singleton cluster $Q_j$ is
mergeable to a non-singleton cluster $Q_i$. $n_w(Q_i)$ is the total number of weight
vectors in the non-singleton cluster $Q_i$. Now, the possibility of merging the
singleton cluster $Q_j$ to $Q_i$, $P(\text{merge}(Q_i, Q_j))$ is

$$P(\text{merge}(Q_i, Q_j)) \propto n_w(Q_i)$$ (6.8)

For the merge of the singleton cluster $Q_j$ to $Q_i$ to be justified, either

$$\frac{d_s(Q_i, Q_j)}{S_{nn\text{max}}(Q_i)} \leq 1$$

must hold or if this ratio is somewhat greater than 1, it should be as close to 1
as possible. In other words, the possibility of merging the singleton cluster $Q_j$
to \( Q_i \), \( P(\text{merge}(Q_i, Q_j)) \) satisfies

\[
P(\text{merge}(Q_i, Q_j)) \propto \frac{1}{\left( \frac{d_s(Q_i, Q_j)}{S_{nn(\text{max})}(Q_i)} - 1 \right)}
\]  \hspace{1cm} (6.10)

Equations 6.8 and 6.10 give,

\[
\left( \frac{d_s(Q_i, Q_j)}{S_{nn(\text{max})}(Q_i)} - 1 \right) \propto \frac{1}{n_w(Q_i)}
\]  \hspace{1cm} (6.11)

Using Equations 6.10 and 6.11, we get

if \( \left( d_s(Q_i, Q_j) \leq \left( \frac{K}{n_w(Q_i)} + 1 \right) S_{nn(\text{max})}(Q_i) \right) \)

then merge the non-singleton cluster \( Q_i \) and the singleton cluster \( Q_j \). Here, \( K \) is a constant that can be changed by the user thus providing some flexibility.

If this flexibility is not desired, the default value for \( K \) is 1.

Part II:

if \( Q_i \) is not a singleton cluster and no mergeable cluster was found in Part I

\[
\{ \\
\text{find the cluster } Q_{j,min} \text{ which is closest to } Q_i, i.e., } d_s(Q_i, Q_{j,min}) = \min_k d_s(Q_i, Q_k) \\
\text{if } Q_{j,min} \text{ is a singleton cluster} \\
\{ \\
\text{find the cluster } Q_{k,min} \text{ closest to } Q_{j,min}, i.e., } k.min = \arg \min_k d_s(Q_{j,min}, Q_k) \\
\text{if } Q_{k,min} \text{ is the same cluster as } Q_i \\
\{ \\
\text{ } Q_{j,min} \text{ is mergeable to } Q_i. \text{ Return } Q_{j,min} \text{ to be merged with } Q_i \text{ at } (1). \\
\text{This means that if the non-singleton cluster } Q_i \text{ and the singleton cluster } Q_{j,min} \text{ are mutually closest to each other, then } Q_{j,min} \text{ should be} \\
m\text{merged to } Q_i. \\
\}
\}
\]

In Part II here, since the probability of existence of singleton clusters is small,
we facilitate the merging of singleton clusters to non-singleton clusters thereby reducing the number of singleton clusters.

**Case II:** If $Q_i$ is a singleton cluster, we look for a singleton cluster which is mergeable to $Q_i$.

if $Q_i$ is a singleton cluster
{
    find the cluster $Q_{j.min}$ which is closest to $Q_i$, i.e., $j.min = \arg \min_k d_s(Q_i, Q_k)$
    if $Q_{j.min}$ is a singleton cluster
    {
        if $Q_{j.min}$ and $Q_i$ have the same parent in the dendrogram constructed in Phase I
        {
            $Q_{j.min}$ is mergeable to $Q_i$. Return $Q_{j.min}$ to be merged with $Q_i$ at Step (1). (If two singleton clusters have the same parent in the dendrogram, it means that they are very close in the input space and so it is justified to merge them).
        }
    }
}

The main advantages of our technique are

**Fully automated** This is a fully-automated technique in contrast to a lot of other techniques available which are only semi-automatic, based on human judgement or require some other parameters like the number of clusters etc.

**Allows singleton clusters** This technique allows the existence of single weight vector clusters if the weight vector is sufficiently different from other clusters. This is important because a rare class in the input data set might be represented by a single PE in the SOM.
No bias towards spherical clusters The various distance measures and criterion
used have no bias towards grouping weights into spherical clusters only.

Allows some flexibility for the user There is one constant, $K$ in the algorithm
that can be set by the user to alter the results to some extent.

This clustering algorithm was applied on some data sets and the clustering ob-
tained was compared with the known clustering to assess the performance of the
algorithm. In the next section, we discuss those simulations and the results obtained.

6.3 Simulation results

In this section we will discuss the results of applying our technique to four different
data sets. The input to the algorithm is the attributes of the SOM (obtained by any
Self-Organizing Map algorithm like KSOM, BDH or Conscience): the weights and
the number of inputs mapped to each weight.

6.3.1 Data Set I

This data set consisted of 1000 2-dimensional vectors $v = (v_1, v_2)$, Gaussian dis-
btributed around four different means (Type 1: $E[v] = (7, 0)$ (280 samples), Type 2:
$E[v] = (7, 7)$ (250 samples), Type 3: $E[v] = (0, 7)$ (150 samples), Type 4: $E[v] = (0, 0)$
(320 samples). All have the same variance: $E[(v_1 - E[v_1])^2] = 0.1, E[(v_2 - E[v_2])^2 = 0.1]$. 
So this data set consists of 4 different classes. The weights vectors $w = (w_1, w_2)$ re-
sulting from applying KSOM on Data Set I are plotted in the 2-dimensional input
Figure 6.3  **Left:** The (100) weight vectors resulting from applying KSOM on Data Set I plotted in the 2-dimensional input space. Only those weights which had non-zero number of inputs mapped to them were used for clustering. **Right:** Weight vectors and number of inputs mapped to each PE (the maximum here is 26) in the SOM obtained by applying KSOM to Data Set I.

space in Figure 6.3 (Left) A 10 × 10 grid of PEs was used, so there are a 100 weight vectors altogether. The two pieces of information, weight vector associated and the number of inputs mapped to each PE used for clustering the weights are depicted in Figure 6.3 (Right). The weights with non-zero number of inputs mapped to them were used for clustering. Clearly, there are 4 different kinds of weight vectors here. The result of applying the proposed clustering technique is depicted in Figure 6.4. The clustering technique separated the weights into 4 different clusters that are the same as the expected clusters.
Figure 6.4 The proposed clustering technique separated the weights corresponding to Data Set I into 4 different clusters that are the same as the expected clusters.

6.3.2 Data Set II

Data Set II is very similar to Data Set I, except that there are non-spherical classes in this data set. Also two of the classes in Data Set II have larger variances and are relatively closer to each other. There are 1000 2-dimensional vectors \( \mathbf{v} = (v_1, v_2) \), Gaussian distributed around four different means (Type 1: \( E[\mathbf{v}] = (7, 0) \), \( E[(v_1 - E[v_1])^2] = 0.5 \), \( E[v_2 - E[v_2]]^2 = 0.4 \) (280 samples), Type 2: \( E[\mathbf{v}] = (7, 4) \), \( E[(v_1 - E[v_1])^2] = 0.5 \), \( E[v_2 - E[v_2]]^2 = 0.1 \) (250 samples), Type 3: \( E[\mathbf{v}] = (0, 7) \), \( E[(v_1 - E[v_1])^2] = 0.1 \), \( E[v_2 - E[v_2]]^2 = 0.1 \) (150 samples), Type 4: \( E[\mathbf{v}] = (0, 0) \), \( E[(v_1 - E[v_1])^2] = 0.1 \), \( E[v_2 - E[v_2]]^2 = 0.1 \) (320 samples). So this data set consists of 4 different classes. The weight vectors \( \mathbf{w} = (w_1, w_2) \) resulting from applying KSOM on Data Set II are plotted in the 2-dimensional input space in Figure 6.5 (Left) A 10 \( \times \) 10 grid of PEs was used, so there are a 100 weight vectors altogether. The two pieces of information, weight vector associated and the number of inputs mapped to each PE
Figure 6.5  **Left:** The 100 weight vectors resulting from applying KSOM on Data Set II plotted in the 2-dimensional input space. Only those weights which had non-zero number of inputs mapped to them were used for clustering. **Right:** Weight vectors and number of inputs mapped to each PE (the maximum here is 31) in the SOM obtained by applying KSOM to Data Set II.

used for clustering the weights are depicted in Figure 6.5 (Right). The weights with non-zero number of inputs mapped to them were used for clustering. Clearly, there are 5 different kinds of weight vectors here. These weights are forming non-spherical clusters. The result of applying the proposed clustering technique is depicted in Figure 6.6. The clustering technique separated the weights into 5 different clusters which is what we expect by looking at the weights in the input space. A singleton cluster has been detected. It is quite far from the other clusters to be merged with any of them. Moreover, within-cluster distances of all the other clusters are smaller than between-cluster distance between this singleton cluster and any of the other clusters, so this singleton cluster should remain separate and not be merged with any of the other existing clusters.
6.3.3 Noisy 8-class data set

This 6-dimensional 8-class data set has been described in Chapter 4, Section 4.2.1. The weights obtained by applying BDH with \( \alpha_{\text{desired}} = 1 \) are depicted in Figure 6.7 (Left). Density information, i.e., the number of inputs mapped to each PE in the SOM is depicted in Figure 6.7 (Right). Once again, only those weights which had non-zero PEs mapped to them were used for clustering.

Clustering results obtained by applying the proposed technique are depicted in figures 6.8 (Left) and (Right). Using \( K = 1 \) results in the weights being clustered into 15 classes with small differences between the clusters. From figure 6.8 (Left), we can see that the relatively small "extra" clusters are at the boundaries of other large clusters in the SOM. This means that those were probably somewhere in between two different classes and hence couldn’t be merged with either of them. Using \( K = 4 \), the weights were separated into 8 clusters, which matched the expected clusters.
Figure 6.7  **Left:** Weights obtained by applying BDH with $\alpha_{\text{desired}} = 1$ on the 6-dimensional, noisy 8-class data set. A $20 \times 20$ SOM was used. **Right:** Height of the bar represents the number of inputs mapped to the corresponding PE in the $20 \times 20$ SOM.

Figure 6.8  Clustering of the weights of the SOM (corresponding to BDH with $\alpha_{\text{desired}} = 1$ on the 6-dimensional, noisy 8-class data set) by the proposed technique. Different clusters are represented by different colors. Weight vectors are plotted and PEs with no inputs mapped are marked as in Figure 6.7. **Left:** Using $K = 1$ resulted in 15 clusters. **Right:** Using $K = 4$ resulted in 8 clusters.
6.3.4 Noisy 11-class data set

This 6-dimensional 11-class data set has been described in Chapter 4, Section 4.2.1. The weights obtained by applying BDH with $a_{desired} = -0.8$ are depicted in Figure 6.9 (Left). Density information, i.e., the number of inputs mapped to each PE in the SOM is depicted in Figure 6.9 (Right). Once again, only those weights which had non-zero PEs mapped to them were used for clustering.

**Figure 6.9** Left: Weights obtained by applying BDH with $a_{desired} = -0.8$ on the 6-dimensional, noisy 11-class data set. A $20 \times 20$ SOM was used. Right: Height of the bar represents the number of inputs mapped to the corresponding PE in the $20 \times 20$ SOM.

Clustering results obtained by applying the proposed technique are depicted in figures 6.10 (Left) and (Right). Using $K = 1$ results in the weights being clustered into 17 classes with small differences between the clusters. From figure 6.10 (Left), we can see that the relatively small “extra” clusters are at the boundaries of other large
clusters in the SOM. This means that those were probably somewhere in between two different classes and hence couldn’t be merged with either of them. Using $K = 4$, the weights are separated out into 11 clusters, which match the expected clusters.

![Image of clustered SOM](image)

**Figure 6.10**  Clustering of the weights of the SOM (corresponding to BDH with $\alpha_{desired} = -0.8$ on the 6-dimensional, noisy 11-class data set) by the proposed technique. Different clusters are represented by different colors. Weight vectors are plotted and PEs with no inputs mapped are marked as in Figure 6.9. **Left:** Using $K = 1$, resulted in 17 clusters. **Right:** Using $K = 4$ resulted in 11 clusters.

From the clustering results described in this chapter, it is clear that the performance of the proposed technique is quite good. It detected non-spherical as well as singleton clusters in Data Set II. This fully automated technique that requires no additional parameters and uses only the information available from the SOM promises to eliminate any need for human judgement and so a possibility of misinterpretation of results. The constant $K$ can be optionally controlled by the user, the default value is 1.
Chapter 7
Conclusions

The Bauer, Der Hermann algorithm [13] (BDH), for magnification control in SOMs is theoretically guaranteed to work only for a very small family of "allowed" data. Most of the real data, however, falls into the category of "forbidden" data. In this thesis, we have investigated the performance of the BDH algorithm on "forbidden" data. After carrying out extensive analysis, we have been able to identify two regions of the values of the desired magnification factor, $\alpha_{\text{desired}}$, where the BDH algorithm can be used for practical applications.

1. If $0.6 \lesssim \alpha_{\text{desired}} \lesssim 1.1$, the actual magnification factor achieved by the SOM, $\alpha_{\text{achieved}}$, is $\approx 1$. This means that by applying the BDH algorithm with $0.6 \lesssim \alpha_{\text{desired}} \lesssim 1.1$, we can achieve an equiprobabilistic mapping of the input space (i.e., $\alpha_{\text{achieved}} \approx 1$) (Chapter 4). Such an SOM is the best (most faithful) representation of the input space.

2. If $\alpha_{\text{desired}} < 0$, negative magnification is achieved. Rare classes that otherwise find little or no representation in the map by using KSOM or Conscience algorithm, are enhanced in the SOM by using BDH with $\alpha_{\text{desired}} < 0$ thereby improving the detectability of these classes (Chapters 4, 5).
Maps with $\alpha_{\text{achieved}} \approx 1$ are equiprobabilistic maps which represent the input space most faithfully. Rare classes in the input data are enhanced in the SOM when negative magnification is achieved ($\alpha_{\text{achieved}} < 0$). Naturally the representation of the non-rare classes in such a map is somewhat distorted. So, the SOM obtained with $\alpha_{\text{achieved}} < 0$ and the one obtained with $\alpha_{\text{achieved}} \approx 1$ together provide a more complete picture of the clustering of a data set.

Our analysis was confirmed when we observed magnification of three rare classes in a SOM obtained by BDH with $\alpha_{\text{achieved}} < 0$ in comparison to Conscience SOM for the real multi-spectral Ocean City image (Chapter 5). This strongly indicates the BDH algorithm’s potential for detecting and discovering rare, interesting classes in complex, high-dimensional data.

As a final step in the SOM clustering process, after the SOM has converged, clusters in the SOM need to be identified. Many approaches including different visualization methods, semi-automatic techniques etc. are used in grouping the weights of the SOM into clusters. However, there does not exist any standard fully automated scheme that does this well. We developed a technique to do this in a fully automated way (Chapter 6). We obtained good results on applying our technique on several test data sets. This fully automated technique that is capable of detecting non-spherical as well as singleton clusters, requires no additional parameters and uses only the information available from the SOM, thereby mostly eliminating the need for human
judgement and hence significantly decreasing the possibility of misinterpretation of results. An optional constant can be controlled to provide some flexibility to the user.

As we know, Self-Organizing Map algorithms are powerful tools for effective, precise and detailed knowledge discovery (clustering) for complex, high-dimensional data as opposed to many conventional techniques that may fail to handle such data. By establishing the usable functionality of the BDH algorithm for magnification control we have contributed significantly to the area of clustering and knowledge discovery by Self-Organizing maps. We also hope that the results that we have obtained on also provide some feedback for enhancing the theory behind BDH for all kinds of data. Our second contribution, the fully automated technique for identifying clusters in the SOM enables the SOM clustering process to be carried out in an automated way and promises to eliminate the requirement of additional information and hence the possibility of misinterpretation due to human judgement.
Appendix A
On the evaluation of the magnification exponent of an SOM

A.1 Introduction

The following power law relates the density of weights in the input space $Q(w)$ to the pdf $P(w)$ of the input samples,

$$Q(w) = P(w)^\alpha \times \text{constant}$$  \hspace{1cm} (A.1)

$Q(w)$ being the density of weights in the input space, is the number of reference vectors in a small volume $dw$ of the input space. If both sides of Equation are divided by $N_W$, the total number of weight vectors, on the left will be $Q(w)/N_W$, the pdf of the weight vectors and on the right, it can be absorbed as a new constant such that $\text{new constant} = \text{constant}/N_W$. From now onwards we will use the same power law as in Equation A.1, but with the understanding that $Q(w)$ now denotes the pdf of the weight vectors.

From Equation A.1, it is clear that evaluation of the value of $\alpha$ ($\alpha_{\text{achieved}}$) requires an estimation of the two pdfs. A histogram based method can be used to estimate the pdfs and evaluate $\alpha$. 
A.2 A histogram based method for evaluation of $\alpha$

Let $p_i$ and $q_i$ be samples of the two densities $P(x)$ and $Q(w)$ respectively (Equation A.1). $p_i$ and $q_i$ can be obtained from the input samples and weights by partitioning the input space into $N_B$ bins and constructing frequency histograms in the following way:

$$ q_i = \left( \frac{n_{W_i}}{N_W} \right) / V_i, \quad i = 1, 2, \ldots, N_B $$

(A.2)

where $n_{W_i}$ is the number of weights in the the $i^{th}$ bin, $N_W$ is the total number of weights and $V_i$ is the volume of the $i^{th}$ bin. and

$$ p_i = \left( \frac{n_{I_i}}{N_I} \right) / V_i, \quad i = 1, 2, \ldots, N_B $$

(A.3)

where $n_{I_i}$ is the number of input samples in the the $i^{th}$ bin, $N_I$ is the total number of input samples and $V_i$ is the volume of the $i^{th}$ bin.

The value of $\alpha$ that best satisfies

$$ q_i = constant \times p_i^\alpha, \quad i = 1, 2, \ldots, N_B $$

(A.4)

in other terms, one that minimizes the error measure,

$$ E(\alpha) = \sum_{i=1}^{N_B} (q_i - constant \times p_i^\alpha)^2 $$

(A.5)

is the value of $\alpha_{achieved}$ by the SOM. $\alpha_{achieved}$ can be found by varying $\alpha$ in a range around the value of $\alpha_{desired}$, determine the corresponding value of the error $E(\alpha)$ and then choose the $\alpha_{achieved}$ as that value of $\alpha$ that minimizes $E(\alpha)$. 
The constant for a particular value of $\alpha$ can be determined by noting that $\sum_{i=1}^{N_B} q_i \ast (V_i) = 1$ ($q_i \ast (V_i)$ = probability that a weight lies in the $i^{th}$ bin, summing this over all bins is 1). So multiplying both sides of Equation A.4 by $V_i$ and then summing over $N_B$ bins gives,

$$\text{constant} = \frac{1}{\sum_{i=1}^{N_B} P_i}$$

This histogram based method for determining the value of $\alpha_{\text{achieved}}$ seems simple enough and easily extendible to higher dimensions, but actually there are some difficulties in doing that. The bin in a $d$-dimensional space is basically a hypercube. Now, in order to have an accurate estimation of the $pdf$ and hence that of $\alpha_{\text{achieved}}$, it is important to determine the correct size of the bin. As it turns out, this is not an easy task! In the following sections, we discuss selection of bin size and other issues involved in $pdf$ estimation using the analysis in [41].

### A.3 Determination of bin width in a 1-dimensional case

We start with the simplest case of 1-dimensional data. If a histogram is used to estimate the $pdf$ $f(x)$ ($x = x$, as the data is 1-dimensional). The estimated $pdf$ is given by

$$\hat{f}(x) = \frac{\nu_k}{nh} = \frac{1}{nh} \sum_{i=1}^{n} I_{[t_k,t_{k+1})}(x_i) \text{ for } x \in B_k$$

where $I_A(x) = 1$ if $x \in A$

$$= 0 \text{ if } x \notin A$$
$h$ is the bin width, $\nu_k$ is the number of samples in the interval $B_k = [t_k, t_{k+1})$ and $n = \sum_k \nu_k$. The analysis of the histogram variable $\hat{f}(x)$ is done by recognizing that the bin counts are Binomial random variables:

$$\nu_k \sim B(n, p_k), \quad \text{where} \quad p_k = \int_{B_k} f(t) \, dt$$

Then, the asymptotic mean integrated squared error (AMISE) in pdf estimation is:

$$AMISE(h) = \frac{1}{nh} + \frac{1}{12} h^2 R(f') \quad \text{where} \quad R(\phi) = \int \phi(x)^2 \, dx$$

On minimizing the above asymptotic $MISE$, we get the optimum bin width $h^*$ as:

$$h^* = \left[ \frac{6}{R(f')} \right]^{1/3} n^{-1/3}$$

$$AMISE^* = (3/4)^{2/3} R(f')^{1/3} n^{-2/3} \quad (A.7)$$

where $R(\phi) = \int \phi(x)^2 \, dx$

Thus the asymptotically optimal bin width depends on the unknown density. A rule of thumb is to use the bin width that results when $f = \mathcal{N}(\nu, \sigma^2)$

### A.3.1 Normal reference rule for 1-dimensional data

If we use the rule of thumb, $f = \mathcal{N}(\nu, \sigma^2)$, then $R(f') = 1/(4\sqrt{\pi} \sigma^3)$. Using this in Equation A.7,

$$h^* = (24\sqrt{\pi} \sigma^3 / n)^{1/3} \approx 3.5\sigma n^{-1/3} \quad (A.8)$$

So,

$$\text{Normal bin width reference rule :} \quad \hat{h} = 3.5\hat{\sigma} n^{-1/3} \quad (A.9)$$
A more robust rule, replacing the unknown scale parameter $\sigma$ by a multiple of the interquartile range (IQ)* is given by:

$$\hat{h} = 2(IQ)n^{-1/3}$$  \hspace{1cm} (A.10)

If the data has any skewness and some non-zero kurtosis, a factor corresponding to them can be used to correct the bin width further.

While simple ideas such as the Normal reference rule in Equation A.9 are useful, it would be better to use data-based procedures that approximately minimize the *mean integrated squared error* (MISE) and/or the *integrated squared error* (ISE). Next, some results incorporating this requirement are discussed.

From the expression for the minimizer of AMISE in Equation A.7, it is clear that any *lower bound* on $R(f')$ leads to an *upper bound* on the bin width. The simplest prior knowledge is that the density is zero outside an interval $(a,b)$. So, the optimization problem can be formulated as follows:

$$\min_f \int_{-\infty}^{\infty} f'(x)^2 dx \text{ such that support of } f = [-0.5, 0.5]$$  \hspace{1cm} (A.11)

On solving the above optimization problem, the following upper bound on the bin width is obtained:

$$h^* \leq \frac{b - a}{\sqrt{2n}}$$  \hspace{1cm} (A.12)

---

*Divide a set of data into two groups (high and low) of equal size at the statistical median if there is an even number of data points, or two groups consisting of points on either side of the statistical median itself plus the statistical median if there is an odd number of data points. Find the statistical medians of the low and high groups, denoting these first and third quartiles by $Q_1$ and $Q_3$. The interquartile range is then defined by $IQ = Q_3 - Q_1$ ([42])
Rearranging would give,

\[ \text{number of bins} \geq \sqrt[3]{2n} \]  \hspace{1cm} (A.13)

If we solve the optimization problem in Equation A.11 again but with fixed-range constraint replaced by the constraint that the variance of \( f \) is equal to \( \sigma^2 \), then the solution to this problem gives the following upper bound on the bin width:

\[ h^* \leq \left( \frac{686\sigma^3}{5\sqrt{7n}} \right)^{1/3} \approx 3.729\sigma n^{-1/3} \]  \hspace{1cm} (A.14)

The version based on interquantile range (IQ) is particularly robust:

\[ h^* \leq 2.603(IQ)n^{-1/3} \]  \hspace{1cm} (A.15)

Having understood the 1-dimensional case, let us now extend the analysis to the multidimensional case.

### A.4 The multivariate histogram

The case of multivariate histograms is somewhat more complicated than the 1-dimensional case. Given a sample from pdf \( f(x) \), where \( x \in \mathbb{R}^d \) is determined by a partition of space. Let us consider a regular partition by hyper-rectangles of size \( h_1 \times h_2 \times \ldots \times h_d \).

Now, if we consider a generic hyper-rectangular bin labelled \( B_k \) containing \( \nu_k \) points. As usual \( \Sigma_k \nu_k = n \). Then,

\[ \hat{f}(x) = \frac{\nu_k}{nh_1h_2\ldots h_d} \text{ for } x \in B_k \]  \hspace{1cm} (A.16)
By extending the 1-dimensional analysis suitably, the multivariate AMISE is obtained as follows:

\[ AMISE(h) = \frac{1}{nh_1h_2\ldots h_d} + \frac{1}{12} \sum_{i=1}^{d} h_i^2 R(f_i) \]  \hspace{1cm} (A.17)

On minimizing the above asymptotic MISE, we get the asymptotically optimal bin widths \( h_k^* \) as:

\[ h_k^* = R(f_k)^{-1/2} \left( 6 \prod_{i=1}^{d} R(f_i)^{1/2} \right)^{1/(2+d)} n^{-1/(2+d)} \]

\[ AMISE^* = \frac{1}{4} 6^{2/(2+d)} \left( \prod_{i=1}^{d} R(f_i) \right)^{1/(2+d)} n^{-2/(2+d)} \]  \hspace{1cm} (A.18)

A.4.1 Normal reference rule for multivariate data

Once again, we can consider the case of the normal distribution, i.e., \( X \sim N(\mu, \sum), \sum = \text{Diag}(\sigma_1^2, \sigma_2^2, \ldots, \sigma_d^2) \). Then, letting \( c_d = \sigma_1 \sigma_2 \ldots \sigma_d \), we have

\[ R(f_i) = \left( 2^{d+1} \pi^{d/2} \sigma_i^2 c_d \right)^{-1} \]  \hspace{1cm} (A.19)

\[ h_k^* = 2 \cdot 3^{1/(2+d)} \pi^{d/(4+2d)} \sigma_k n^{-1/(2+d)} \]

\[ AMISE^* = 2^{-(1+d)} 3^{2/(2+d)} \pi^{-d^2/(4+2d)} c_d^{-1} n^{-2/(2+d)} \]  \hspace{1cm} (A.20)

Now, the constant in the bandwidth increases slowly from 3.4908 in one dimension to the limiting value of \( 2\sqrt{\pi} = 3.5449 \) as \( d \to \infty \). Hence a very useful formula is:

**Normal reference rule**: \( h_k^* \approx 3.5 \sigma_k n^{-1/(2+d)} \)  \hspace{1cm} (A.21)
A.4.2 A special case: $d = 2$ with nonzero correlation

A special case to consider is the Normal bivariate case with non-zero correlation. In that case, $f(x_1, x_2) = \mathcal{N}(\mu_1, \mu_2, \sigma_1^2, \sigma_2^2, \rho)$. Then $R(f_1) = [8\pi(1 - \rho^2)^{3/2}\sigma_1^3\sigma_2^3]^{-1}$ and $R(f_2) = [8\pi(1 - \rho^2)^{3/2}\sigma_2^3\sigma_1^3]^{-1}$. Using Equation A.18, we get

\[
h_i^* = 3.504\sigma_i(1 - \rho^2)^{3/8}n^{-1/4} \\
AMISE^* = \frac{0.122}{\sigma_1\sigma_2}(1 - \rho^2)^{-3/4}n^{-1/2}
\] (A.22)

The effect of the correlation $\rho$ is to introduce powers of the quantity $(1 - \rho^2)$ into the equations. Thus, if the data are not independent but are clustering along a line, smaller bin widths are required to "track" this feature. If the density is degenerate (i.e., $\rho = \pm 1$), then the $MSE$ blows up. This result also indicates that if the data fall into any lower-dimensional manifold, a histogram will never be consistent!

A.5 Curse of dimensionality

The term curse of dimensionality to describe the exponential growth in combinatorial optimization as the dimension increases. Here, it is the number of bins that grows exponentially as the dimension increases. Let us see how histograms are affected by this phenomenon.

We shall use the multivariate Normal density with $\sum = I_d$ to draw some conclusions. From Table A.1, we can see that for the same number of samples, the error increases with dimensionality. Unfortunately, $MISE$ is not a dimensionless quantity
Table A.1  Example of asymptotically optimal bin widths and errors for $f = \mathcal{N}(0, I_d)$

<table>
<thead>
<tr>
<th>Dimension $d$</th>
<th>$h_d^*$</th>
<th>$AMISE_d^*$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$3.491n^{-1/3}$</td>
<td>$0.430n^{-2/3}$</td>
</tr>
<tr>
<td>2</td>
<td>$3.504n^{-1/4}$</td>
<td>$0.122n^{-2/4}$</td>
</tr>
<tr>
<td>3</td>
<td>$3.512n^{-1/5}$</td>
<td>$0.035n^{-2/5}$</td>
</tr>
<tr>
<td>4</td>
<td>$3.518n^{-1/6}$</td>
<td>$0.010n^{-2/6}$</td>
</tr>
</tbody>
</table>

and, hence, the error coefficients are not directly comparable.

So, we shall use the procedure described by Epanechnikov [43], to compare histogram errors and performance across dimensions. The following dimensionless rescaling of $MISE$ is used:

$$\epsilon_d = \frac{MISE}{R(f)} \quad \{ \approx 2^{-1}3^{2\frac{2}{d+2}}\pi^{\frac{d}{d+2}}n^{-\frac{2}{d+2}} \text{ when } f = \mathcal{N}(0, I_d) \} \quad (A.23)$$

For the Normal case, a table of equivalent sample sizes may be computed (Table A.2).

This table graphically illustrates the curse of dimensionality and the intuition that density estimation in more than four or five dimensions will not work. This prediction is the most pessimistic. If a Kernel Density Estimator is used, the number of samples required can be reduced somewhat. If 50 input samples are required in $\mathbb{R}^1$, then equivalent number of samples in $\mathbb{R}^8$ are around a million. The criterion in [44], $RCV(0)$ measures the amount of noise superimposed on the average value of the estimate at the multivariate origin. As a rule of thumb, $RCV < 1/3$, then the
Table A.2 Equivalent sample sizes across dimensions for the multivariate normal density, based on Epanechnikov's criterion

<table>
<thead>
<tr>
<th>Dimension $d$</th>
<th>Columns of Equivalent Sample Sizes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10 100 1,000</td>
</tr>
<tr>
<td>2</td>
<td>22 471 10,155</td>
</tr>
<tr>
<td>3</td>
<td>48 2,222 103,122</td>
</tr>
<tr>
<td>4</td>
<td>105 10,472 1,047,198</td>
</tr>
<tr>
<td>5</td>
<td>229 49,360 10,634,200</td>
</tr>
</tbody>
</table>

The estimate is unlikely to be totally overwhelmed by noise. If $30\% RCV(0)$ is acceptable, then the required number of input samples in $\mathbb{R}^8$ reduces from a million to 20,900. This prediction is by far the most optimistic. Instead of $L_2$ if the multivariate $L_1$ Kernel Error is used, then for all $x \in \mathbb{R}^d$, the asymptotically optimal $L_1$ and $L_2$ pointwise bandwidths satisfy

$$0.9635 \leq \frac{h_1^*(x)}{h_2^*(x)} \leq 1 \quad (A.24)$$

A.6 Conclusion

The number of input samples required for estimating the pdf for high-dimensional data is prohibitive. For the purpose of evaluating $\alpha_{achieved}$, both the input and weight pdfs need to be estimated. In a practical situation, we do not have enough sample points for either of them. As a consequence of this, the evaluation of $\alpha_{achieved}$ in a
general high-dimensional case is a difficult problem.
References


