A Local Network Neighbourhood Artificial Immune System

by
Alexander J. Graaff

Submitted in partial fulfillment of the requirements for the degree
Philosophiae Doctor
in the Faculty of Engineering, Built Environment and Information Technology
University of Pretoria
Pretoria, South Africa

June 3, 2011

A Local Network Neighbourhood Artificial Immune System

by

Alexander J. Graaff

Abstract

As information is becoming more available online and will forevermore be part of any business, the true value of the large amounts of stored data is in the discovery of hidden and unknown relations and connections or traits in the data. The acquisition of these hidden relations can influence strategic decisions which have an impact on the success of a business. Data clustering is one of many methods to partition data into different groups in such a way that data patterns within the same group share some common trait compared to patterns across different groups. This thesis proposes a new artificial immune model for the problem of data clustering. The new model is inspired by the network theory of immunology and differs from its network based predecessor models in its formation of artificial lymphocyte networks. The proposed model is first applied to data clustering problems in stationary environments. Two different techniques are then proposed which enhances the proposed artificial immune model to dynamically determine the number of clusters in a data set with minimal to no user interference. A technique to generate synthetic data sets for data clustering of non-stationary environments is then proposed. Lastly, the original proposed artificial immune model and the enhanced version to dynamically determine the number of clusters are then applied to generated synthetic non-stationary data clustering problems. The influence of the parameters on the clustering performance is investigated for all versions of the proposed artificial immune model and supported by empirical results and statistical hypothesis tests.

Keywords: Data Clustering, Artificial Lymphocytes, Affinity Maturation, Clonal Selection, Somatic Hyper Mutation, Artificial Immune Networks, Immune Network Topologies, Clustering Performance Measures, Dynamic Clustering, Non-stationary Data Clustering.

Thesis supervisor: Prof. A.P. Engelbrecht
Department of Computer Science
Degree: Philosophiae Doctor

A Local Network Neighbourhood Artificial Immune System

deur

Alexander J. Graaff

Opsomming

Soos wat inligting meer aanlyn toeganglik raak en vir altyd meer deel vorm van enige besigheid, is die eintlike waarde van groot hoeveelhede data in die ontdekking van verskuilde en onbekende verwantskappe en konneksies of eienskappe in die data. Die verkryging van sulke verskuilde verwantskappe kan die strategiese besluitneming van 'n besigheid beinvloed, wat weer 'n impak het op die sukses van 'n besigheid. Data groepering is een van baie metodes om data op so 'n manier te groepeer dat data patrone wat deel vorm van dieselfde groep 'n gemeenskaplike eienskap deel in vergelyking met patrone wat verspreid is in ander groepe. Hierdie tesis stel 'n nuwe kunsmatige immuun model voor vir die probleem van data groepering. Die nuwe model is geinspireer deur die netwerk teorie in immunologie en verskil van vorige netwerk gebaseerde modelle deur die model se formasie van kunsmatige limfosiet netwerke. Die voorgestelde model word eers toegepas op data groeperingsprobleme in statiese omgewings. Twee verskillende tegnieke word dan voorgestel wat die voorgestelde kunsmatige immuun model op so 'n manier verbeter dat die model die aantal groepe in 'n data stel dinamies kan bepaal met minimum tot geen gebruiker invloed. 'n Tegniek om kunsmatige data stelle te genereer vir data groepering in dinamiese omgewings word dan voorgestel. Laastens word die oorspronklik voorgestelde model sowel as die verbeterde model wat dinamies die aantal groepe in 'n data stel kan bepaal toegepas op kunsmatig genereerde dinamiese data groeperingsprobleme. Die invloed van die parameters op die groepering prestasie is ondersoek vir alle weergawes van die voorgestelde kunsmatige immuun model en word toegelig deur empiriese resultate en statistiese hipotese toetse.

Sleutelwoorde: Data Groepering, Kunsmatige Limfosiete, Affiniteit Volwassewording, Klonale Seleksie, Somatiese Hiper Mutasie, Kunsmatige Immuun Netwerke, Immuun Netwerk Topologiee, Groepering Prestasie Maatreels, Dinamiese Groepering, Groepering van Dinamiese Data.

Tesis studieleier: Prof. A.P. Engelbrecht
Departement Rekenaarwetenskap
Graad: Philosophiae Doctor

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

– Soli Deo Gloria –

"To God Alone the Glory"

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Acknowledgments

My sincere gratitude to God for all the privileges in my life and the opportunity to undertake a research study such as this. The research study, especially the research on the natural process of immunology, gave me an intense appreciation and admiration for the detail that God has put into His creation. All glory to the Creator that I can be part of His creation and that He enables me to understand a diminutive part of it.

A warm thanks to Professor Andries Engelbrecht for his continuous support, guidance and patience. Prof. Engelbrecht was always willing to listen and assist in refining my proposed ideas and methods (even the most bizarre ideas).

Also, thanks to Mr. Nelis Franken for all the brainstorming sessions on statistical hypothesis testing and probability distributions.

Last but not least, I want to thank my family and my wife, Daleen, for their support during this study. You are always there to give balance to my life.

Contents

Page

List of Figures **xiii**

List of Tables **xvii**

List of Algorithms **xviii**

1 Introduction **1**

1.1 Motivation 2
1.2 Objectives 2
1.3 Methodology 3
1.4 Contributions 4
1.5 Thesis Outline 5

2 Clustering and Quality Measures **8**

2.1 Data Clustering 9
2.2 Similarity Measures 10
2.3 Clustering Algorithms 12
2.3.1 Hierarchical Clustering 12
2.3.2 Partitional Clustering 15
2.3.3 Other Clustering Methods 19
2.4 Cluster Quality Validation 21
2.5 Cluster Quality in Dynamic and Uncertain Environments 27
2.6 Outlier Detection and Analysis 36
2.7 Alternative Computational Models for Clustering 38
2.7.1 Particle Swarm Optimisation 39
2.7.2 Self-organising Feature Map 42

2.8 Conclusion 46

3 The Natural Immune System **48**

3.1 Classical View 48
3.2 Antibodies and Antigens 49
3.3 The White Cells 51
3.3.1 The Lymphocytes 51
3.3.2 The B-Cell Lymphocyte 52
3.3.3 The Helper-T-Cell (HTC) 53
3.3.4 The Natural-Killer-T-Cell (NKTC) 54
3.4 Immunity Types 54
3.5 The Process of Affinity Maturation 56
3.6 The Network Theory 57
3.7 The Danger Theory 59
3.8 The Dendritic Cell System 61
3.9 Conclusion 63

4 Artificial Immune Systems **64**

4.1 A Basic AIS Framework 65
4.2 Representation of Antigens and Antibodies 70
4.3 Affinity as Quality Measure 71
4.3.1 A Complementary Matching Rule 72
4.3.2 The r-contiguous Matching Rule 73
4.3.3 The r-chunks Matching Rule 73
4.4 Negative Selection Models 75
4.5 Clonal Selection Models 81
4.6 Idiotypic Network Models 86
4.7 Idiotypic Network Topologies 115
4.8 Danger Theory Models 118
4.9 Conclusion 121

5 A Local Network Neighbourhood Artificial Immune System with Application to Unsupervised Data Clustering **123**

5.1 The Algorithm 123
5.2 Initialising an ALC and the ALC population 124

5.3 Reacting to an Antigen . . . 126
5.3.1 Calculating the Affinity . . . 126
5.3.2 Proliferating the Clonal Selected ALC . . . 127
5.3.3 Normalising the Affinity of an Antigen Mutated Clone . . . 127
5.3.4 Generating a Mutated Clone of an Activated ALC . . . 128
5.3.5 Secreting an Antibody for Co-stimulation . . . 128
5.4 Adapting the ALCs in a Local Network Neighbourhood . . . 129
5.4.1 Determining the Local Network Neighbourhood of an Activated ALC . . . 129
5.4.2 Average Network Affinity in a Local Network Neighbourhood . . . 130
5.4.3 Co-stimulating the Local Network Neighbourhood . . . 130
5.4.4 Clonal Expansion of a Local Network Neighbourhood . . . 131
5.4.5 Non-proliferation of the ALC Population . . . 131
5.5 Similarities and Differences with Other Network based AIS Models . . . 131
5.5.1 Training Data . . . 133
5.5.2 Population of ALCs . . . 133
5.5.3 ALC Presentation . . . 133
5.5.4 Affinity Measurement . . . 133
5.5.5 Learning the Antigen Structure . . . 134
5.5.6 Determining the Number of Clusters . . . 134
5.5.7 The Number of Parameters . . . 135
5.6 Time Complexity of LNNAIS . . . 136
5.7 Experimental Results and Analysis . . . 136
5.7.1 Data clustering problems . . . 136
5.7.2 Experimental setup and methodology . . . 138
5.7.3 Testing for statistical significance - data group 1 . . . 142
5.7.4 Testing for statistical significance - data group 2 . . . 142
5.7.5 Testing for statistical significance - data group 3 . . . 142
5.7.6 Testing for statistical significance - data group 4 . . . 144
5.7.7 ALC Population Size - Overfitting the Data . . . 150
5.7.8 Influence of LNNAIS parameters . . . 151
5.8 Conclusion . . . 156

6 Dynamically Determining the Number of Clusters Found by a Local Network Neighbourhood Artificial Immune System 166
6.1 Dynamic Data Clustering Methods . . . 167

6.2 Dynamic Clustering Techniques for LNNAIS . . . 169
6.3 Time Complexity of SDOT and IPT . . . 171
6.4 Experimental Results . . . 172
6.4.1 Iris data set . . . 173
6.4.2 Two-spiral data set . . . 176
6.4.3 Hepta data set . . . 178
6.4.4 Engytime data set . . . 178
6.4.5 Chainlink data set . . . 181
6.4.6 Target data set . . . 185
6.4.7 Ionosphere data set . . . 188
6.4.8 Glass data set . . . 190
6.4.9 Image Segmentation data set . . . 190
6.4.10 Spambase data set . . . 193
6.5 Influence of LNN$_{SDOT}$ Parameters . . . 198
6.6 Conclusion . . . 205

7 Data Clustering in Non-stationary Environments using a Local Network Neighbourhood Artificial Immune System 206
7.1 Clustering Performance Measures for Non-stationary Environments . . . 207
7.2 Data Migration Types . . . 208
7.3 Generating Artificial Non-stationary Data . . . 209
7.4 Sensitivity of LNNAIS parameters . . . 211
7.4.1 Pattern Migration . . . 212
7.4.2 Cluster Migration . . . 212
7.4.3 Centroid Migration . . . 218
7.5 Experimental Results . . . 221
7.5.1 Pattern Migration . . . 222
7.5.2 Cluster Migration . . . 237
7.5.3 Centroid Migration . . . 248
7.6 Conclusion . . . 259

8 Conclusion 261
8.1 Summary . . . 261
8.2 Future Research . . . 263

References **267**

A List of Symbols **284**

B Derived Publications **290**

List of Figures

Figure Page

2.1 Linking Techniques in Hierarchical Clustering 13
2.2 Dynamic Objective Function (equation (2.53)) 30
2.3 Clustering in Dynamic Environments 31
2.4 Neighbourhood Topologies in PSO 41
2.5 Self-organising Feature Map 43
2.6 Neighbourhood Arrangements in SOM 44

3.1 Antigen-Antibody-Complex 50
3.2 White Cell Types 50
3.3 Life Cycle of a Lymphocyte 52
3.4 B-Cell Develops into Plasma Cell, Producing Antibodies 53
3.5 Macrophage and NKTC 54
3.6 Immunity Types 55
3.7 Affinity Maturation of Lymphocytes 56
3.8 Idiotypic Network of Antibodies and B-Cells 58
3.9 Co-Stimulation of T-Cell by an APC 60
3.10 Maturation of Dendritic Cells 61

4.1 Shape space ϖ as bounded region $\mathcal{R}^2$ with volume V 71
4.2 r-contiguous matching rule 73
4.3 Adapted Negative Selection 79
4.4 Linear Network Topology 116
4.5 Simple Cyclic Network Topology 117
4.6 Affinity Matrix Topology with Normalisation 117
4.7 Cayley Tree Network Topology 118

5.1 Flow chart of LNNAIS algorithm . . . 125
5.2 Adapting an ALC Network Neighbourhood . . . 130
5.3 Determining the Number of Clusters in LNNAIS . . . 135
5.4 ALC Population Size Ratios of SMAIN, DWB and LNNAIS . . . 150
5.5 Two-spiral data set ($\varepsilon_{clone} = 6$): Effect of the ALC population size with a constant clonal level threshold . . . 152
5.6 Chainlink data set ($\varepsilon_{clone} = 8$): Effect of the ALC population size with a constant clonal level threshold . . . 153
5.7 Glass data set ($\varepsilon_{clone} = 8$): Effect of the ALC population size with a constant clonal level threshold . . . 154
5.8 Image Segmentation data set ($\varepsilon_{clone} = 27$): Effect of the ALC population size with a constant clonal level threshold . . . 155
5.9 Two-spiral data set ($\mathcal{B}_{max} = 39$): Effect of the neighbourhood size with a constant ALC population size . . . 157
5.10 Chainlink data set ($\mathcal{B}_{max} = 24$): Effect of the neighbourhood size with a constant ALC population size . . . 158
5.11 Glass data set ($\mathcal{B}_{max} = 24$): Effect of the neighbourhood size with a constant ALC population size . . . 159
5.12 Image Segmentation data set ($\mathcal{B}_{max} = 20$): Effect of the neighbourhood size with a constant ALC population size . . . 160
5.13 Two-spiral data set ($\rho = 3$): Effect of the clonal level threshold with a constant neighbourhood size . . . 161
5.14 Chainlink data set ($\rho = 3$): Effect of the clonal level threshold with a constant neighbourhood size . . . 162
5.15 Glass data set ($\rho = 3$): Effect of the clonal level threshold with a constant neighbourhood size . . . 163
5.16 Image Segmentation data set ($\rho = 2$): Effect of the clonal level threshold with a constant neighbourhood size . . . 164

6.1 Network Affinity Plot . . . 170
6.2 Optimal number of clusters obtained by K-means and LNNAIS for the iris data set 173
6.3 Convergence of LNNAIS using SDOT and IPT to optimal K for iris data set . . . 174
6.4 Histogram of the number of clusters detected in the iris data set by LNN_{SDOT} . . . 175
6.5 Optimal number of clusters obtained by K-means and LNNAIS for the two-spiral data set . . . 176

6.6 Histogram of the number of clusters detected in the two-spiral data set by LNN$_{SDOT}$ 177
6.7 Convergence of LNNAIS using SDOT and IPT to optimal K for two-spiral data set 177
6.8 Optimal number of clusters obtained by K-means and LNNAIS for the hepta data set 179
6.9 Histogram of the number of clusters detected in the hepta data set by LNN$_{SDOT}$. 179
6.10 Convergence of LNNAIS using SDOT and IPT to optimal K for hepta data set . . 180
6.11 Optimal number of clusters obtained by K-means and LNNAIS for the engytime data set 181
6.12 Histogram of the number of clusters detected in the engytime data set by LNN$_{SDOT}$ 182
6.13 Convergence of LNNAIS using SDOT and IPT to optimal K for engytime data set 182
6.14 Optimal number of clusters obtained by K-means and LNNAIS for the chainlink data set 183
6.15 Histogram of the number of clusters detected in the chainlink data set by LNN$_{SDOT}$ 184
6.16 Convergence of LNNAIS using SDOT and IPT to optimal K for chainlink data set 184
6.17 Optimal number of clusters obtained by K-means and LNNAIS for the target data set 186
6.18 Histogram of the number of clusters detected in the target data set by LNN$_{SDOT}$. 186
6.19 Convergence of LNNAIS using SDOT and IPT to optimal K for target data set . . 187
6.20 Optimal number of clusters obtained by K-means and LNNAIS for the ionosphere data set 189
6.21 Histogram of the number of clusters detected in the ionosphere data set by LNN$_{SDOT}$ 189
6.22 Convergence of LNNAIS using SDOT and IPT to optimal K for ionosphere data set 190
6.23 Optimal number of clusters obtained by K-means and LNNAIS for the glass data set 191
6.24 Histogram of the number of clusters detected in the glass data set by LNN$_{SDOT}$. 192
6.25 Convergence of LNNAIS using SDOT and IPT to optimal K for glass data set . . 192
6.26 Optimal number of clusters obtained by K-means and LNNAIS for the image segmentation data set 194
6.27 Histogram of the number of clusters detected in the image segmentation data set by LNN$_{SDOT}$ 194
6.28 Convergence of LNNAIS using SDOT and IPT to optimal K for image data set . 195
6.29 Optimal number of clusters obtained by K-means and LNNAIS for the spambase data set 195

6.30 Histogram of the number of clusters detected in the spambase data set by LNN$_{SDOT}$ 196
6.31 Convergence of LNNAIS using SDOT and IPT to optimal K for spambase data set 196

7.1 Parallel Coordinates of LNNAIS Parameters for Pattern Migration 213
7.2 Parallel Coordinates of LNNAIS Parameters for Cluster Migration 215
7.3 Parallel Coordinates of LNNAIS Parameters for Centroid Migration 217
7.4 Parallel Coordinates of ρ for Centroid Migration 218
7.5 Heat Maps of LNNAIS Parameters for Centroid Migration ($N = 3$) 219
7.6 Pattern Migration ($N = 8, |C| = 25, \tilde{f} = 3, \tilde{s} = 3$): Quantifying each model's partitioning quality with regards to Q_{ratio}, $|\mathcal{B}|$, J_{intra}, J_{inter} and K 231
7.7 Cluster Migration ($N = 8, |C| = 25, \tilde{f} = 3, \tilde{s} = 3$): Quantifying each model's partitioning quality with regards to Q_{ratio}, $|\mathcal{B}|$, J_{intra}, J_{inter} and K 241
7.8 Centroid Migration ($N = 8, |C| = 25, \tilde{f} = 3, \tilde{s} = 3$): Quantifying each model's partitioning quality with regards to Q_{ratio}, $|\mathcal{B}|$, J_{intra}, J_{inter} and K 251

List of Tables

Table Page

4.1 List of Immunological Terms Mapped to Data Analysis Terms 121

5.1 List of Eleven Benchmarking Data Sets for Clustering 137
5.2 CPSO Parameter Values 139
5.3 SMAIN Parameter Values 139
5.4 DWB Parameter Values 140
5.5 Opt-aiNET Parameter Values 141
5.6 LNNAIS Parameter Values 141
5.7 Descriptive Statistics: Data Group 1 143
5.8 Statistical Hypothesis Testing between LNNAIS and Other Models based on Q_{ratio}: Data Group 1 ($\alpha = 0.05$; with continuity correction; unpaired; non-directional) 144
5.9 Descriptive Statistics: Data Group 2 145
5.10 Statistical Hypothesis Testing between LNNAIS and Other Models based on Q_{ratio}: Data Group 2 ($\alpha = 0.05$; with continuity correction; unpaired; non-directional) 146
5.11 Statistical Hypothesis Testing between LNNAIS and Other Models based on Q_{ratio}: Data Group 3 ($\alpha = 0.05$; with continuity correction; unpaired; non-directional) 146
5.12 Descriptive Statistics: Data Group 3 147
5.13 Descriptive Statistics: Data Group 4 148
5.14 Statistical Hypothesis Testing between LNNAIS and Other Models based on Q_{ratio}: Data Group 4 ($\alpha = 0.05$; with continuity correction; unpaired; non-directional) 149

6.1 LNNAIS Parameter Values 172

6.2 Descriptive Statistics: Iris 175
6.3 Descriptive Statistics: Two-spiral 178
6.4 Descriptive Statistics: Hepta 180
6.5 Descriptive Statistics: Engytime 183
6.6 Descriptive Statistics: Chainlink 185
6.7 Descriptive Statistics: Target 187
6.8 Descriptive Statistics: Ionosphere 188
6.9 Descriptive Statistics: Glass 191
6.10 Descriptive Statistics: Image Segmentation 193
6.11 Descriptive Statistics: Spambase 197
6.12 Statistical Hypothesis Testing between All Models for all data sets based on Q_{ratio} as performance criteria ($\alpha = 0.05$; with continuity correction; unpaired; non-directional) 198
6.13 Effect of $\mathcal{B}_{max}$ on the number of detected clusters, K, by LNN$_{SDOT}$ 199
6.14 Effect of ε_{clone} on the number of detected clusters, K, by LNN$_{SDOT}$ 201
6.15 Effect of ρ on the number of detected clusters, K, by LNN$_{SDOT}$ 203

7.1 Effect of Pattern Migration on Clustering Performance of LNNAIS 214
7.2 Effect of Cluster Migration on Clustering Performance of LNNAIS 216
7.3 Effect of Centroid Migration on Clustering Performance of LNNAIS 220
7.4 LNNAIS Parameter Values - Pattern Migration 222
7.5 LNNAIS Parameter Values - Cluster Migration 223
7.6 LNNAIS Parameter Values - Centroid Migration 223
7.7 DWB Parameter Values - Pattern Migration 224
7.8 DWB Parameter Values - Cluster Migration 225
7.9 DWB Parameter Values - Centroid Migration 226
7.10 SMAIN Parameter Values - Pattern Migration 227
7.11 SMAIN Parameter Values - Cluster Migration 227
7.12 SMAIN Parameter Values - Centroid Migration 228
7.13 Statistical Hypothesis Testing between LNNAIS and Other Models ($\alpha = 0.05$; with continuity correction; unpaired; non-directional) for Pattern migration at different dimensions ($|C|$=25; $\tilde{f}$=3; $\tilde{s}$=3) 228
7.14 Statistical Hypothesis Testing between LNNAIS and Other Models ($\alpha = 0.05$; with continuity correction; unpaired; non-directional) for Pattern migration at different cluster sizes (N=8; $\tilde{s}$=3; $\tilde{f}$=3) 229

7.15 Statistical Hypothesis Testing between LNNAIS and Other Models ($\alpha = 0.05$; with continuity correction; unpaired; non-directional) for Pattern migration at different frequencies of change (N=8; $|C|$=25; $\tilde{s}$=3) 229
7.16 Statistical Hypothesis Testing between LNNAIS and Other Models ($\alpha = 0.05$; with continuity correction; unpaired; non-directional) for Pattern migration at different severities of change (N=8; $|C|$=25; $\tilde{f}$=3) 230
7.17 Descriptive Statistics: Pattern migration at different dimensions ($|C|$=25; $\tilde{f}$=3; $\tilde{s}$=3) 232
7.18 Descriptive Statistics: Pattern migration at different cluster sizes (N=8; $\tilde{s}$=3; $\tilde{f}$=3) 233
7.19 Descriptive Statistics: Pattern migration at different frequencies of change (N=8; $|C|$=25; $\tilde{s}$=3) 234
7.20 Descriptive Statistics: Pattern migration at different severities of change (N=8; $|C|$=25; $\tilde{f}$=3) 236
7.21 Statistical Hypothesis Testing between LNNAIS and Other Models ($\alpha = 0.05$; with continuity correction; unpaired; non-directional) for Cluster migration at different dimensions ($|C|$=25; $\tilde{f}$=3; $\tilde{s}$=3) 238
7.22 Statistical Hypothesis Testing between LNNAIS and Other Models ($\alpha = 0.05$; with continuity correction; unpaired; non-directional) for Cluster migration at different cluster sizes (N=8; $\tilde{s}$=3; $\tilde{f}$=3) 239
7.23 Statistical Hypothesis Testing between LNNAIS and Other Models ($\alpha = 0.05$; with continuity correction; unpaired; non-directional) for Cluster migration at different frequencies of change (N=8; $|C|$=25; $\tilde{s}$=3) 239
7.24 Statistical Hypothesis Testing between LNNAIS and Other Models ($\alpha = 0.05$; with continuity correction; unpaired; non-directional) for Cluster migration at different severities of change (N=8; $|C|$=25; $\tilde{f}$=3) 240
7.25 Descriptive Statistics: Cluster migration at different dimensions ($|C|$=25; $\tilde{f}$=3; $\tilde{s}$=3) 242
7.26 Descriptive Statistics: Cluster migration at different cluster sizes (N=8; $\tilde{s}$=3; $\tilde{f}$=3) 243
7.27 Descriptive Statistics: Cluster migration at different frequencies of change (N=8; $|C|$=25; $\tilde{s}$=3) 244
7.28 Descriptive Statistics: Cluster migration at different severities of change (N=8; $|C|$=25; $\tilde{f}$=3) 246
7.29 Statistical Hypothesis Testing between LNNAIS and Other Models ($\alpha = 0.05$; with continuity correction; unpaired; non-directional) for Centroid migration at different dimensions ($|C|$=25; $\tilde{f}$=3; $\tilde{s}$=3) 249

7.30 Statistical Hypothesis Testing between LNNAIS and Other Models ($\alpha = 0.05$; with continuity correction; unpaired; non-directional) for Centroid migration at different cluster sizes (N=8; $\tilde{s}$=3; $\tilde{f}$=3) 249
7.31 Statistical Hypothesis Testing between LNNAIS and Other Models ($\alpha = 0.05$; with continuity correction; unpaired; non-directional) for Centroid migration at different frequencies of change (N=8; $|C|$=25; $\tilde{s}$=3) 250
7.32 Statistical Hypothesis Testing between LNNAIS and Other Models ($\alpha = 0.05$; with continuity correction; unpaired; non-directional) for Centroid migration at different severities of change (N=8; $|C|$=25; $\tilde{f}$=3) 252
7.33 Descriptive Statistics: Centroid migration at different dimensions ($|C|$=25; $\tilde{f}$=3; $\tilde{s}$=3) 253
7.34 Descriptive Statistics: Centroid migration at different cluster sizes (N=8; $\tilde{s}$=3; $\tilde{f}$=3) 254
7.35 Descriptive Statistics: Centroid migration at different frequencies of change (N=8; $|C|$=25; $\tilde{s}$=3) 255
7.36 Descriptive Statistics: Centroid migration at different severities of change (N=8; $|C|$=25; $\tilde{f}$=3) 256

List of Algorithms

Algorithm Page

2.1 Basic K-means 16
2.2 Fuzzy C-means 18
2.3 Basic Gaussian Mixture using EM 20
2.4 PSO Clustering Algorithm 42
2.5 General SOM Algorithm 43

4.1 AIS Algorithm Template 68
4.2 Training ALCs with *negative selection* 76
4.3 Artificial Immune Network (AINE) 91
4.4 Resource allocation in AINE 92
4.5 Self Stabilising AIS 94
4.6 Stable Memory Artificial Immune Network 96
4.7 Fuzzy Artificial Immune Network (Fuzzy AINE) 98
4.8 Dynamic Weighted B-Cell 101
4.9 aiNet Learning Algorithm 106
4.10 opt-aiNet Learning Algorithm 108
4.11 dopt-aiNet Learning Algorithm 112
4.12 One-dimensional Mutation Algorithm 113
4.13 Gene Duplication Mutation Algorithm 113
4.14 Cell Line Suppression Algorithm 114

5.1 High Level LNNAIS Algorithm 126
5.2 Adapting the Neighbourhood, $\mathcal{N}_{h,\rho}$, to an Activated ALC, $\mathbf{b}_h$ 132

Chapter 1

Introduction

There are certain basic attributes which define a human being. Focusing on the world's human population, two traits which are shared among humans might be that all humans are living organisms and live on planet earth. The human population can easily be divided into two separate sub-populations (groups) based on the gender attribute. Both of the male and female populations can be further divided on the age attribute into multiple sub-populations. The end result might be a group of humans which share common traits such as teenage males between the age of 12 and 20. Another group might represent female humans above the age of 50. Therefore, humans forming part of the same group share some common trait compared to humans which form part of other groups. Thus, natural clustering exists in our daily lives. Whether the attribute is based on choice of music genre, political interest and/or religion, there is a spontaneous and natural clustering in society which is determined by the similarity or dissimilarity of different attributes. Since teenage males will age with time they will eventually form part of a different group. This implies that the population is non-stationary. On a smaller scale, spontaneous clustering also exists in the natural immune system where lymphocytes co-operate by co-stimulating each other in response to an invading antigen. The end result is the formation of lymphocyte networks (groups) with a similar structure to react to the invading antigen. Since the body is also frequently exposed to unseen and unfamiliar antigens, the natural immune system needs to adapt to changes in the antigen structure. Thus, the antigen population is also non-stationary. This thesis proposes an artificial immune model which is based on the network theory of co-stimulating lymphocytes and is applied to the problem of data clustering in stationary and non-stationary environments.

1.1 Motivation

Clustering of observations or features into different partitions in order to discover hidden traits in the data is of considerable value. The discovered traits could influence the strategic decisions of a business, the effect of medicine on certain diseases or highlight emigration/immigration patterns of citizens in a country. It is clear that clustering is a fundamental cornerstone in decision making within various disciplines. Many of the existing network theory based artificial immune systems have been applied to data clustering. The formation of artificial lymphocyte (ALC) networks represents potential clusters in the data. Although these models do not require any user specified parameter of the number of required clusters to cluster the data, these models do have a drawback in the techniques used to determine the number of ALC networks. Another drawback is that these models have a large number of user parameters which control the outcome of the clustering performance. Specifying the optimal set of values for these control parameters is a time-consuming and challenging task, since there could be an optimal set for each data set that needs to be clustered and the optimal set of values has a high probability to change in a non-stationary environment. Furthermore, the techniques utilised by these models to determine the number of ALC networks are either based on a network affinity threshold with a proximity matrix of network affinities between the ALCs in the population or a hybrid approach is taken by clustering the ALC population using a clustering algorithm. Specifying the correct network affinity threshold to obtain the correct or required number of clusters can be a formidable task, especially in a non-stationary environment. A potential drawback to a hybrid approach is that the formed sub-nets might not always contain ALCs with a good or generic representation of the data. Furthermore, both of these techniques are computationally expensive. This thesis proposes a network based artificial immune model which is applied to data clustering in stationary and non-stationary environments. The proposed model is independent of a network affinity threshold and do not need to follow a hybrid approach to determine the number of clusters. Furthermore, the proposed model has considerable less control parameters in comparison to existing network based AIS models. Also, the proposed model is enhanced to dynamically determine the number of clusters in a data set.

1.2 Objectives

The primary objectives of this thesis can be summarised as follows:

- To develop an alternative artificial lymphocyte network topology which is independent of a network affinity threshold and do not need to follow a hybrid approach to determine the number of clusters.

- To develop a theoretical network based artificial immune model which utilises the alternative network topology for data clustering.
- To develop two techniques which can be used with the proposed artificial immune model to dynamically determine the number of clusters in a data set.
- To develop a method for the generation of synthetic non-stationary data which is based on different data migration types.
- To show that the proposed model can be applied to data clustering of non-stationary environments.

1.3 Methodology

The algorithms developed in this thesis are first presented and discussed. Empirical results were obtained using a selection of data clustering problems with known characteristics and which covers a good distribution of problems in stationary environments. The results of two classical clustering algorithms and three network based artificial immune models were also reported for the same selection of stationary data clustering problems. These results showed the relative clustering performance of the proposed model when compared to other existing clustering and network based artificial immune models. The same selection of stationary data clustering problems was used for the purpose of evaluating the capability of the enhanced version of the proposed model to dynamically determine the number of clusters in a data set. Results of the enhanced models were also compared to the results obtained from a classical clustering model to show the relative clustering performance of the enhanced models. Furthermore, the time complexity of these models was also discussed. Various synthetic non-stationary data clustering problems with known characteristics were also used to evaluate the clustering performance of the proposed model in a non-stationary environment. The results of two network based artificial immune models were also reported for the same synthetic non-stationary data clustering problems. These results showed the relative clustering performance of the proposed model when compared to other existing network based artificial immune models. All the reported results are averages and standard deviations taken over 50 runs, since the proposed model is population based and has a stochastic nature. All parameter values for the respective algorithms were found empirically to deliver the best performance for clustering the applicable data set. A non-parametric Mann-Whitney U hypothesis test between the clustering quality of the proposed model and the clustering quality of each of the other models was used to investigate whether there is a statistical significant difference between the clustering quality of two models for a specific data set or not.

1.4 Contributions

The main contributions of this thesis are:

- The development of a novel network based artificial immune model which utilises an index based artificial neighbourhood network topology for data clustering of stationary environments. The developed model has less control parameters than existing network based artificial immune models, is independent of a network affinity threshold and does not need to follow a hybrid approach to determine the number of clusters.
- The development of two techniques which enhances the proposed network based artificial immune model to dynamically determine the number of clusters in a data set.
- The development of a simple method to generate synthetic non-stationary data which follows different data migration types.
- The application of the proposed network based artificial immune model to the clustering of non-stationary environments.
- Empirical analysis of the behaviour of all versions of the proposed network based artificial immune model under different parameter settings.

The following list of published or currently reviewed articles support the main contributions of this thesis:

A.J. Graaff and A.P. Engelbrecht. Chapter 18: Natural Immune System. *Computational Intelligence: An Introduction*, 2nd Edition, A.P. Engelbrecht (Author), John Wiley & Sons, October 2007.

A.J. Graaff and A.P. Engelbrecht. Chapter 19: Artificial Immune Models. *Computational Intelligence: An Introduction*, 2nd Edition, A.P. Engelbrecht (Author), John Wiley & Sons, October 2007.

A.J. Graaff and A.P. Engelbrecht. A local network neighbourhood artificial immune system for data clustering. In *IEEE Congress on Evolutionary Computation*, CEC 2007., pp. 260–267, 2007.

A.J. Graaff and A.P. Engelbrecht. Towards a self regulating local network neighbourhood artificial immune system for data clustering. In *IEEE Congress on Evolutionary Computation*, CEC 2008.(IEEE World Congress on Computational Intelligence), pp. 633–640, 2008.

A.J. Graaff and A.P. Engelbrecht. Optimised Coverage of Non-self with Evolved Lymphocytes in an Artificial Immune System. *International Journal of Computational Intelligence Research*, vol. 2, no. 2, pp. 127–150, 2006.

A.J. Graaff and A.P. Engelbrecht. Clustering Data in an Uncertain Environment using an Artificial Immune System. *Pattern Recognition Letters*, vol. 32, no. 2, pp. 342–351, January 2011.

A.J. Graaff and A.P. Engelbrecht. Using sequential deviation to dynamically determine the number of clusters found by a local network neighbourhood artificial immune system. *Applied Soft Computing*, vol. 11, pp. 2698–2713, March 2011.

A.J. Graaff and A.P. Engelbrecht. Clustering Data in Stationary Environments with a Local Network Neighborhood Artificial Immune System. *International Journal of Machine Learning and Cybernetics*, submitted May 2011.

1.5 Thesis Outline

The thesis is organised as follows:

- *Chapter 2* discusses the problem of data clustering. A formal definition of data clustering is given with an elaboration on different similarity measures and existing clustering approaches. This is followed by an overview of different clustering performance measures to evaluate the partitioning quality of a clustering algorithm applied to stationary data. The different performance measures applied to optimisation algorithms for problems in non-stationary environments are then discussed. These performance measures are used to quantify and define performance measures that can be used to evaluate the partitioning quality of clustering algorithms in non-stationary environments. Furthermore, a brief introduction to outliers and outlier detection is given as well as a discussion of two alternative computational models which can be applied to the problem of data clustering.

- *Chapter 3* reviews the functional process of the natural immune system. The different theories in immunology regarding the functioning and organisational behavior between lymphocytes are discussed. These theories include the classical view, clonal selection theory, network theory, and danger theory. The classical view is first discussed in detail, since the other theories are based on concepts and elements within the classical view. The classical view forms a base onto which the other theories are explained. A brief review of the dendritic cell system is also given.

- *Chapter 4* discusses some of the most familiar artificial immune system (AIS) models which are inspired by the different theories in the science of immunology. The chapter highlights the basic components of an AIS model and introduces the shape space model. Furthermore, an overview of different measures of affinity between an artificial lymphocyte and an antigen pattern within a specific shape space is given which is followed by an overview on the different matching rules to determine whether an ALC binds to an antigen pattern. The remainder of the chapter briefly discusses some of the AIS models which are respectively inspired by the negative selection, clonal selection and danger theories. Since the proposed AIS model in this thesis is inspired by and mostly based on the network theory, a more detailed overview is given on existing network based AIS models within the context of data clustering. Also, different theoretical approaches to determine the possible interactions in an ALC network are discussed.

- *Chapter 5* presents a novel network theory inspired artificial immune system. Specifically, the network topology of co-stimulated lymphocytes inspired the modelling of the local network neighbourhood artificial immune system (LNNAIS). The chapter introduces the concept of an index based neighbourhood topology which is utilised by LNNAIS to determine the network connectivity between ALCs. Each of the formed local ALC neighbourhood structures represents a cluster in a data set. The differences and similarities between existing network based AIS models and the proposed LNNAIS model are also discussed. The proposed LNNAIS model is compared to classical clustering algorithms and existing network based AIS models which are applied to data clustering problems. Furthermore, a sensitivity analysis is also done on the proposed model to investigate the influence of the model's parameters on the quality of the clusters.

- *Chapter 6* proposes two techniques which can be used with the proposed local network neighbourhood artificial immune model to dynamically determine the number of clusters in a data set. The first technique utilises cluster validity indices and is similar to the multiple execution approach, though computationally less expensive. The second technique is based on sequential deviation outlier detection. The results of a multiple execution approach of K-means clustering is compared to the results obtained from both the proposed LNNAIS techniques to dynamically determine the number of clusters in a data set. The influence of the parameters of LNNAIS on the number of dynamically determined clusters in a data set is also investigated.

- *Chapter 7* defines and discusses different non-stationary environments. A technique to generate synthetic data sets for each of the defined non-stationary environments is proposed. Different synthetic data sets are then generated based on the defined non-stationary environments. The proposed local network neighbourhood artificial immune model and the enhanced version of the model to dynamically determine the number of clusters are applied to the clustering of the generated synthetic non-stationary data. The results are compared to the results obtained from two existing network based artificial immune models to cluster the non-stationary data. The influence of the different non-stationary environments on the parameters of the proposed model is also investigated.
- *Chapter 8* highlights the conclusions of this thesis and presents ideas relating to possible future work.
- *Appendix A* lists and defines the symbols used throughout this thesis.
- *Appendix B* lists the publications derived from this thesis.

Chapter 2

Clustering and Quality Measures

This chapter gives a formal definition of data clustering. A brief overview of different clustering techniques is given. Different similarity measures are discussed as well as different cluster quality measures. These quality measures are then discussed in the context of dynamic and uncertain environments, i.e. clustering non-stationary data.
The chapter is organised as follows:

- Section 2.1 gives a formal definition of data clustering and the notation used by the rest of the chapter.
- Section 2.2 discusses the different distance-based *similarity* measures.
- Section 2.3 discusses the most familiar clustering algorithms which are categorised into *hierarchical* clustering or *partitional* clustering methods.
- Section 2.4 introduces the different categories of cluster validity measures to evaluate the partitioning quality of a clustering algorithm. The section discusses the cluster validity indices which form part of the *relative* criteria.
- Section 2.5 introduces different measures to evaluate the performance of an optimisation algorithm which is applied to problems in non-stationary environments. These performance measures are used to quantify and define measures that can be used to evaluate the partitioning quality of clustering algorithms in non-stationary environments.
- Section 2.6 defines outliers and explains three different approaches for outlier detection.
- Section 2.7 discusses two alternative computational algorithms which can be applied to the problem of data clustering.

- Section 2.8 concludes the chapter by giving an overall summary of the chapter and discussing the relevance of each section to the work in this thesis.

2.1 Data Clustering

Patterns in a data set can be structured into different groups in such a way that patterns within the same group are more *similar* compared to patterns across different groups. Each of the formed clusters is represented by a *centroid* [122]. Data clustering can formally be defined as follows [15, 96]:
Let P be the data set of patterns in N-dimensional space that needs to be clustered. Thus, $P = \{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_i, \ldots, \mathbf{p}_{I-1}, \mathbf{p}_I\}$ where $\mathbf{p}_i$ is an N-dimensional *feature vector* (pattern) and I is the number of feature vectors. The partitioning of P into K clusters, $\{C_1, C_2, \ldots, C_K\}$, satisfies the following conditions:

1. $|C_k| \neq 0, k = 1, 2, \ldots, K$, meaning that clusters are not allowed to be empty;

2. $P = \cup_{k=1}^{K} C_k$, meaning that each feature vector is assigned to a cluster;

3. $\left|C_k \cap C_j\right| = 0, k \neq j$, meaning that each feature vector is assigned to only one cluster (in the case of *crisp* or *hard* clustering, i.e. *exclusive clustering*); or

4. $\left|C_k \cap C_j\right| > 0, k \neq j$, meaning that each feature vector can be assigned to more than one cluster with a certain degree. *Fuzzy* clustering is an example of *overlapping clustering* for which this condition holds.

The most general measure of *similarity* or *dissimilarity* between feature vectors is based on the distance between these vectors (e.g. Euclidean distance). A cluster's centroid can describe a specific *concept*. Feature vectors with a *similar* or common concept are grouped together. Clustering algorithms are applied to data clustering and compression [26, 64, 174], image segmentation [95, 145, 151], and vector and color image quantization [9, 145, 185]. The following section discusses some of the distance-based similarity measures between feature vectors.

2.2 Similarity Measures

This section discusses the different distance-based *similarity* measures. One of these distance measures is the Minkowski distance between multidimensional feature vectors, defined as [96]

$$\begin{aligned}
\sigma_\varepsilon\left(\mathbf{p}_i,\mathbf{p}_j\right) &= \left[\sum_{n=1}^{N}\left(\mathbf{p}_{i,n}-\mathbf{p}_{j,n}\right)^\varepsilon\right]^{\frac{1}{\varepsilon}} &(2.1)\\
&= \left\|\mathbf{p}_i-\mathbf{p}_j\right\|^\varepsilon &(2.2)
\end{aligned}$$

where N is the dimensions of feature vectors $\mathbf{p}_i$ and $\mathbf{p}_j$. The Euclidean distance is derived from the Minkowski measure by setting $\varepsilon = 2$ [15, 96]. The Euclidean distance is the most commonly used *similarity* measure, which is defined as

$$\begin{aligned}
\sigma_2\left(\mathbf{p}_i,\mathbf{p}_j\right) &= \left[\sum_{n=1}^{N}\left(\mathbf{p}_{i,n}-\mathbf{p}_{j,n}\right)^2\right]^{\frac{1}{2}} &(2.3)\\
&= \sqrt{\sum_{n=1}^{N}\left(\mathbf{p}_{i,n}-\mathbf{p}_{j,n}\right)^2} &(2.4)\\
&= \left\|\mathbf{p}_i-\mathbf{p}_j\right\|^2 &(2.5)
\end{aligned}$$

The Manhattan distance between two feature vectors is the sum of the absolute differences of their features (attributes) and can be derived from the Minkowski measure by setting $\varepsilon = 1$ [15]. The Manhattan distance is defined as

$$\sigma\left(\mathbf{p}_i,\mathbf{p}_j\right) = \sum_{n=1}^{N}\left|\mathbf{p}_{i,n}-\mathbf{p}_{j,n}\right| \tag{2.6}$$

The distance between two feature vectors can also be calculated as the maximum absolute difference between the values of each dimension. This distance measure is known as the Chebychev distance, defined as [132]

$$\sigma\left(\mathbf{p}_i,\mathbf{p}_j\right) = \max_{n=1,\ldots,N}\left|\mathbf{p}_{i,n}-\mathbf{p}_{j,n}\right| \tag{2.7}$$

The Chebychev distance is more appropriate in cases where the (dis)similarity between two feature vectors is reflected in individual dimensions. The Chebychev distance is also sensitive to outliers.

A drawback of the Minkowski distance measure (including all the derivatives like Euclidean distance) is what is known as the 'curse of dimensionality' [14]. The general understanding of the 'curse of dimensionality' is that with an increase in dimensionality of space the distribution of distances between the feature vectors in space becomes uniform [1]. In the context of data clustering this means that the larger the dimensionality of the search space, the larger the total space that needs to be explored in order to capture a part of the data. The Manhatten distance measure is however more preferable than the Euclidean distance measure for high dimensional data [1].

Another measure of similarity between two feature vectors is the cosine similarity measure [15]. Different to the previously discussed distance measures, the cosine similarity measures the angle between two feature vectors. The cosine similarity, Γ, between two feature vectors $\mathbf{p}_i$ and $\mathbf{p}_j$ is defined as

$$\Gamma\left(\mathbf{p}_i, \mathbf{p}_j\right) = arccos\left(\frac{\mathbf{p}_i \bullet \mathbf{p}_j}{\|\mathbf{p}_i\| \, \|\mathbf{p}_j\|}\right) \tag{2.8}$$

where $\mathbf{p}_i \bullet \mathbf{p}_j$ is the *dot product* between vectors $\mathbf{p}_i$ and $\mathbf{p}_j$ and $\Gamma \in [0, \pi]$. The value of Γ indicates the degree of similarity or dissimilarity between two vectors. Γ values closer to 0 imply a higher similarity between two vectors, and Γ values closer to π imply a higher dissimilarity between two vectors. Thus, if $\Gamma = \pi$, then the two vectors are exact opposites from one another. $\Gamma = \frac{\pi}{2}$ means that the two vectors are independent, and when $\Gamma = 0$ the two vectors are exactly the same. An advantage of the cosine similarity measure compared to the Minkowski measure is that the dissimilarity (distance) does not increase with an increase in the number of dimensions. The cosine similarity measure is therefore not influenced by the 'curse of dimensionality', making the cosine similarity measure more appropriate for clustering data of high dimensionality.

The Mahalanobis distance calculates the probability that a feature vector belongs to a set of given feature vectors [15, 96]. The distance between the feature vector and the average of the set gives an indication of the probability that a feature vector belongs to the set. Thus, a closer distance to the average has a higher probability of membership. The Mahalanobis distance can also measure the dissimilarity between two feature vectors, and is defined as

$$\sigma\left(\mathbf{p}_i, \mathbf{p}_j\right) = \sqrt{\left(\mathbf{p}_i - \mathbf{p}_j\right)^T \mathbf{Z}^{-1} \left(\mathbf{p}_i - \mathbf{p}_j\right)} \tag{2.9}$$

where $\mathbf{Z}$ is the covariance matrix of the given set of feature vectors and $\left(\mathbf{p}_i - \mathbf{p}_j\right)^T$ is the transpose of vector $\left(\mathbf{p}_i - \mathbf{p}_j\right)$. Thus, from the above definition, each feature vector is given a weight which is based on the vector's variance. The Mahalanobis distance is only appropriate to a set of feature vectors with a multivariate Gaussian distribution.

For all of the previously discussed distance measures, it is assumed that a feature vector consists of continuous features (attributes), i.e. $\mathbf{p}_{i,n} \in \Re, \forall n$ where n is the n-th attribute of feature vector $\mathbf{p}_i$. In cases where attributes are nominal-valued, the Hamming distance is used to measure similarity or rather dissimilarity [73]. The Hamming distance between two feature vectors of equal length is the number of positions which are different between the two vectors, defined as

$$\sigma\left(\mathbf{p}_i, \mathbf{p}_j\right) = \sum_{n=1}^{N} 1 \quad \forall \mathbf{p}_{i,n} \neq \mathbf{p}_{j,n} \tag{2.10}$$

Thus for feature vectors in binary space, i.e. $\mathbf{p}_i \in \{0,1\}^N, \forall i$, the above function can be re-defined as

$$\sigma\left(\mathbf{p}_i, \mathbf{p}_j\right) = \sum_{n=1}^{N} \oplus \left(\mathbf{p}_{i,n}, \mathbf{p}_{j,n}\right) \tag{2.11}$$

where $\oplus$ is the exclusive-or between the bits of $\mathbf{p}_i$ and $\mathbf{p}_j$, n is the bit-index and N is the size of the binary string (dimensions).

Another familiar similarity measure not covered in this section is *Pearson's* correlation coefficient. The interested reader is referred to [75, 83] for more information.

2.3 Clustering Algorithms

Data clustering algorithms can be categorised into *hierarchical* clustering or *partitional* clustering methods. This section highlights the differences between the aforementioned categories and discusses the most familiar clustering algorithms found in each category.

2.3.1 Hierarchical Clustering

Hierarchical clustering methods iteratively partition a data set into a hierarchy of clusters. This means that each level of the hierarchy consists of a number of clusters, which are obtained by further partitioning of the clusters in the preceding level. A hierarchical clustering algorithm is either *agglomerative* or *divisive* [56, 96]. In both cases, a similarity measurement is used to either merge clusters or divide clusters, generating a tree-like structure.

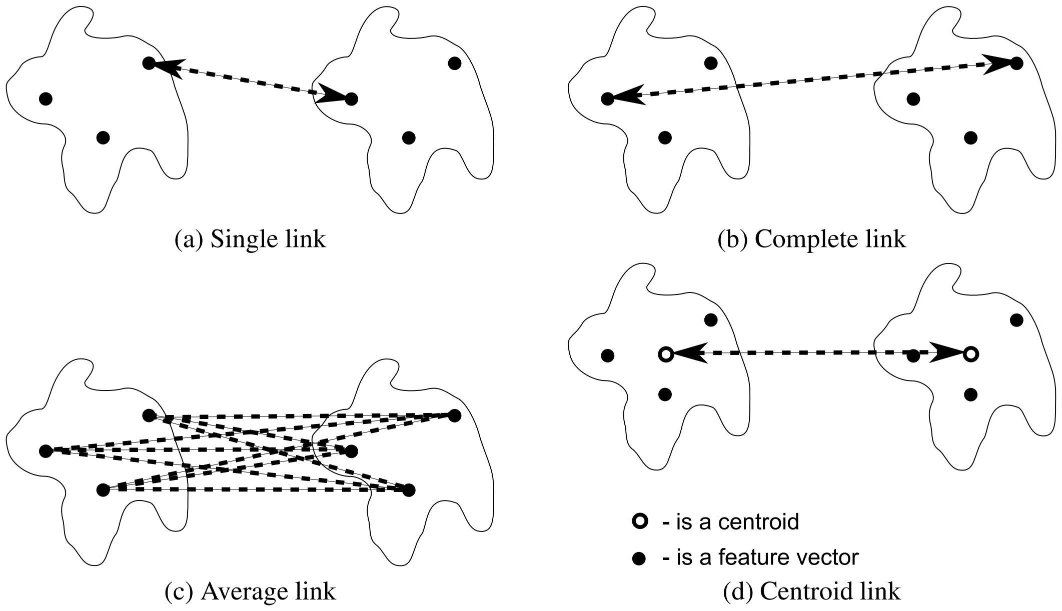


Figure 2.1 Linking Techniques in Hierarchical Clustering

In *agglomerative* hierarchical clustering, each feature vector in the data set initially represents a cluster [96]. In each iteration, *similar* clusters are merged. The process continues until only one cluster is left [96]. Thus, *agglomerative* algorithms follow a bottom-up approach generating a tree-like structure known as a *dendogram*. A *dendogram* shows which clusters were merged in each layer of the tree, i.e. each layer in the dendogram is equivalent to an iteration representing a partitioning of the data set. The root node of the tree consists of one cluster and each leaf node of the tree represents a feature vector.

In the case of *divisive* hierarchical clustering, a top-down approach is followed where all feature vectors are initially assigned to a single cluster as the root node. In each iteration, clusters containing the most *dissimilar* feature vectors are split. The process continues until each feature vector represents a cluster as a leaf node in the *dendogram*.

There are different *linking* techniques to determine the two most *similar* clusters. Each of these techniques makes use of a proximity matrix containing the pairwise *similarities* between clusters [96]. Since the different types of *linking* techniques are applicable to both *divisive* and *agglomerative* hierarchical clustering, the remainder of this section focuses on *agglomerative* hierarchical clustering. The most popular and familiar *linking* techniques are [56, 96]:

- **Single link:** Also known as the *nearest-neighbour method* [56], the *similarity* between two clusters is measured as the minimum distance between two feature vectors, one from each cluster, i.e.

$$\ell_{single}\left(C_i, C_j\right) = \min_{\forall \mathbf{p}\in C_i, \forall \mathbf{q}\in C_j} \{\sigma(\mathbf{p}, \mathbf{q})\} \tag{2.12}$$

where σ is a similarity measure, C_i and C_j are the i^{th} and j^{th} clusters respectively. A drawback of the single link technique is that the formed clusters are stretched out, i.e. a *chaining* effect [56, 96]. *Chaining* occurs when two clusters with highly dissimilar elements in each cluster are merged due to single elements being similar. Figure 2.1(a) illustrates the single link technique.

- **Complete link:** Also known as the *furthest-neighbour method* [56], the *similarity* between two clusters is measured as the maximum distance between two feature vectors, one from each cluster, i.e.

$$\ell_{complete}\left(C_i, C_j\right) = \max_{\forall \mathbf{p}\in C_i, \forall \mathbf{q}\in C_j} \{\sigma(\mathbf{p}, \mathbf{q})\} \tag{2.13}$$

  The complete link technique generates compact clusters [56, 96]. Figure 2.1(b) illustrates the complete link technique.

- **Average link:** The *similarity* between two clusters is measured as the average distance between all feature vectors from within the two clusters, i.e.

$$\ell_{average}\left(C_i, C_j\right) = \frac{1}{|C_i|\,|C_j|} \sum_{\forall \mathbf{p}\in C_i, \forall \mathbf{q}\in C_j} \sigma(\mathbf{p}, \mathbf{q}) \tag{2.14}$$

  The two clusters with the lowest $\ell_{average}$ value are merged into one cluster [56]. Figure 2.1(c) illustrates the average link technique.

- **Centroid link:** The distance between two centroids of different clusters can also measure the *similarity* between two clusters, i.e.

$$\ell_{centroid}\left(C_i, C_j\right) = \sigma\left(\mathbf{c}_i, \mathbf{c}_j\right) \tag{2.15}$$

  where $\mathbf{c}_i$ and $\mathbf{c}_j$ are the centroids of clusters C_i and C_j, respectively. The centroid of a cluster is defined in equation (2.18). The two clusters with the lowest $\ell_{centroid}$ value are merged into one cluster. Figure 2.1(d) illustrates the centroid link technique.

Hierarchical clustering algorithms do not have a pre-specified number of clusters. The number of clusters can be determined at any level of the *dendogram* or can be based on a *similarity* threshold [96]. Hierarchical clustering algorithms also make no assumption of the distribution of the data set, i.e. the algorithms are independent of the initial conditions [56].

There are, however, a few drawbacks to the hierarchical approach to clustering data. Hierarchical clustering algorithms are not suitable to cluster data with overlapping clusters, or data that consists of clusters with varying shapes, sizes and/or densities [56]. Hierarchical clustering algorithms are also not suitable for very large data sets, since the proximity matrix of pairwise *similarities* does not scale well with large data sets. Once two clusters are merged, feature vectors assigned to a cluster cannot be re-assigned to a different cluster. Therefore hierarchical clustering algorithms are static and merged clusters cannot be separated [56].

2.3.2 Partitional Clustering

Partitional clustering algorithms *partition* feature vectors in a data set into a number of non-hierarchical clusters. Partitioning of these feature vectors optimises a specific objective function [96]. The objective function is optimised such that the *inter-cluster* distance is maximised and the *intra-cluster* distance minimised. The *inter-cluster* distance measures the average separation between the centroids of all possible pairs of clusters and is calculated as

$$J_{inter} = \frac{2}{K \times (K-1)} \sum_{k=1}^{K-1} \sum_{j=k+1}^{K} \sigma\left(\mathbf{c}_k, \mathbf{c}_j\right) \tag{2.16}$$

A larger J_{inter} value indicates a higher average separation between cluster centroids, whereas a smaller value indicates a lower separation between cluster centroids. The *intra-cluster* distance measures the *compactness* of the clusters and is calculated as

$$J_{intra} = \frac{\sum_{k=1}^{K} \sum_{\forall \mathbf{p} \in C_k} \sigma\left(\mathbf{p}, \mathbf{c}_k\right)}{|P|} \tag{2.17}$$

J_{intra} calculates the average of all the distances between each feature vector and the cluster centroid with which the feature vector is associated. Thus larger distances between the feature vectors and the associated cluster centroids will indicate less compact clusters and vice versa. Partitional clustering algorithms can be *exclusive*, *overlapping* or *probabilistic*. Each of these categories is explained next.

Exclusive Clustering: Also known as *crisp* or *hard* clustering, a feature vector is only grouped with a single cluster. The most familiar algorithm in this category is the iterative K-means clustering algorithm [52]. K-means initialises K centroids, where K is the number of clusters into which a data set is partitioned. Based on a similarity measure, each feature vector in the data set is then assigned to only one of these centroids. A feature vector, $\mathbf{p}$, is assigned to a centroid, $\mathbf{c}$, if $\mathbf{p}$ is most similar to $\mathbf{c}$. Thus the subset of feature vectors assigned to a centroid forms a cluster.

After each feature vector in the data set is assigned to a centroid, the centroid of each cluster is recalculated according to the feature vectors assigned to the cluster. Algorithm 2.1 lists the pseudo code of a basic K-means algorithm [96].

Algorithm 2.1: Basic K-means

```
Randomly initialise $K$ centroids;
while some stopping condition(s) not true do
    for each feature vector $\mathbf{p}_i \in P$ do
        Calculate the similarity between $\mathbf{p}_i$ and $\mathbf{c}_k, k = 1, \ldots, K$;
        Assign $\mathbf{p}_i$ to centroid $\mathbf{c}_k$ with which $\mathbf{p}_i$ has the highest similarity;
    end
    Recalculate the centroid of each cluster;
end
```

The *similarity* between a feature vector, $\mathbf{p}_i$, and a centroid, $\mathbf{c}_k$, is calculated using the Euclidean distance measure as defined in equation (2.3). Thus a lower value of σ implies a higher similarity. The centroid (mean), $\mathbf{c}_k$, of cluster, C_k is calculated as

$$\mathbf{c}_k = \frac{1}{|C_k|} \sum_{\forall \mathbf{p} \in C_k} \mathbf{p} \tag{2.18}$$

The K-means algorithm optimises the *sum of squared distances* [70] as objective function by minimising the *intra-cluster* distance. The *sum of squared distances* is defined as [96]

$$J_{SSE} = \sum_{k=1}^{K} \sum_{\forall \mathbf{p} \in C_k} \sigma(\mathbf{p}, \mathbf{c}_k)^2 \tag{2.19}$$

The J_{SSE} determines the clustering quality of the clustered data set.

The *stopping criteria* for K-means can be one of the following [26, 96]:

- when there is no change in the centroids,
- there is minimal reassignment of feature vectors to different centroids,

- the J_{SSE} is small enough or there is a minimal decrease in J_{SSE}, or
- a specified number of iterations have been reached.

Although K-means is a very simple clustering algorithm, it has a few drawbacks. Since K-means minimises the sum of squared errors, the algorithm is susceptible to *outliers* in a data set which inflate the J_{SSE} [81]. *Outliers* can be removed, but in cases where the data is dynamic, *outliers* might indicate a change in the data. *Outlier* analysis is discussed in section 2.6. Since the centroids are randomly initialised, each run of the K-means algorithm delivers different clustering results. Thus, the random initialisation of K cluster centroids also determines the clustering quality [17].

An enhancement to K-means is the bisecting K-means which is less susceptible to the initialisation of K centroids, since all feature vectors are initially grouped into one cluster [156]. Predicting the correct number of K clusters also influences the clustering quality [71]. The centroids can be initialised by randomly selecting K feature vectors from the data set. This is known as the K-medoids algorithm [107]. The most centrally located feature vector in a cluster is that cluster's *medoid*. Thus the objective of K-medoids is to find the optimal *medoids* in a data set.

Overlapping Clustering (also known as *fuzzy* clustering): A feature vector is grouped with all clusters to a certain degree of membership [188]. The most familiar algorithm in this category is the Fuzzy C-means clustering algorithm, which is explained next [16]. The Fuzzy C-means algorithm initialises a membership matrix, $\mathbf{M}^{I\times K}$, where I is the number of feature vectors in data set P, and K is the number of clusters (centroids) [56]. Thus, an element m_{ik} of matrix $\mathbf{M}$, is the degree of membership of a feature vector $\mathbf{p}_i$ to the centroid $\mathbf{c}_k$ of cluster C_k. m_{ik} satisfies the following constraints:

- $m_{ik} \in [0,1], \quad i = 1,\ldots,I \quad \text{and} \quad k = 1,\ldots,K;$
- $0 < \sum_{i=1}^{I} m_{ik} < I, \qquad k = 1,\ldots,K$, i.e. no empty clusters are allowed and no cluster may contain all feature vectors; and
- $\sum_{k=1}^{K} m_{ik} = 1, \qquad i = 1,\ldots,I.$

The degree of membership, m_{ik}, is defined as [56, 70]

$$m_{ik} = \frac{\left[\frac{1}{\sigma(\mathbf{p}_i,\mathbf{c}_k)^2}\right]^{\frac{1}{\phi-1}}}{\sum_{k=1}^{K}\left[\frac{1}{\sigma(\mathbf{p}_i,\mathbf{c}_k)^2}\right]^{\frac{1}{\phi-1}}} \tag{2.20}$$

where ϕ is the weighting exponent ($\phi \geq 1$) which controls the degree of *fuzziness* of the resulting clusters [56]. Thus, a higher value of ϕ increases the *fuzziness* of the algorithm. The centroid, $\mathbf{c}_k$, of cluster C_k is calculated as [56, 70]

$$\mathbf{c}_k = \frac{\sum_{i=1}^{I} (m_{ik})^{\phi} \mathbf{p}_i}{\sum_{i=1}^{I} (m_{ik})^{\phi}} \tag{2.21}$$

Algorithm 2.2 provides pseudo code for the Fuzzy C-means algorithm [56].

Algorithm 2.2: Fuzzy C-means

```
Randomly initialise $K$ centroids;
Initialise matrix $\mathbf{M}$ by calculating $m_{ik}$ as defined in equation (2.20);
repeat
    Recalculate the centroid of each cluster using equation (2.21);
    Update the degree of memberships $m_{ik}$ with $m'_{ik}$, which is calculated using
    equation (2.20);
until $\max_{ik} \left\{ \left\| m_{ik} - m'_{ik} \right\| \right\} < \varepsilon$;
```

The objective function optimised by the Fuzzy C-means algorithm, is defined as [56, 70]

$$J_{FCM}(\mathbf{M}, C) = \sum_{i=1}^{I} \sum_{k=1}^{K} m_{ik}^{\phi} \sigma(\mathbf{p}_i, \mathbf{c}_k)^2 \tag{2.22}$$

where C is the set of K centroids and $\mathbf{M}$ is the matrix of membership degrees. Since Fuzzy C-means assigns a feature vector to a centroid with a certain degree of membership, the application of Fuzzy C-means is more realistic than K-means, because feature vectors tend to overlap. Similar to the K-means algorithm, the number of clusters needs to be specified. Fuzzy C-means may also converge to local optima [96].

Probabilistic Clustering: Probabilistic models assume that feature vectors are generated from different distributions, i.e. K clusters in a data set implies K different and unknown distributions in the data set [15, 56]. Thus, the data set consists of a mixture of density functions, one for each cluster [15, 21]. The probability density of the data is the sum of all the individual densities and is defined as [171]

$$G(\mathbf{p}; \Xi) = \sum_{k=1}^{K} \chi_k g(\mathbf{p}; \xi_k) \tag{2.23}$$

where g is a probability density function with parameters ξ_k. Ξ is the set of distribution parameters for each cluster, i.e. $\Xi = \{\xi_1, \ldots, \xi_k, \ldots, \xi_K\}$. Let g be the Gaussian density function, then $\xi_k = (\chi_k, \Omega_k, \mathbf{Z}_k)$ where Ω_k is the mean vector and $\mathbf{Z}_k$ the covariance matrix for the distribution of cluster k [15, 171]. The parameter χ in equation (2.23) is known as the *mixing* probability parameter [21, 171], and is the probability that feature vector $\mathbf{p}$ is generated from distribution k. Thus G is a mixture of Gaussian distributions with an unknown set of parameters, Ξ [21, 171]. The maximum likelihood method is a statistical technique to find Ξ [21]. Thus, the objective function that is optimised is defined as [21]

$$J_{EM}(\Xi) = \sum_{i=1}^{I} \log \left[\sum_{k=1}^{K} \chi_k g(\mathbf{p}_i; \xi_k) \right] \tag{2.24}$$

The above equation is optimised by the expectation maximisation (EM) algorithm [41]. The EM algorithm consists of an *expectation* step followed by a *maximisation* step in each iteration [21].

Algorithm 2.3 gives basic pseudo code for optimising equation (2.24) using EM. The algorithm stops when there is a small change in equation (2.24), which indicates that EM converged. There are a few drawbacks to the Gaussian Mixture model which are [56, 71]:

- the number of clusters needs to be specified,
- it is assumed that all clusters have a Gaussian distribution, and
- EM depends on the initial estimate of ξ.

2.3.3 Other Clustering Methods

Another clustering method is spectral clustering which is based on spectral graph theory [8, 143]. The patterns in a data set which need to be partitioned are represented as vertices and linked with weighted edges to form a connected graph. The connected graph can also be presented as a matrix of the distances between the patterns in the data set. The spectral clustering algorithm searches through the graph for edges which need to be pruned (or cut). The pruning of edges in the graph delivers a number of disjointed sub-graphs. Pruning is done in such a way that the similarities between vertices of the same sub-graph are higher than the pruned edge between two sub-graphs. The minimum-, ratio- or normalised-cut measures can be used to determine which edges need to be pruned [65, 100, 143]. Although spectral clustering can generate arbitrary-shaped clusters, there are two drawbacks to spectral clustering which are:

Algorithm 2.3: Basic Gaussian Mixture using EM

```
Set the number of iterations $t = 0$;
Estimate the initial values for $\xi_k^{(t)} = \left(\chi_k^{(t)}, \Omega_k^{(t)}, \mathbf{Z}_k^{(t)}\right)$;
repeat
    Calculate the expected values of the unknown data (expectation step) using
        $$\chi^{(t)}(k|\mathbf{p}_i) = \frac{\chi_k^{(t)} g\left(\mathbf{p}_i; \xi_k^{(t)}\right)}{\sum_{k=1}^{K} \chi_k^{(t)} g\left(\mathbf{p}_i; \xi_k^{(t)}\right)} \quad (2.25)$$
    Calculate a new estimate for $\xi_k^{(t+1)}$ (maximisation step) using
        $$\Omega_k^{(t+1)} = \frac{\sum_{i=1}^{I} \chi^{(t)}(k|\mathbf{p}_i)\mathbf{p}_i}{\sum_{i=1}^{I} \chi^{(t)}(k|\mathbf{p}_i)} \quad (2.26)$$
        $$\mathbf{Z}_k^{(t+1)} = \frac{\sum_{i=1}^{I} \chi^{(t)}(k|\mathbf{p}_i)\left(\mathbf{p}_i - \Omega_k^{(t+1)}\right)^T\left(\mathbf{p}_i - \Omega_k^{(t+1)}\right)}{\sum_{i=1}^{I} \chi^{(t)}(k|\mathbf{p}_i)} \quad (2.27)$$
        $$\chi_k^{(t+1)} = \frac{1}{I}\sum_{i=1}^{I} \chi^{(t)}(k|\mathbf{p}_i) \quad (2.28)$$
    $t = t + 1$;
until $\left(J^{t+1} - J^t\right) < \varepsilon$;
```

- the method is computationally expensive, and
- the clustering performance is influenced by a user-specified kernel width parameter.

A common drawback of the discussed clustering methods in the previous section is that these methods have difficulty in identifying non-convex clusters. A solution to partitioning data with non-convex clusters is to change the set of feature vectors used to represent the data using a kernel method. A kernel function projects the feature vectors in a data set to a higher dimension where the feature vectors are linearly separable for partitioning. A well-known kernel-based clustering method is the Support Vector Machine (SVM). SVM is a binary classifier that constructs a linearly separating hyperplane between feature vectors of two classes. The hyperplane separates the feature vectors in such a way that the distance between the hyperplane and the feature vectors nearest to the hyperplane are maximised. SVM is repetitively executed for multi-class data sets. In the context of clustering data with non-convex clusters, the feature vectors in the data set are transformed with a non-linear kernel function into a higher dimensional space which is linearly separable. SVM is then used to construct the hyperplanes (boundaries) between the transformed feature vectors. The initial feature vectors in the data set are then labelled according to the identified boundaries of the clusters in the data set.

2.4 Cluster Quality Validation

Since the identified number of groups (clusters) and the partitioning of data patterns between these groups may differ among different clustering algorithms, the quality of the partitioning needs to be evaluated, i.e. cluster validation quantitatively evaluates the clustering result of a clustering algorithm [170]. The different cluster validity measures are categorised into three criteria [67]:

- *internal* criteria - an example of this criteria is when a proximity matrix is used to evaluate the clustering results,
- *external* criteria - when an expected clustering result is pre-specified and the clustering results are evaluated against the expected clustering result, and
- *relative* criteria - clustering results are compared to other clustering schemes which are obtained by different input parameter values to the same algorithm.

A challenge in data clustering is to determine the optimal number of clusters in the data set. A drawback of the first two criteria to determine the optimal number of clusters is the statistical testing with high computational cost and the pre-specified clustering expectation. An approach to validate the number of clusters formed is to visually present the clustering results. In multi-dimensional problems where the number of dimensions is greater than three, visualisation of the formed clusters becomes difficult [67, 119].

Another approach to determine the optimal number of clusters is to execute the clustering algorithm multiple times, each time with a different number of clusters and validating the clustered data set with a cluster validity index, i.e. *relative* criteria. The cluster validity index is then plotted as a function of the number of clusters obtained for each execution of the algorithm. The number of clusters generated from the input parameters with the highest (or lowest) cluster validity index is then selected as the optimal number of clusters [151, 170]. This section discusses some of the most familiar cluster validity indices, which form part of the *relative* criteria to evaluate the partitioning quality of a clustering algorithm.

Dunn's index: The cluster validity index of Dunn [44] identifies clusters which are well separated and compact. Large values of the index imply well separated and compact clusters. Dunn's index is calculated as [66]

$$Q_D(K) = \min_{k=1,\ldots,K} \left\{ \min_{j=k+1,\ldots,K} \left\{ \frac{\sigma'\left(C_k, C_j\right)}{\max_{k=1,\ldots,K} \{\upsilon(C_k)\}} \right\} \right\} \tag{2.29}$$

where $\sigma'\left(C_k, C_j\right)$ is the dissimilarity between two clusters defined as

$$\sigma'\left(C_k, C_j\right) = \ell_{single}\left(C_k, C_j\right) \tag{2.30}$$

and $\upsilon(C_k)$ is the diameter of cluster C_k defined as

$$\upsilon(C_k) = \max_{\forall \mathbf{p},\mathbf{q} \in C_k} \{\sigma(\mathbf{p}, \mathbf{q})\} \tag{2.31}$$

where σ is the Euclidean distance as defined in equation (2.3) and ℓ_{single} is defined in equation (2.12). A data set with well-separated clusters has large inter-cluster distances as well as small intra-cluster distances for compact clusters. Thus, from the above Dunn-index definition, inter-cluster distances, σ', are maximised and intra-cluster distances, υ, minimised to maximise the value of Q_D. The maximum Q_D index value for a specific value of K indicates the optimal clustering of the data set. Problems with the Q_D index listed in [66] are that it is

- computationally complex, and
- sensitive to noise in the data set (noise increases the value of υ).

Net Information Gain index (NIG): An enhancement to the Dunn-index is the net information gain (NIG) validity index [103]. NIG measures the information change between clusters when a new cluster is introduced. NIG is applicable to clustering algorithms which are executed multiple times to determine the optimal number of clusters [103]. Initially all feature vectors in the first execution of the algorithm form a single cluster, i.e. execution E_1 has one cluster, C_1. With each execution of the algorithm, E_i, the data set, P, is re-clustered into i clusters using a clustering algorithm like K-means, i.e. $P = \cup_{k=1}^{i} C_k^i$. The *migration* of feature vectors between clusters from execution E_i to E_{i+1} forms the base for cluster quality measurement using NIG. Three different feature vector *migration* types are defined in [103] and discussed next. Let C_k^i be the k^{th} cluster in execution E_i, i.e. $1 \leq k \leq i$. Then, migration types are defined as:

1. *stagflation* - feature vectors forming a single cluster C_k^i in execution E_i continue to be part of that cluster in execution E_{i+1}, i.e. $C_j^{i+1} = C_k^i$ where $1 \leq j \leq (i+1)$.

2. *leakage* - a few feature vectors forming part of cluster C_k^i in execution E_i can be grouped with other feature vectors to form a different cluster C_j^{i+1} in execution E_{i+1} where $1 \leq j \leq (i+1)$.

3. *disassociation* - feature vectors forming part of a cluster C_k^i in execution E_i, divide into two or more smaller clusters in execution E_{i+1}, i.e. $\cup_{j=1}^{J} C_j^{i+1} \subseteq C_k^i$, where J is the number of clusters evolved from cluster C_k^i and $2 \leq J \leq (i+1)$.

When a cluster C_k^i divides into more than two clusters, the two most dominant clusters in execution E_{i+1} are selected to calculate the information change on cluster C_k^i. The two most dominant clusters are those clusters containing the most and second most number of migrated feature vectors from cluster C_k^i, respectively. The information gain/loss on cluster C_k^i from execution E_i to execution E_{i+1} is calculated as

$$inf\left(C_k^i\right) = d\left(C_k^i\right) \times M\left(C_k^i\right) \tag{2.32}$$

where $d\left(C_k^i\right)$ is the direction of the magnitude of change in information. The magnitude of change, $M\left(C_k^i\right)$, measures the *migration* of feature vectors from cluster C_k^i, defined as [103]

$$M\left(C_k^i\right) = -\sum_{j=1}^{J} p_j \ln p_j \tag{2.33}$$

where J is the number of clusters to where feature vectors of cluster C_k^i migrate to and p_j is the fraction of feature vectors migrating from cluster C_k^i to cluster C_j^{i+1}. If migrated feature vectors in cluster C_j^{i+1} overlap with feature vectors in cluster C_k^i then the direction of change $d\left(C_k^i\right) = -1$, i.e. *information loss*. If migrated feature vectors in cluster C_j^{i+1} are well separated from patterns in cluster C_k^i then the direction of change $d\left(C_k^i\right) = 1$, i.e. *information gain*. The centroid diameter and centroid linkage are used to measure the overlap between clusters [103]. The centroid diameter of a cluster C_k is defined as [103]

$$\upsilon\left(C_k\right) = 2\left[\frac{\sum_{\forall \mathbf{p} \in C_k} \sigma\left(\mathbf{p}, \mathbf{c}_k\right)}{|C_k|}\right] \tag{2.34}$$

where $|C_k|$ is the number of feature vectors in cluster C_k, $\mathbf{c}_k$ is the centroid of cluster C_k, and σ is a distance measure. The centroid linkage between two clusters C_k and C_j is defined in equation (2.15) [103]. The direction of change for cluster C_k^i from execution E_i to execution E_{i+1} is defined as

$$d\left(C_k^i\right) = \begin{cases} 1 & \text{if} \quad \ell_{centroid}\left(C_k^i, C_j^{i+1}\right) \geq \frac{1}{2}\left[\upsilon\left(C_k^i\right) + \upsilon\left(C_j^{i+1}\right)\right] \\ -1 & \text{otherwise} \end{cases} \tag{2.35}$$

The net information gain between two executions E_i and E_{i+1} is calculated as

$$NIG_{i+1} = \sum_{k=1}^{i} inf\left(C_k^i\right) \tag{2.36}$$

The total information content of the i^{th} execution is calculated as the cumulative sum of NIG's over all executions prior to and including execution i, defined as

$$Q_{NIG} = \sum_{g=0}^{i} NIG_g \tag{2.37}$$

The execution of the algorithm with the largest Q_{NIG} index value is considered as the execution with optimal clustering.

C-index: Let $\mathcal{E}$ be the ascending ordered set of distances between all possible pairs of feature vectors in data set P, i.e. $|\mathcal{E}| = \frac{|P|\times(|P|-1)}{2}$. Let $\mathcal{S}$ be the sum of m feature vector pair distances, where each feature vector pair is of the same cluster, i.e. $\mathbf{p},\mathbf{q} \in C_k$ where $\mathbf{p}$ and $\mathbf{q}$ are a pair in cluster C_k such that $\mathbf{p} \neq \mathbf{q}$. Then the C-index [88] is calculated as [19]

$$Q_C = \frac{\mathcal{S} - \mathcal{S}_{min}}{\mathcal{S}_{max} - \mathcal{S}_{min}} \tag{2.38}$$

In the above definition of Q_C, $\mathcal{S}_{max}$ and $\mathcal{S}_{min}$ are defined as

$$\mathcal{S}_{min} = \sum_{i=1}^{m} e_i \tag{2.39}$$

$$\mathcal{S}_{max} = \sum_{i=|\mathcal{E}|-m+1}^{|\mathcal{E}|} e_i \tag{2.40}$$

where $e_i \in \mathcal{E}$ and $m \geq 1$. Thus, S_{min} and S_{max} are the sum of the m smallest and m largest distances between feature vector pairs in P, respectively. The denominator in the definition of Q_C normalises the index value such that $Q_C \in [0,1]$. Smaller values of Q_C imply clusters of better quality. The optimal number of clusters minimises the Q_C index. The C-index is a suitable validity index for clusters of similar sizes.

Davies-Bouldin index: Davies and Bouldin (DB) proposed an index that measures the average similarity between each cluster and the cluster most similar to it [31]. The DB-index is calculated as [68]

$$Q_{DB} = \frac{1}{K}\sum_{k=1}^{K} \max_{\substack{j=1,\ldots,K \\ j\neq k}} \left\{ \frac{\frac{1}{2}\upsilon\left(C_k\right) + \frac{1}{2}\upsilon\left(C_j\right)}{\sigma\left(\mathbf{c}_k, \mathbf{c}_j\right)} \right\} \tag{2.41}$$

where K is the number of clusters, σ is the Euclidean distance as defined in equation (2.3), υ is the cluster centroid diameter as defined in equation (2.34), and $Q_{DB} \in [0,\infty)$. In the above definition, Q_{DB} has a small value when the distance between centroids $\mathbf{c}_k$ and $\mathbf{c}_j$ is large and the corresponding clusters C_k and C_j of these centroids are compact. Thus, an optimal number of K clusters minimises the value of Q_{DB}.

Halkidi-Vazirgiannis index: The *S_Dbw* index proposed by Halkidi and Vazirgiannis is calculated as [69]

$$Q_{S\_Dbw}\left(K\right) = Scat\left(K\right) + Dens\_bw\left(K\right) \tag{2.42}$$

The *S_Dbw*-index is defined as the summation of the average scattering (compactness, i.e. *intra-cluster variance*) of the clusters and the density among the clusters (separation, i.e. *inter-cluster density*). $Scat\left(K\right)$ is the average scattering of K clusters and $Dens\_bw\left(K\right)$ is the density among the K clusters. $Scat\left(K\right)$ is defined as [69]

$$Scat\left(K\right) = \frac{1}{K}\sum_{k=1}^{K} \frac{\psi\left(C_k, \mathbf{c}_k\right)}{\psi\left(P, \mathbf{p}\right)} \tag{2.43}$$

where $\mathbf{p}$ is the centroid of data set P and ψ is the variance of a set of feature vectors, defined as [69]

$$\psi\left(V, \mathbf{v}_j\right) = \sqrt{\sum_{n=1}^{N} \left[\frac{1}{|V|}\sum_{i=1}^{|V|} \left(\mathbf{x}_{i,n} - \mathbf{v}_{j,n}\right)^2\right]^2} \tag{2.44}$$

where N is the dimensionality of feature vectors, $\mathbf{x} \in V$. Therefore, the average standard deviation of all clusters is defined as [69]

$$\iota = \frac{1}{K}\sqrt{\sum_{k=1}^{K} \psi\left(C_k, \mathbf{c}_k\right)} \tag{2.45}$$

$Dens\_bw\left(K\right)$ is defined as [69]

$$Dens\_bw\left(K\right) = \frac{1}{K \times (K-1)} \sum_{k=1}^{K} \sum_{\substack{j=1 \\ k \neq j}}^{K} \frac{density\left(\mathbf{u}_{kj}\right)}{\max\{density\left(\mathbf{c}_k\right), density\left(\mathbf{c}_j\right)\}} \tag{2.46}$$

where $\mathbf{u}_{kj}$ is the middle point of cluster centroids $\mathbf{c}_k$ and $\mathbf{c}_j$, and is calculated as $\mathbf{u}_{kj} = \frac{\mathbf{c}_k + \mathbf{c}_j}{2}$. The $density\left(\mathbf{u}_{kj}\right)$ of a feature vector $\mathbf{u}_{kj}$ calculates the number of feature vectors in the neighbourhood of vector $\mathbf{u}_{kj}$. In order to determine whether a feature vector is within the neighbourhood of another vector, the following neighbourhood function is defined [69]

$$n\left(\mathbf{p}_i, \mathbf{u}\right) = \begin{cases} 0 & \text{if} \quad \sigma\left(\mathbf{p}_i, \mathbf{u}\right) > \iota \\ 1 & \text{otherwise} \end{cases} \tag{2.47}$$

Thus, a feature vector $\mathbf{p}_i$ is within the neighbourhood of $\mathbf{u}$ if the distance from $\mathbf{u}$ is less than ι which is the average standard deviation of all clusters as defined in equation (2.45); ι is the radius of the neighbourhood. The $density\left(\mathbf{u}_{kj}\right)$ is then calculated as [69]

$$density\left(\mathbf{u}_{kj}\right) = \sum_{\forall \mathbf{p}_i \in \left\{C_k \cup C_j\right\}} n\left(\mathbf{p}_i, \mathbf{u}_{kj}\right) \tag{2.48}$$

A small value of $Scat\left(K\right)$ indicates compact clusters and a small value of $Dens\_bw\left(K\right)$ indicates well separated clusters [69]. Thus the number of clusters, K, that minimises the Q_{S_Dbw} index value is considered as the optimal number of clusters in the data set.

Ray-Turi index: Ray and Turi proposed a validity index which is based on the ratio of *intra-clustering* distance to *inter-clustering* distance [151]. The proposed index is calculated as [151]

$$Q_{ratio} = \frac{J_{intra}}{inter_{min}} \tag{2.49}$$

where J_{intra} is defined in equation (2.17), $inter_{min}$ is calculated as

$$inter_{min} = \min_{\substack{k=1,\ldots,K-1 \\ j=k+1,\ldots,K}} \left\{ \sigma\left(\mathbf{c}_k, \mathbf{c}_j\right) \right\} \tag{2.50}$$

and σ is the Euclidean distance as defined in equation (2.3). In the above definition of *intra*, the average *compactness* of the clusters is calculated by averaging over all the distances between each cluster's centroid and the feature vectors within that cluster. The definition of $inter_{min}$ simply calculates the smallest distance between the centroids of the clusters to determine the smallest separation between clusters. J_{intra} needs to be minimised and $inter_{min}$ needs to be maximised for more compact and more separated clusters. Thus, the defined ratio validity index, Q_{ratio}, needs to be minimised to have optimal clustering. Therefore the optimal number of clusters, K, minimises the value of Q_{ratio}.

Turi proposed a modification to the above ratio of *intra-clustering* distance to *inter-clustering* distance by multiplying the ratio with a Gaussian function of the number of clusters [173]. The modified index is calculated as [173]

$$Q_{RT} = Q_{ratio} \times \left[c \times g\left(\mu, \sigma\right) + 1\right] \tag{2.51}$$

where g is a Gaussian function with mean, μ, and standard deviation, σ and c is some constant. Function g is defined as

$$g\left(\mu, \sigma\right) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\left[-\frac{(K-\mu)^2}{2\sigma^2}\right]} \tag{2.52}$$

where K is the number of clusters. The Gaussian function penalises the ratio for small values of K in favour of larger values of K.

Other familiar cluster validity indices not covered in this section are among others Entropy, Purity and Silhouette. The interested reader is referred to [189] for more information.

2.5 Cluster Quality in Dynamic and Uncertain Environments

Section 2.4 gave an overview of different cluster quality measurements to quantitatively evaluate the clustering results of a clustering algorithm applied to stationary data sets, i.e. static environments. A static environment is defined as feature vectors in space which do not move to different spatial positions over time, i.e. the feature vectors are static and will remain at the same positions at any given point in time. The cluster validity indices form part of the *relative* criteria.

This section discusses different types of non-stationary environments and introduces the different performance measures applied to optimisation algorithms for problems in non-stationary environments, i.e. dynamic environments. These performance measures are used to quantify the quality of partitioning by clustering algorithms in dynamic environments (discussed in section 7.1). A dynamic environment in the context of this thesis is defined as feature vectors in space which move or adapt to different spatial positions over time [64].

The goal of optimisation algorithms (such as particle swarm optimisation (PSO) which is discussed in section 2.7.1) is to locate an optimum to an optimisation problem. There are different classes of optimisation algorithms [162]. For the purpose of this section, stochastic population based optimisation algorithms are considered where a population of candidate solutions, $A(t)$, is maintained at each time step, t. The fitness of each candidate solution, $\mathbf{a} \in A(t)$, is calculated using the objective function, f, that needs to be minimised or maximised. The candidate solution with the *best* fitness, *best* (t), is selected as the solution that best optimises the objective function, f, at a specific time step, t. An objective function can also change over time. These changes result in a dynamic search space with different optima at each point in time. Optimisation algorithms for dynamic environments need to track optima over time by detecting and tracking changes in the search space. The remainder of this section assumes maximisation of the objective function.

Changes in a dynamic environment can occur at any point in time with different effects to the optima of the objective function. Figure 2.2 illustrates the different types of dynamic environments for the following dynamic function:

$$f(\mathbf{x}, \omega(t)) = \begin{cases} \frac{\omega_1(t)}{8} \times \pi \times \exp\left[-\frac{1}{2} \times \left((x_1 + \omega_2(t))^2 + (x_2 + \omega_2(t))^2\right)\right] & \text{if } (x_1 < 0) \text{ and } (x_2 < 0) \\ \frac{\omega_1(t)}{4} \times \pi \times \exp\left[-\frac{1}{2} \times \left((x_1 - \omega_2(t))^2 + (x_2 + \omega_2(t))^2\right)\right] & \text{if } (x_1 > 0) \text{ and } (x_2 < 0) \\ \frac{\omega_3(t)}{4} \times \pi \times \exp\left[-\frac{1}{2} \times \left(x_1^2 + (x_2 - 5)^2\right)\right] & \text{if } \quad (x_2 > 0) \\ 0 \quad \text{otherwise} \end{cases} \tag{2.53}$$

where $\omega(t)$ is the control parameters which determine the magnitude of change in the dynamic environment at a specific time t. The different types of dynamic environments in [101, 180] are grouped into three main types [46]:

1. The locations of the optima change but the values of the optima remain the same (as illustrated in figure 2.2(b)).
2. The locations of the optima remain the same (no change) but the values of the optima change (as illustrated in figure 2.2(d)).
3. Both the locations and values of the optima change (as illustrated in figures 2.2(c) and 2.2(e)).

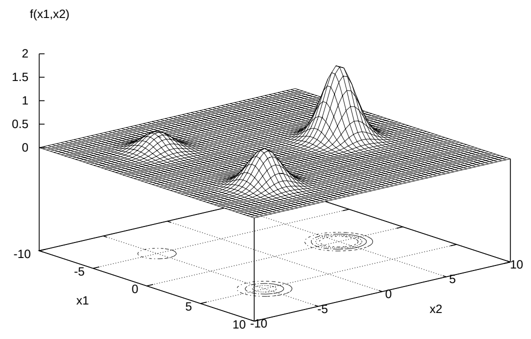
For each of these dynamic environment types the number of optima may change, in that new optima may appear and existing optima may disappear.

The cluster validity indices discussed in section 2.4 are based on two functions, namely *inter-* and *intra-error* (as defined in equations (2.16) and (2.17), respectively). The *inter-error* function needs to be maximised to obtain well separated clusters and the *intra-error* function needs to be minimised to obtain more compact clusters. Since the different cluster validity indices discussed in section 2.4 are either maximised or minimised to obtain the best partitioning of a data set, these indices can be used as fitness functions to achieve optimal clustering in dynamic environments.

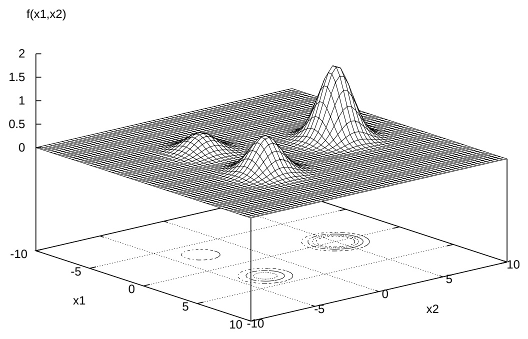
Thus, from a clustering perspective in a dynamic environment, the initial formed clusters of a data set can *adapt* over time, which means that at each time step the feature vectors in different clusters can follow different *migration* types to and from other clusters. These *migration* types were defined in section 2.4 as part of the discussion on the net information gain index (Q_{NIG}) [103]. The *migration* of feature vectors from one cluster to another implies that the centroids of the different clusters also move in space to different positions. Therefore centroids may move, disappear and/or new centroids may appear.

From the above list of dynamic environment types, the definition of clustering in section 2.1 and the different *migration* types discussed in section 2.4, the dynamic environments investigated in this thesis are defined as adapting feature vectors such that:

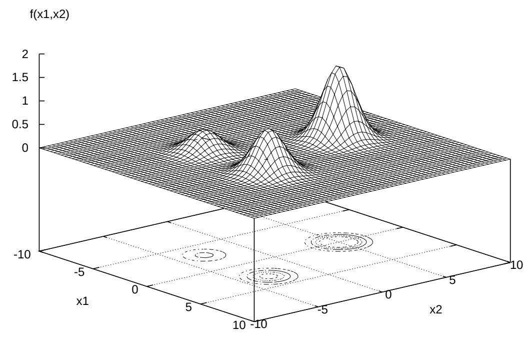
- the number of clusters remains static with the same centroids but the *compactness* of each cluster changes, i.e. movement of feature vectors within a static number of clusters (as illustrated in figure 2.3(b)).



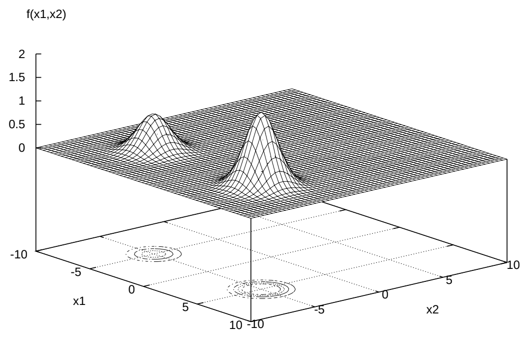
(a) Static function $\omega_1 = 1, \omega_2 = 5, \omega_3 = 2$



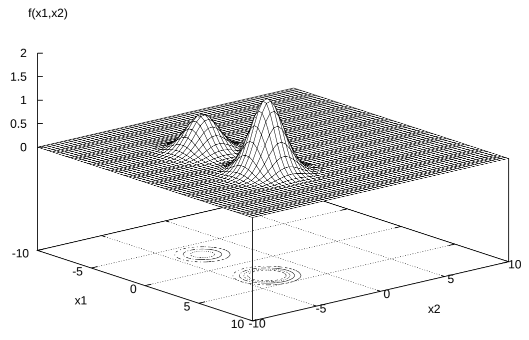
(b) Dynamic function $\omega_1 = 1, \omega_2 = 3, \omega_3 = 2$



(c) Dynamic function $\omega_1 = 1.2, \omega_2 = 3, \omega_3 = 2$

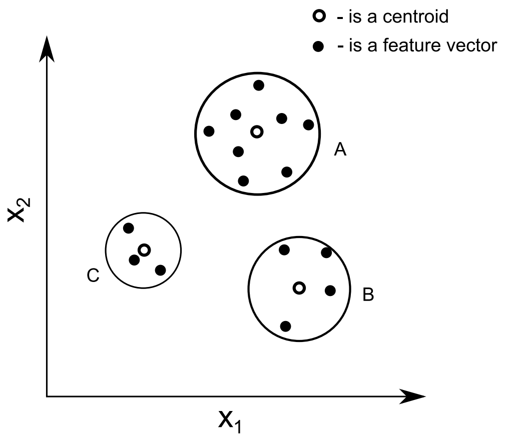


(d) Dynamic function $\omega_1 = 2, \omega_2 = 5, \omega_3 = 0$

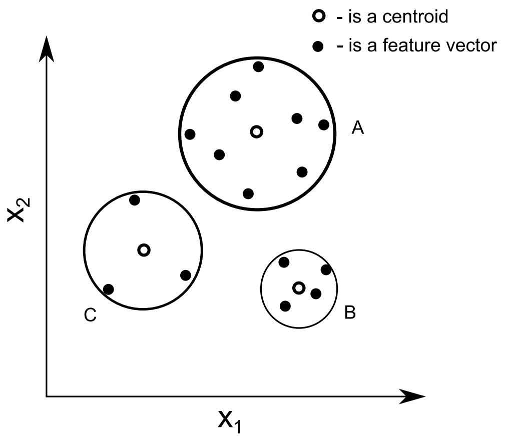


(e) Dynamic function $\omega_1 = 2, \omega_2 = 3, \omega_3 = 0$

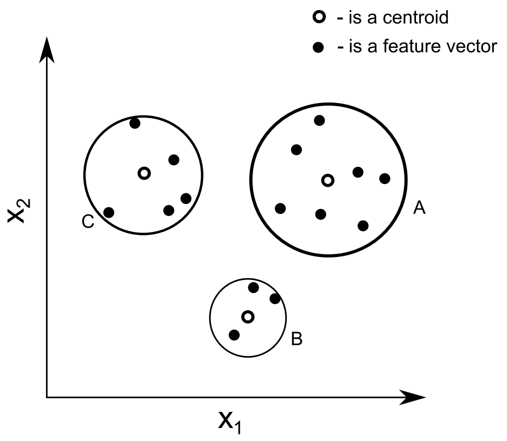
Figure 2.2 Dynamic Objective Function (equation (2.53))



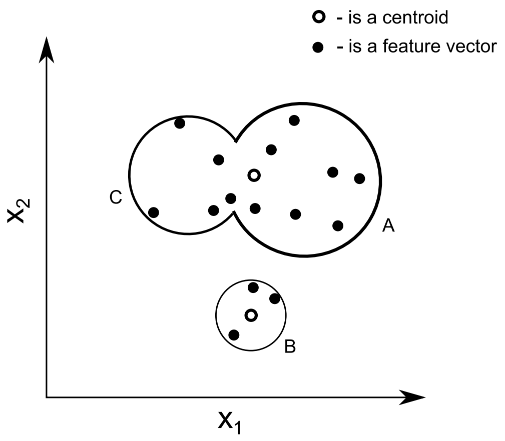
(a) Static environment - no change



(b) Dynamic environment - feature vectors move within a static number of clusters



(c) Dynamic environment - a static number of centroids move with migration of feature vectors between clusters



(d) Dynamic environment - moving centroids with merging/dividing clusters and migrating feature vectors

Figure 2.3 Clustering in Dynamic Environments

- the number of clusters remains static with changing centroids and the *compactness* of each cluster changes, i.e. movement of a static number of clusters as well as *migration* of feature vectors (as illustrated in figure 2.3(c)).

- the number of clusters changes, resulting in different centroids and a change in the *compactness* of each cluster, i.e. clusters *merge/divide* as a result of feature vectors *migrating* between clusters and/or moving centroids (as illustrated in figure 2.3(d)).

The remainder of this section discusses the different performance measures of an optimisation algorithm which inspired the definition of a performance measure for clustering algorithms in dynamic environments (discussed in section 7.1). Performance measures that can be used to quantify different aspects of the performance of optimisation algorithms at specific time intervals include (assuming maximisation of the objective function):

- **Accuracy:** This performance measure calculates the instantaneous accuracy of the best solution found by an optimisation algorithm at a certain time step, t. Feng [50] introduced an accuracy measure in static environments which can be calculated in dynamic environments as [180]

$$accuracy\,(t) = \frac{f\,(best\,(t)) - f_{min}\,(t)}{f_{max}\,(t) - f_{min}\,(t)} \tag{2.54}$$

  where f is the fitness function, $best\,(t)$ is the best candidate solution in the population at time step t and $f_{max}\,(t)$ and $f_{min}\,(t)$ are the maximum and minimum values of f in the search space at time step t, respectively, defined as:

$$f_{max}\,(t) = \max_{\forall \mathbf{x}(t)\in\Re^N} \{f\,(\mathbf{x}\,(t))\} \tag{2.55}$$

$$f_{min}\,(t) = \min_{\forall \mathbf{x}(t)\in\Re^N} \{f\,(\mathbf{x}\,(t))\} \tag{2.56}$$

  Note that $accuracy\,(t) \in [0,1]$, where an accuracy of one is the best possible value.

- **Stability:** An optimisation algorithm is defined to be *stable* if changes in the environment have a minor or no affect on the measured accuracy [180]. The stability of an optimisation algorithm at a certain time step, t, is calculated as [180]

$$stab\,(t) = \max\{0, accuracy\,(t-1) - accuracy\,(t)\} \tag{2.57}$$

  where $stab\,(t) \in [0,1]$. A $stab\,(t)$ value close to zero implies a high stability. If $stab\,(t) > 0$, then there is a difference in the accuracy between consecutive time steps. In the above calculation of *stability* (assuming maximisation), whenever $accuracy\,(t) > accuracy\,(t-1)$, the result is $stab\,(t) = 0$ which indicates that the optimisation algorithm is *stable*. This, however, is not true since there was an effect on the accuracy. The only time a $stab\,(t)$ value will have a deviation from zero is when the measured accuracy decreases (in the case of maximising the objective function). Whenever the measured accuracy improves due to a change in the environment, the $stab\,(t)$ value will remain zero and therefore give no indication of any change in the environment or the ability of the optimisation algorithm to track a moving optimum. Instead of measuring the *stability* of an optimisation algorithm, this thesis proposes that the *sensitivity* of the optimisation algorithm to a change in the environment can be measured using

$$sens(t) = |accuracy(t-1) - accuracy(t)| \tag{2.58}$$

where $sens(t) \in [0,1]$. A $sens(t)$ value close to zero indicates a less *sensitive* change in the environment whereas a $sens(t)$ value closer to one indicates a more *sensitive* change. For both $stab(t)$ and $sens(t)$, the calculated values give an indication of the ability of the optimisation algorithm to track a moving optimum.

- **Recovery:** An optimisation algorithm also needs to react to a changing environment [180]. The reaction of an optimisation algorithm to a change in the environment at time t is measured as the time taken to locate the moved optimum at time $t^{'}$. An optimisation algorithm succeeds in the location of the moved optimum when the ratio of the accuracy at time $t^{'}$ to the accuracy at time t is greater or equal to the specified accuracy threshold. The accuracy threshold is calculated as $1-\varepsilon$ where ε is the minimum ratio of accuracy for an optimisation algorithm to succeed in locating the moved optimum. The ε-*reactivity* of an optimisation algorithm at time t is calculated as [180]

$$react_{\varepsilon}(t) = \begin{cases} t^{'} - t | t < t^{'} \leq T, \quad t^{'} \in \mathbb{N} & \text{if } \frac{accuracy\left(t^{'}\right)}{accuracy(t)} \geq (1-\varepsilon) \\ T - t & \text{otherwise} \end{cases} \tag{2.59}$$

  where T is the total number of time steps and $\varepsilon \in [0,1]$. A smaller $react_{\varepsilon}$ value implies a higher reactivity.

A drawback to the above performance measures is that the maximum and minimum values of f at each time step t in the dynamic search space need to be known to determine accuracy, stability and reactivity [180]. The maximum and minimum fitness values might also change over time, due to the dynamic behaviour of the search space. This implies that the best fitness value at time t, might be the worst fitness value at time $t+1$ [131]. Thus, prior *knowledge* of the dynamic search space at each time step needs to be known to calculate the accuracy of the best solution found by an optimisation algorithm.

In cases where there is *no knowledge* of the dynamic search space, the above accuracy measure is inappropriate. In absence of information about the search space, the following accuracy measures can be used (assuming maximisation) [180]:

Current best fitness: The current best fitness is the most familiar accuracy measure for optimisation algorithms applied to dynamic environments, calculated as [180]

$$currentBest\left(t\right) = \max_{\forall \mathbf{a} \in A(t)} \left\{ f\left(\mathbf{a}\right) \right\} \tag{2.60}$$

where $f\left(\mathbf{a}\right)$ is the fitness of solution $\mathbf{a}$ in the population of candidate solutions A at time t. The *currentBest* performance measure calculates the fitness of the solution that best optimises the objective function at each time step.

Current best offline fitness: The current best offline fitness accuracy measure calculates the maximum current best accuracy up to a certain point in time [131, 180]. Thus, it measures the solution that best optimises the objective function over all time steps. The current best offline fitness accuracy measure is calculated as [180]

$$currentBestOff\left(t\right) = \max_{1 \leq t' \leq t} \left\{ currentBest\left(t'\right) \right\} \tag{2.61}$$

where *currentBest* is defined in equation (2.60). The *currentBestOff* is less suitable in dynamic environments since it irrationally compares *currentBest* measures at different steps in time at which a change in the environment could occur. Thus, measuring the accuracy of an optimisation algorithm over a period of time as the solution that best optimises the objective function at a certain time within that period, has no meaning in a dynamic environment.

Current average fitness: The current average fitness calculates the average accuracy of population A at time t, defined as [102, 180]

$$currentAvg\left(t\right) = \frac{1}{\left|A\left(t\right)\right|} \sum_{\forall \mathbf{a} \in A(t)} f\left(\mathbf{a}\right) \tag{2.62}$$

A drawback to the current average fitness accuracy measure is that if some of the solutions in population A diverge from the optimal solution due to a change in the environment, the average accuracy of population A will decrease even though other solutions in population A converge to the new optimum. In such a case, the current average fitness accuracy measure gives a deceptive view of the optimisation algorithm's ability (or inability) to track moving optima.

Window accuracy: The window accuracy assumes that the *best* fitness does not change within a certain time-span, thus the accuracy is only measured within a certain window size, W [131, 180]. The window accuracy measure is calculated as [180]

$$windowAcc(t) = \max_{\forall \mathbf{a} \in A(t)} \left\{ \frac{f(\mathbf{a}) - windowWorst}{windowBest - windowWorst} \right\} \tag{2.63}$$

where

$$windowBest = \max_{t-W \le t' \le t} \left\{ \max_{\forall \mathbf{a} \in A(t')} \{f(\mathbf{a})\} \right\} \tag{2.64}$$

$$windowWorst = \min_{t-W \le t' \le t} \left\{ \min_{\forall \mathbf{a} \in A(t')} \{f(\mathbf{a})\} \right\} \tag{2.65}$$

The assumption of no change of the *best* fitness is a risk and a disadvantage of the *windowAcc* measure [131, 180].

The measured performance of different optimisation algorithms can be compared by averaging each algorithm's performance measure at each step in time over the total running time T [102]. This will give each algorithm a mean value of measured performance which is used for comparison. An accuracy measure related to the above *currentBest* measure is proposed in [131], namely the collective mean fitness (cmf). The collective mean fitness takes the fitness trajectory across the entire dynamic landscape into account by averaging the mean value of measured performance over the number of independent runs, E, of the algorithm on the same problem. The collective mean fitness is defined as [131]

$$cmf = \frac{\sum_{r=1}^{E} \left(\frac{\sum_{t=1}^{T} f(best(t))}{T} \right)}{E} \tag{2.66}$$

where T is the total number of time steps, and $f(best(t))$ is the *best* fitness value at time t. The cmf is thus the sum of all average *best* fitness values, averaged over a number of runs. The number of time steps T needed by an optimisation algorithm is important in optimisation of a dynamic environment. A too small value of T, will result in an unstable value of cmf. A large value of T is necessary for the optimisation algorithm to be exposed to extreme changes in the dynamic environment [131].

Section 7.1 proposes a clustering performance measure to quantify the quality of partitioning by clustering algorithms in dynamic environments. The proposed clustering performance measure is derived from the above collective mean fitness measure and uses cluster validity indices.

As mentioned earlier, the cluster validity indices are based on the *inter-* and *intra-errors* to determine the cluster separation and cluster compactness, respectively. In the context of clustering of dynamic environments, these *errors* can be used, in addition to the proposed clustering performance measure in section 7.1, to quantify the quality of partitioning by clustering algorithms over time.

As an example, assume a fixed number of clusters, K. The average *inter-error* plotted against time, will increase in value if the clusters become more separated in time, i.e. clusters move away from one another. If there is any *migration* of feature vectors between clusters, it is expected that the average *intra-error* plotted against time will fluctuate from the time of *migration* until the feature vectors become stationary again. In clustering problems where K is not fixed, the validity indices determine the optimal partitioning into K clusters at a specific time, indicating whether a new cluster emerged or whether clusters merged.

2.6 Outlier Detection and Analysis

Referring to the definition of data clustering in section 2.1, each cluster (or centroid) represents a *concept* or *trend* in the data set. Based on a *similarity* measure, an *outlier* feature vector is either not grouped with any cluster or has a major deviation from the centroid of a cluster with which the *outlier* is associated. Therefore an *outlier* is also known as an *exception* and is defined as a vector which is not *similar* to any of the centroids. *Outliers* are grossly different from and/or inconsistent with feature vectors of the same data set [74], which can be a result of inherent data variability [74].

In the context of a dynamic environment a feature vector can only be classified as an *outlier* at a specific point in time. An *outlier* at time t might disappear at time $t+1$ or even more *outliers* might occur within the same area. The latter could indicate a new trend in the data for a certain period of time.

Outlier detection and analysis is referred to as *outlier mining* and is described as follows [186]: In a data set of I feature vectors, the expected number of outlier vectors, o, are those feature vectors which are the most *dissimilar*, *exceptional* and/or *inconsistent* compared to the remainder of the data set. Outlier detection can be categorised into three approaches, namely the statistical approach, distance-based approach and deviation-based approach [74, 186]. Each of these categories is briefly explained next.

Statistical approach: Feature vectors in a data set are identified as *outliers* with a *statistical discordancy* test by examining a *null* hypothesis and an *alternative* hypothesis. A *null* hypothesis can state that all feature vectors in a data set are from the same distribution. Thus, the *statistical discordancy* test verifies whether a feature vector is significantly large in relation to the distribution of the data set [74]. The *null* hypothesis is kept in case no statistical significant evidence supports the rejection thereof.

The *alternative* hypothesis states that a feature vector comes from a different distribution as the one defined in the *null* hypothesis [74]. The interested reader is referred to [74] for more information on the different *alternative* distributions.

A drawback of the statistical approach is the assumption of a specific distribution (like normal, Poisson etc.) for the data set and thus requires knowledge of the distribution parameters (like average, standard deviation etc.) and the expected number of outliers. Another major drawback is that the hypothesis testing is for outlier detection of single *features*, i.e. only a single attribute is tested in a feature vector. There is also no guarantee that all outliers are detected [74].

Distance-based approach: This approach is based on a distance measure between feature vectors in a data set. A global distance-based neighbourhood radius is defined for each feature vector, $\mathbf{p}_i$. If a fraction of the feature vectors is not within the distance radius of $\mathbf{p}_i$, then $\mathbf{p}_i$ is detected as an outlier in the data set [117]. The different distance-based outlier detection algorithms are the index-based, nested-loop and cell-based algorithms [74, 117]. A drawback to these distance-based algorithms is the user specified parameters of neighbourhood radius and the number of feature vectors in the data set that needs to be within the specified radius.

Deviation-based approach: The most general *concepts* can be derived from the feature vectors in a data set. Feature vectors which deviate from these general *concepts* are seen as outliers. There are two techniques in deviation-based outlier detection [74], namely sequential exception and the on-line analytical processing (OLAP) data cube technique. The first of these two techniques is discussed next and the interested reader is referred to [74] for more information on the OLAP technique:

- *Sequential exception technique:* The sequential exception technique is based on a process followed by humans to detect an outlier after being represented with a series of *similar* feature vectors [5]. An outlier is defined as a feature vector that deviates from the series.

  A sequence of subsets, $\{S_1, S_2, \ldots, S_o\}$, is built from a data set, P, consisting of I feature vectors, i.e. $2 \leq o \leq I$. Thus, $S_{o-1} \subset S_o : S_o \subseteq S$. A function of *dissimilarity* (not necessarily distance based) is calculated between each subset. The *dissimilarity* function is defined as any function that returns a low value to indicate more *similar* feature vectors and a high value to indicate less *similar* feature vectors [74, 186]. An example of a *dissimilarity* function is defined in equation (2.44), which calculates the variance of a set of feature vectors.

  A *smoothing factor* function is calculated for each subset, S_o, in the sequence. The subset with the highest *smoothing factor* becomes the set of outliers [5, 74]. The *cardinality* of each subset is used to scale the *smoothing factor*. The *cardinality* of a set is defined as the number of feature vectors in the set [5, 74]. The *smoothing factor* is calculated as [5]

$$sf(S_o) = |S_o - S_{o-1}| \times (D(S_o) - D(S_{o-1})) \tag{2.67}$$

  where $|\bullet|$ is the cardinality of the set, D is the function of dissimilarity, and the exception set S_e is defined as that set where [5]

$$sf(S_e) \geq sf(S_o) \quad \forall S_o \subset S \tag{2.68}$$

  Thus, the *smoothing factor* (sf), calculates the reduction in *dissimilarity* when removing a subset S_o of feature vectors from set S. If all feature vectors in S are *similar*, the *smoothing factor* is zero [5]. The exception set S_e has the highest sf value [5].

2.7 Alternative Computational Models for Clustering

This section discusses two alternative computational algorithms which can be applied to the problem of data clustering. Both of these algorithms implement neighbourhood topologies to influence the search behaviour of the algorithm. The model proposed in this thesis also utilises the concept of a neighbourhood topology. The two algorithms which are briefly discussed are the Particle Swarm Optimisation (PSO) algorithm, introduced by Kennedy and Eberhart [108, 109] and the Self-organising Feature Map (SOFM or SOM), introduced by Kohonen [118].

2.7.1 Particle Swarm Optimisation

Particle Swarm Optimisation (PSO) algorithms model the formation and social behaviour found in bird flocks [108, 109]. PSO is a population-based stochastic search algorithm [109]. Thus, PSO maintains a population or a swarm of *particles*. Each *particle* represents a potential solution to an optimisation problem. In PSO, particles 'fly' through a multi-dimensional space in search of the *optimal* or *best* solution. The *best* solution to an optimisation problem is the particle with the highest *fitness*. The *fitness* of a particle is usually a function of the *objective* that needs to be optimised.

The position of each particle is presented by a feature vector in a multi-dimensional space. A particle moves through the search space by adjusting its position towards its own *best* experienced solution and towards the *best* particle in the *neighbourhood*. Since a particle needs to be able to adjust towards its own *best* experienced solution, a particle needs to maintain its *personal best position*. The *neighbourhood* can either be the entire swarm of particles or a subset thereof. The former case is known as *gbest* PSO and the latter as *lbest* PSO. In addition to the feature vector and *personal best position* contained by a particle, a particle also maintains its *current velocity*.

The rest of this section uses the following notation:

- S^N: The swarm or population of particles in N-dimensional search space;
- $\mathbf{x}_i$: The current feature vector or position of the i-th particle in S^N;
- $\mathbf{b}_i$: The i-th particle's *personal best position*;
- $\mathbf{v}_i$: The current velocity of the i-th particle in S^N;
- $\mathbf{V}_{max}$: The maximum allowed velocity of any particle in S^N;
- $\mathbf{g}_i$: The position of the *best* particle in the *neighbourhood* of the i-th particle in S^N;
- f: The fitness function (objective that needs to be optimised).

The i-th particle's position is adjusted by using

$$\mathbf{x}_i(t+1) = \mathbf{x}_i(t) + \mathbf{v}_i(t+1) \tag{2.69}$$

where $\mathbf{v}_i(t+1)$ is the updated velocity of the particle at time step $t+1$. The velocity of the i-th particle is updated by using

$$\mathbf{v}_{i,n}(t+1) = w\mathbf{v}_{i,n}(t) + c_1 r_{1,n}(t)\left(\mathbf{b}_{i,n}(t) - \mathbf{x}_{i,n}(t)\right) + c_2 r_{2,n}(t)\left(\mathbf{g}_{i,n}(t) - \mathbf{x}_{i,n}(t)\right) \tag{2.70}$$

where w is the *inertia weight*, c_1 and c_2 are the acceleration constants, $r_{1,n}(t)$, $r_{2,n}(t) \sim U(0,1)$ and $n = 1,\dots,N$. The velocity update in equation (2.70) basically consists of three components. These are:

- *inertia*: With the *inertia weight*, w, a fraction of the particle's previous velocity contributes to the update [160]. Large values of w result in better exploration of the search space, whereas lower w values result in better exploitation of the search space [160].

- *social component*: This is the $\left(\mathbf{g}_{i,n}(t) - \mathbf{x}_{i,n}(t)\right)$ term, which is the *distance* in the n-th dimension to the *best* particle in a *neighbourhood*. The *best* particle $\mathbf{g}_i$ at time t, in a *neighbourhood* with radius d, is determined by using the following equation:

$$f(\mathbf{g}_i(t)) = min\left\{f(\mathbf{b}_{i-d}(t)), f(\mathbf{b}_{i-d+1}(t)), \dots, f(\mathbf{b}_i(t)), \dots, f(\mathbf{b}_{i+d}(t))\right\} \tag{2.71}$$

  If $d = \frac{|S|}{2}$, then the *neighbourhood* is the entire swarm of particles and the *best* particle at time t in the *neighbourhood* will be the same for all particles in the swarm. The resulting PSO is referred to as the *gbest* PSO. If $d < \frac{|S|}{2}$, then the *neighbourhood* is a subset of the swarm. The resulting PSO is referred to as the *lbest* PSO [160].

- *cognitive component*: This is the $\left(\mathbf{b}_{i,n}(t) - \mathbf{x}_{i,n}(t)\right)$ term, which is the *distance* in the n-th dimension to the *personal best position* of the i-th particle. The *personal best position* of the i-th particle at time t is updated by

$$\mathbf{b}_i(t+1) = \begin{cases} \mathbf{b}_i(t) & \text{if} \quad f(\mathbf{x}_i(t+1)) \geq f(\mathbf{b}_i(t)) \\ \mathbf{x}_i(t+1) & \text{if} \quad f(\mathbf{x}_i(t+1)) < f(\mathbf{b}_i(t)) \end{cases} \tag{2.72}$$

In order to limit the step size with which a particle's position is adjusted, the velocities can be clamped [45]. Therefore, if a particle's velocity exceeds the specified maximum velocity, $\mathbf{V}_{max}$, the particle's velocity is set to $\mathbf{V}_{max}$. The velocity of a particle, prior to the position update, is adjusted using,

$$\mathbf{v}_{i,n}(t+1) = \begin{cases} \mathbf{v}^*_{i,n}(t+1) & \text{if} \quad \mathbf{v}^*_{i,n}(t+1) < \mathbf{V}_{max,n} \\ \mathbf{V}_{max,n} & \text{if} \quad \mathbf{v}^*_{i,n}(t+1) \geq \mathbf{V}_{max,n} \end{cases} \tag{2.73}$$

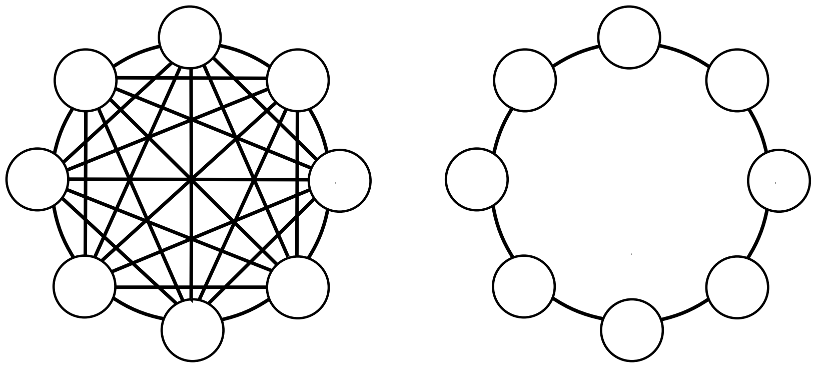


Figure 2.4 Neighbourhood Topologies in PSO

where $\mathbf{v}^*_{i,n}(t+1)$ is calculated using equation (2.70) and $\mathbf{V}_{max,n}$ is the maximum allowed velocity in dimension n, controlling the granularity of the search. The values of $\mathbf{V}_{max}$ are selected as a fraction of the domain of each dimension of the search space, using

$$\mathbf{V}_{max,n} = \delta\left(\mathbf{x}_{max,n} - \mathbf{x}_{min,n}\right) \tag{2.74}$$

where $\mathbf{x}_{max,n}$ and $\mathbf{x}_{min,n}$ are the maximum and minimum values of the n-th dimension and $\delta \in (0,1]$. Figure 2.4 illustrates the two most common neighbourhood topologies used in PSO. These are the *star* and *ring* topologies [47]. The *star* neighbourhood topology is a fully meshed network of particles where every particle is connected to every other particle in the network topology. Each particle can therefore communicate with every other particle. The *ring* topology arranges particles in a ring structure such that each particle has a number of particles to the right and left forming the particle's neighbourhood.

PSO algorithms can be applied to the problem of data clustering [174, 145]. Algorithm 2.4 lists the pseudo code for a basic PSO clustering algorithm where t_{max} is the maximum number of iterations. Each particle in the swarm represents a possible partitioning of the data set. Thus, each particle represents K number of centroids, such that $N = K \times I$ where I is the number of features. Each particle is defined as: $\mathbf{x}_i = (\mathbf{c}_{i,1}, \mathbf{c}_{i,2}, \ldots, \mathbf{c}_{i,K})$ where $\mathbf{c}_{i,k}$ is the cluster centroid of the k-th cluster, $C_{i,k}$, represented by the i-th particle. In the context of clustering, the objective function optimised by the PSO, is defined as the quantization error [174]

$$J_{PSO} = \frac{\sum_{k=1}^{K} \frac{1}{|C_{i,k}|} \sum_{\forall \mathbf{p} \in C_{i,k}} \sigma\left(\mathbf{p}, \mathbf{c}_{i,k}\right)}{K} \tag{2.75}$$

Algorithm 2.4: PSO Clustering Algorithm

```
Initialise each particle to contain $K$ randomly initialised centroids;
**for** $t$=1 to $t_{max}$ **do**
    **for** each particle $i$ **do**
        **for** each pattern $\mathbf{p}$ **do**
            Calculate the Euclidean distance $\sigma\left(\mathbf{p}, \mathbf{c}_{i,k}\right)$ (as defined in equation (2.3)) for all clusters $C_{i,k}$;
            Group pattern $\mathbf{p}$ with cluster $C_{i,k}$ such that $\sigma\left(\mathbf{p}, \mathbf{c}_{i,k}\right) = min_{k=1,\ldots,K}\left\{\sigma\left(\mathbf{p}, \mathbf{c}_{i,k}\right)\right\}$;
        **end**
        Calculate the fitness of particle $i$;
    **end**
    Determine the *best* particle position, $\mathbf{g}_i$, in the swarm using equation (2.71);
    Determine the *personal best position*, $\mathbf{b}_i$, using equation (2.72);
    Update the cluster centroids of each particle using equations (2.69) and (2.70);
**end**
```

where σ is the Euclidean distance as defined in equation (2.3) and $|C_{i,k}|$ is the number of patterns grouped with cluster $C_{i,k}$. Thus, J is the fitness function of the particles, which is measured as the quantization error ($f = J_{PSO}$).

The same PSO algorithm in [174] was applied to image segmentation in [145], but with a different fitness function. An advantage of the PSO clustering algorithm compared to K-means clustering, is that the algorithm is less sensitive to the initialisation of cluster centroids, since PSO performs a parallel search for an optimal partitioning of the data set [145, 174]. The interested reader is referred to [47] for more information on swarm intelligence algorithms.

2.7.2 Self-organising Feature Map

The Self-organising Feature Map (SOM) is a single-layered unsupervised artificial neural network algorithm which consists of a single output layer known as a *map*, $\mathbf{M}$ [118]. The structure of the map is a two-dimensional *grid* of artificial neurons with R rows and C columns, $\mathbf{M}^{R\times C}$. The map is usually a square with $R = C$ but can also be any rectangular shape with $R \neq C$.

Each pattern, $\mathbf{p}$, in a data set, P^N (where N is the number of dimensions), is associated with a single neuron in the map [96]. The number of neurons in the map is less than the number of patterns in the data set, i.e. $R \times C < |P^N|$. Each neuron in the map represents an N-dimensional

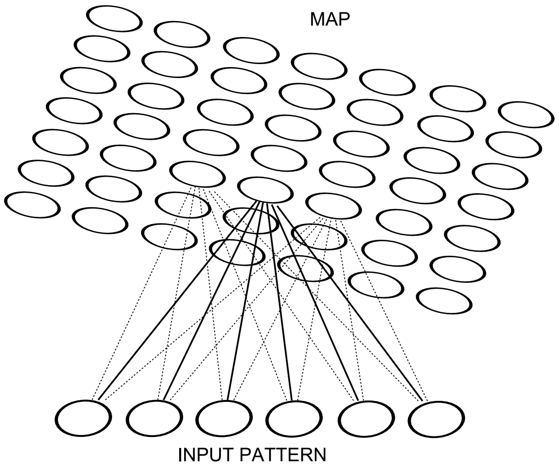


Figure 2.5 Self-organising Feature Map

weight vector, $\mathbf{w}_{rc}$, known as a *codebook* vector where r and c are the row and column indices, respectively. Figure 2.5 illustrates the association of an input pattern to the map of neurons. Training of the SOM is based on a *competitive* learning strategy where neurons compete to be the best matching neuron (BMN) to the input patterns which results in similar patterns being grouped together and represented by a single neuron [96]. Thus each codebook vector, $\mathbf{w}_{rc}$, forms the centroid of a cluster [48].

Algorithm 2.5: General SOM Algorithm

```
Initialise the learning rate $\gamma(0)$ and neighbourhood $\Lambda\left(\mathbf{w}',0\right)$;
Initialise the codebook vector of each neuron in the map, i.e. $\mathbf{w}_{rc}\ \forall r,c$;
repeat
    for each input pattern p do
        Determine the best matching neuron (BMN), $\mathbf{w}'$, using equation (2.3);
        Determine the neighbourhood, $\Lambda\left(\mathbf{w}',t\right)$, of BMN $\mathbf{w}'$;
        Using competitive learning, update the codebook vector of each neuron in
        neighbourhood $\Lambda\left(\mathbf{w}',t\right)$ by using equation (2.76);
    end
    Monotonically decrease the learning rate $\gamma(t)$;
    Reduce the neighbourhood $\Lambda\left(\mathbf{w}',t\right)$;
until some stopping criterion is satisfied;
```

Algorithm 2.5 lists general pseudo code of SOM training algorithms. There are various methods to initialise the codebook vector of each neuron. The two most common initialisation methods

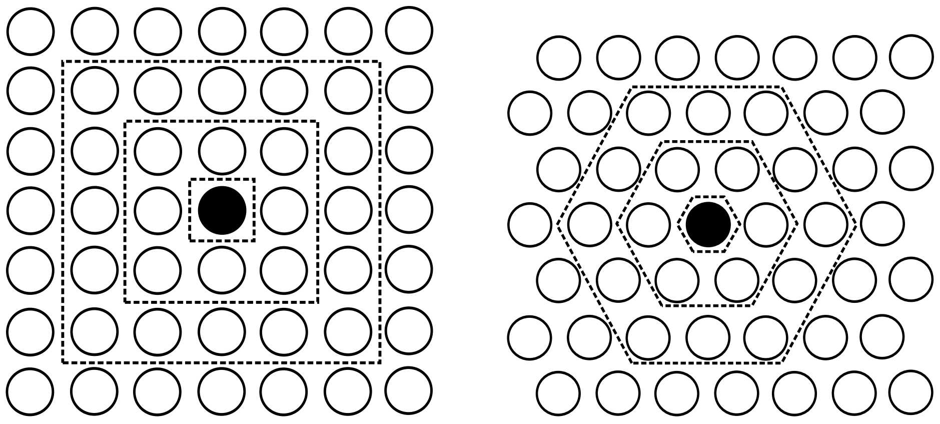

(a) Square (b) Hexagon

Figure 2.6 Neighbourhood Arrangements in SOM

are:

- randomly initialise each codebook vector, or
- randomly select input patterns as initial codebook vectors.

The codebook vector of each *neighbouring* neuron is updated after each pattern is presented to the SOM. The neuron most *similar* to an input pattern is selected as the pattern's *best matching neuron* (BMN). Similarity between an input pattern $\mathbf{p}$ and a neuron's codebook vector $\mathbf{w}_{rc}$, is measured with the Euclidean distance as defined in equation (2.3). Thus the BMN, $\mathbf{w}^{'}$, of an input pattern, $\mathbf{p}$, is the minimum Euclidean distance to the input pattern. The BMN and its neighbouring neurons are then moved closer to the input pattern by updating the codebook vectors. The neighbourhood of a BMN can be arranged as a square or hexagon lattice (as illustrated in figure 2.6) or can be determined by using a smooth Gaussian function [48]. The codebook vector, $\mathbf{w}_{rc}$, of a neighbouring neuron is updated by using

$$\mathbf{w}_{rc}(t+1) = \begin{cases} \mathbf{w}_{rc}(t) + \gamma(t)\left[\mathbf{p} - \mathbf{w}_{rc}(t)\right] & \text{if} \quad \mathbf{w}_{rc}(t) \in \Lambda\left(\mathbf{w}^{'}, t\right) \\ \mathbf{w}_{rc}(t) & \text{otherwise} \end{cases} \tag{2.76}$$

where $\gamma(t)$ is the learning rate and $\Lambda\left(\mathbf{w}^{'}, t\right)$ is the set of neighbourhood neurons of the best matching neuron, $\mathbf{w}^{'}$, for input pattern $\mathbf{p}$ at time step t. The neighbourhood radius decreases with each time step to reduce the influence of distant neighbouring neurons. The iterative learning process stops when one of the following criteria is met:

- the neighbourhood $\Lambda\left(\mathbf{w}^{'}, t\right)$ only includes the BMN $\mathbf{w}^{'}$,

- the maximum number of time steps (iterations) is exceeded,
- there are no changes in the codebook vectors, or
- the quantization error is sufficiently small. The quantization error is calculated as the sum of Euclidean distances between all input patterns and the codebook vector of BMN, defined as [48]

$$J_{SOM} = \sum_{\forall \mathbf{p} \in P} \sigma\left(\mathbf{p}, \mathbf{w}'(t)\right) \tag{2.77}$$

A SOM maps multi-dimensional data onto a two-dimensional map which is a much simpler representation of the data and is easier to interpret. Thus, a SOM maintains the topology of the data [48]. There are however a few drawbacks to SOMs [48, 96]:

- The random initialisation of codebook vectors can result in increased training times.
- Initialisation of codebook vectors to randomly selected input patterns may result in premature convergence.
- SOMs are only applicable to clustering hyper-spherical data.
- The clustering result is dependent on the number of neurons in the map.

In order to determine the clusters within the data set, the boundaries between clusters in the map need to be identified. Boundaries in the map can be determined by means of the unified distance matrix (U-matrix) technique or Ward clustering technique [3]. The former is a matrix which contains a geometrical approximation of the codebook vector distribution in the map. The latter technique is an agglomerative hierarchical clustering method which partitions the codebook vectors into a specified number of clusters. The linking of two adjacent clusters is based on the Ward distance measure which is calculated as

$$\ell_{Ward}\left(C_i, C_j\right) = \frac{|C_i| \times |C_j|}{|C_i| + |C_j|} \sigma\left(\mathbf{c}_i, \mathbf{c}_j\right) \tag{2.78}$$

where σ is the Euclidean distance measure as defined in equation (2.3). The two clusters with the smallest ℓ_{Ward} distance are merged and the centroid of the new cluster, $C_k = C_i \cup C_j$, is calculated as

$$\mathbf{c}_k = \frac{1}{|C_i| + |C_j|}\left(|C_i|\,\mathbf{c}_i + |C_j|\,\mathbf{c}_j\right) \tag{2.79}$$

2.8 Conclusion

The chapter gave a formal definition of data clustering and discussed different distance-based similarity measures which can be used to determine the similarity between two feature vectors. The chapter also discussed the most familiar clustering algorithms which are categorised into hierarchical or partitional clustering methods. This was followed by an overview of different cluster validity indices which evaluate the partitioning quality of a clustering algorithm.

Since the cluster quality of a clustering algorithm can be influenced by outliers in a data set, a brief introduction was also given on outlier mining and different techniques for outlier detection were discussed. Another influence on the cluster quality of a clustering algorithm is a changing environment. This means that the feature vectors are non-stationary. Thus, the cluster quality of a clustering algorithm needs to be measured over a period of time. Therefore, the chapter discussed existing measures to determine the performance of optimisation algorithms in a non-stationary environment. These performance measures are used to quantify and define performance measures that can be used to evaluate the cluster quality of a clustering algorithm in non-stationary environments.

The chapter ended with a discussion of the Particle Swarm Optimisation and Self-organising Feature Map algorithms and how these algorithms can be applied for clustering data. Both of these algorithms implement different neighbourhood techniques to adapt solutions to feature vectors in the data.

The proposed model in this thesis is inspired by and based on the network theory in immunology. The proposed model is a partitional clustering method which also implements a neighbourhood technique and uses the Euclidean distance as similarity measure between two feature vectors (discussed in chapter 5). The cluster validity indices of Davies-Bouldin and Ray-Turi, which are defined in equations (2.41) and (2.51) respectively, are used to evaluate the clustering quality of a clustering algorithm in this thesis. These cluster validity indices were selected since an optimal partitioning of a data set minimises all of these indices. An enhancement to the proposed algorithm in this thesis implements the sequential exception technique as discussed in section 2.6 to determine the boundaries between clusters and thereby dynamically determines the number of clusters within a data set (discussed in chapter 6). Both the original and enhanced version of the proposed clustering algorithm are also applied to clustering of non-stationary environments (discussed in chapter 7). Section 7.1 revisits the clustering performance measures discussed in this chapter and proposes a clustering performance measure for non-stationary data clustering.

Since the proposed model in this thesis is inspired by the network theory in immunology, the next chapter reviews the functional process of the natural immune system and discusses the different theories in immunology regarding the functioning and organisational behavior between lymphocytes.

Chapter 3

The Natural Immune System

The body has many defense mechanisms, which among others are the skin of the body, the membrane that covers the hollow organs and vessels, and the adaptive immune system. The adaptive immune system reacts to a specific foreign body material or pathogenic material (referred to as *antigen*). During these reactions the adaptive immune system adapts to better detect the encountered antigen and a 'memory' is built up of regular encountered antigen. The obtained memory speeds up and improves the reaction of the adaptive immune system to future exposure to the same antigen. Due to this reason defense reactions are divided into three types: non-specific defense reactions, inherited defense reactions and specific defense reactions [127]. The adaptive immune system forms part of the specific defense reactions.

Different theories exist in the study of immunology regarding the functioning and organisational behavior between lymphocytes in response to encountered antigen. These theories include the classical view, clonal selection theory, network theory, and danger theory. Since the clonal selection, danger theory and network theory are based on concepts and elements within the classical view (as discussed in section 3.1), the classical view will first be discussed in detail to form a bases onto which the other three theories will be explained in sections 3.5, 3.6 and 3.7 respectively.

3.1 Classical View

The classical view of the immune system is that the immune system distinguishes between what is normal (*self*) and foreign (*non-self* or antigen) in the body. The recognition of antigens leads to the creation of specialised activated cells, which inactivate or destroy these antigens. The natural immune system mostly consists of lymphocytes and lymphoid organs. These organs are the tonsils and adenoids, thymus, lymph nodes, spleen, Peyer's patches, appendix, lymphatic vessels, and bone marrow. Lymphoid organs are responsible for the growth, development and deployment of the lymphocytes in the immune system. The lymphocytes are used to detect any antigens in the body. The immune system works on the principle of a pattern recognition system, recognising *non-self* patterns from the *self* patterns [149].

The initial classical view was defined by Burnet [22] as B-Cells and Killer-T-Cells with antigen-specific receptors. Antigens triggered an immune response by interacting with these receptors. This interaction is known as stimulation (or signal 1). It was Bretscher and Cohn [20] who enhanced the initial classical view by introducing the concept of a helper T-Cell (see section 3.3.3). This is known as the *help* signal (or signal 2). In later years, Lafferty and Cunningham added a co-stimulatory signal to the helper T-Cell model of Bretscher and Cohn. Lafferty and Cunningham [120] proposed that the helper T-Cell is co-stimulated with a signal from an antigen-presenting cell (APC). The motivation for the co-stimulated model was that T-Cells in a body had a stronger response to cells from the same species in comparison to cells from different species. Thus, the APC is species specific. Burnet also introduced the theory of *clonal selection* [22].

The rest of this chapter explains the development of the different cell types in the immune system, antigens and antibodies, immune reactions and immunity types and the detection process of foreign body material as defined by the different theories.

3.2 Antibodies and Antigens

Within the natural immune system, antigens are material that can trigger immune response. An immune response is the body's reaction to antigens so that the antigens are eliminated to prevent damage to the body. Antigens can be either bacteria, fungi, parasites and/or viruses [157]. An antigen must be recognised as foreign (*non-self*). Every cell has a huge variety of antigens in its surface membrane. The foreign antigen is mostly present in the cell of micro-organisms and in the cell membrane of 'donor cells'. Donor cells are transplanted blood cells obtained through transplanted organs or blood. The small segments on the surface of an antigen are called *epitopes* and the small segments on antibodies are called *paratopes* (as shown in figure 3.1). Epitopes trigger a specific immune response and antibodies' paratopes bind to these epitopes with a certain binding strength, measured as affinity [127]. Note that the binding between an epitope and paratope has a complementary match in shape.

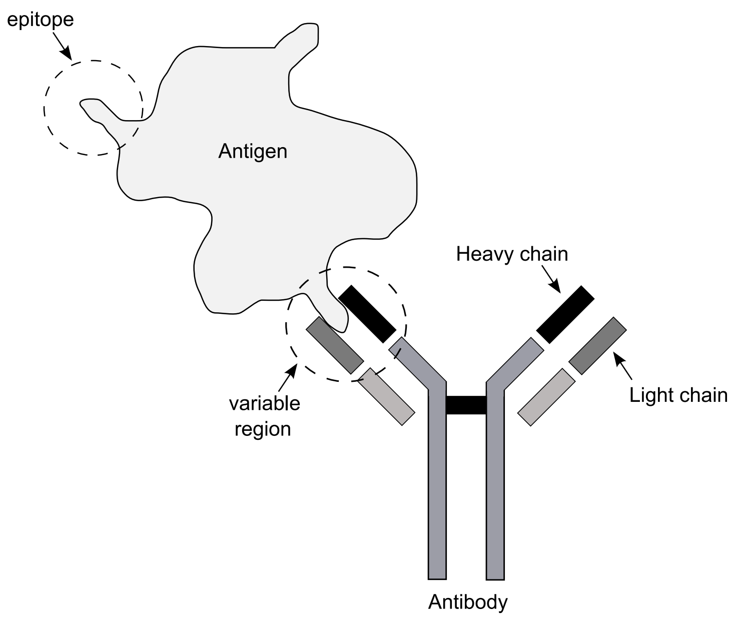


Figure 3.1 Antigen-Antibody-Complex

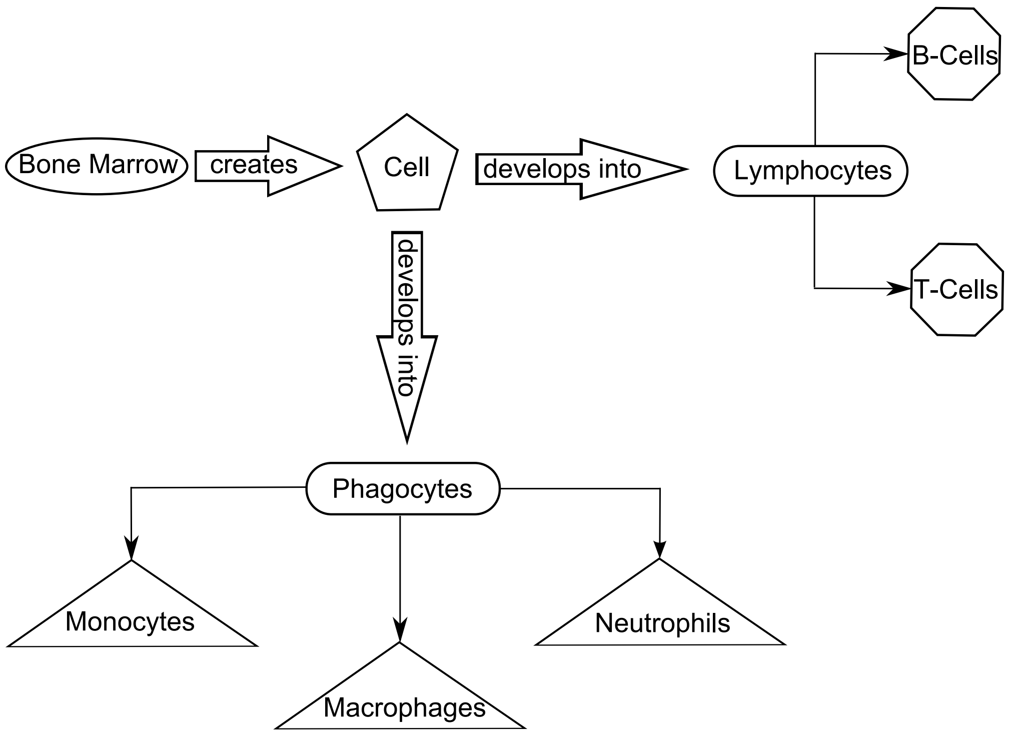


Figure 3.2 White Cell Types

Antibodies are chemical proteins. In contradiction to antigens, antibodies form part of *self* and are produced when lymphocytes encounter antigen (*non-self*). An antibody has a Y-shape (as shown in figure 3.1). Both arms of the Y consist of two identical heavy and two identical light chains. The chains are differentiated as *heavy* and *light* since the heavy chain contains double the number of amino-acids than the light chain. The tips of the arms are called the variable regions and vary from one antibody to another [157]. The variable regions (paratopes) enable the antibody to match antigen and bind to the epitopes of an antigen. After a binding between an antibody and an antigen's epitope, an antigen-antibody-complex is formed, which results into the de-activation of the antigen [127]. There are five classes of antibodies: IgM, IgG, IgA, IgE, IgD [127].

3.3 The White Cells

All cells in the body are created in the bone marrow (as illustrated in figure 3.2). Some of these cells develop into large cell- and particle-devouring white cells known as phagocytes [157]. Phagocytes include monocytes, macrophages and neutrophils. Macrophages are versatile cells that secrete powerful chemicals and play an important role in T-Cell activation. Other cells develop into small white cells known as lymphocytes.

3.3.1 The Lymphocytes

There are two types of lymphocytes: the T-Cell and B-Cell, both created in the bone marrow. On the surface of the T-Cells and B-Cells are receptor molecules that bind to other cells. The T-Cell binds only with molecules that are on the surface of other cells. The T-Cell first becomes mature in the thymus, whereas the B-Cell is already mature after creation in the bone marrow. A T-Cell becomes mature if and only if it does not have receptors that bind with molecules that represent *self* cells. It is therefore very important that the T-Cell can differentiate between *self* and *non-self* cells.

Thus lymphocytes have different states: immature, mature, memory and annihilated (figure 3.3 illustrates the life cycle of lymphocytes). These states are discussed in the subsections to follow below. Both T-Cells and B-Cells secrete lymphokines and macrophages secrete monokines. Monokines and lymphokines are known as cytokines and their function is to encourage cell growth, promote cell activation or destroy target cells [157]. These molecules on the surface of a cell are named the major histocompatibility complex molecules (MHC-molecules). Their main function is to bring to light the internal structure of a cell. MHC-molecules are grouped into two classes: Type I and Type II. MHC-molecules of Type I is on the surface of any cell and MHC-molecules of Type II mainly on the surface of B-Cells [149]. There are two types of T-Cells: The Helper-T-Cell and Natural-Killer-T-Cell. Each of these types of lymphocytes is described in detail below.

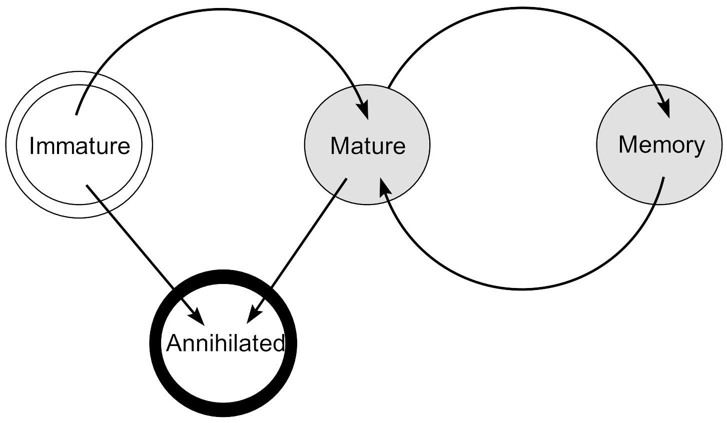


Figure 3.3 Life Cycle of a Lymphocyte

3.3.2 The B-Cell Lymphocyte

B-Cells are created in the bone marrow with monomeric IgM-receptors on their surfaces. A monomeric receptor is a chemical compound that can undergo a chemical reaction with other molecules to form larger molecules. In contrast to T-Cells, B-Cells leave the bone marrow as mature lymphocytes. B-Cells mostly exist in the spleen and tonsils. It is in the spleen and tonsils that the B-Cells develop into plasma cells after the B-Cells are exposed to antigens. After developing into plasma cells, the plasma cells produce antibodies that are effective against antigens [127].

The B-Cell has antigen-specific receptors and recognises in its natural state the antigens. When contact is made between a B-Cell and antigen, clonal proliferation on the B-Cell takes place and is strengthened by Helper-T-Cells (as explained in the next subsection). During clonal proliferation two types of cells are formed: plasma cells and memory cells. The function of memory cells is to proliferate to plasma cells for a faster reaction to frequently encountered antigens and produce antibodies for the antigens. A plasma cell is a B-Cell that produces antibodies.

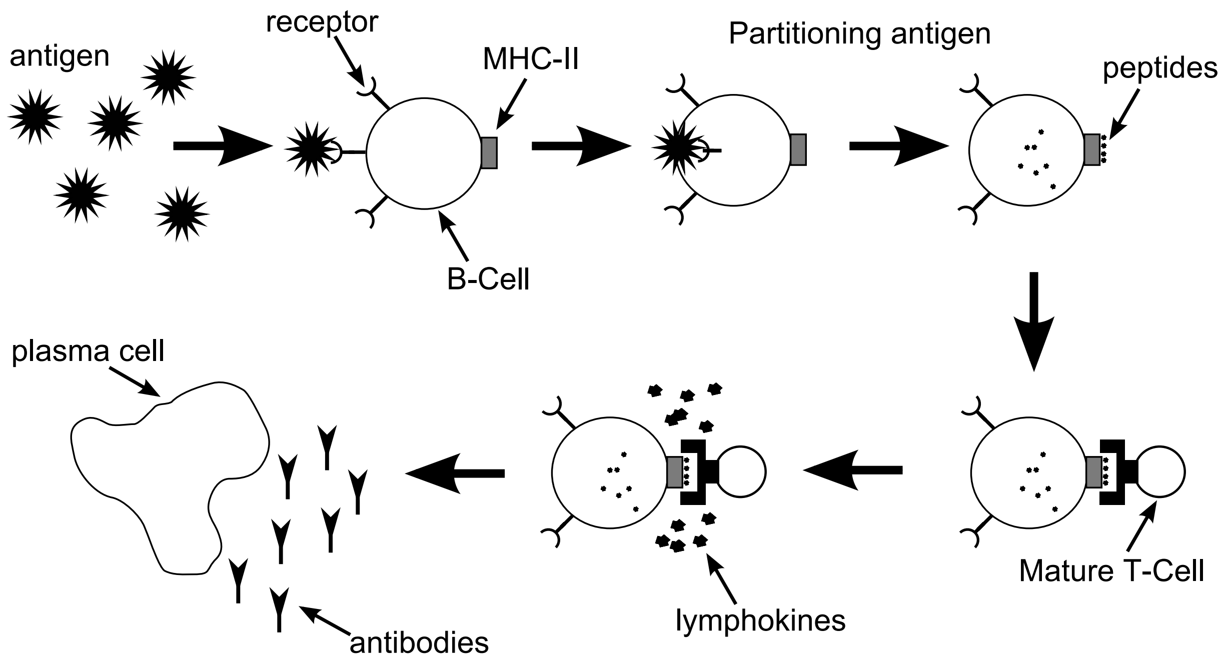


Figure 3.4 B-Cell Develops into Plasma Cell, Producing Antibodies

3.3.3 The Helper-T-Cell (HTC)

When a B-Cell's receptor matches an antigen, the antigen is partitioned into peptides (as shown in figure 3.4). The peptides are then brought to the surface of the B-Cell by an MHC-molecule of Type II. Macrophages also break down antigen and the broken down antigen is brought to the surface of the macrophage by an MHC-molecule of Type II. The HTC binds to the MHC-molecule on the surface of the B-Cell or macrophage and proliferates or suppresses the B-Cell response to the partitioned cell, by secreting lymphokines. This response is known as the primary response. When the HTC bounds to the MHC with a high affinity, the B-Cell is proliferated. The B-Cell then produces antibodies with the same structure or pattern as represented by the peptides. The production of antibodies is done after a *cloning process* of the B-Cell.

When the HTC does not bind with a high affinity, the B-Cell response is suppressed. Affinity is a force that causes the HTC to elect an MHC on the surface of the B-Cell with which the HTC has a stronger binding to unite, rather than with another MHC with a weaker binding. A higher affinity implies a stronger binding between the HTC and MHC. The antibodies then bind to the antigens' epitopes that have the same complementary structure or pattern. Epitopes are the portions on an antigen that are recognised by antibodies. When a B-Cell is proliferated enough, i.e. the B-Cell frequently detects antigens, it goes into a memory status, and when suppressed frequently, it becomes annihilated and replaced by a newly created B-Cell. The immune system uses the B-Cells with memory status in a secondary response to frequently seen antigens of the same structure. The secondary response is much faster than the primary response, since no HTC signal or binding to the memory B-Cell is necessary for producing antibodies [149].

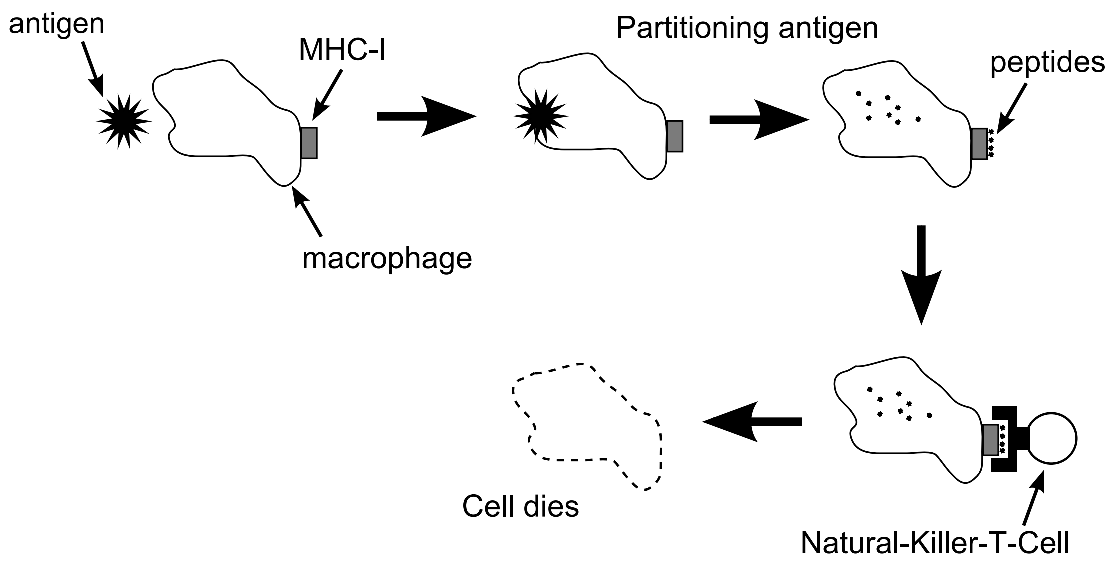


Figure 3.5 Macrophage and NKTC

3.3.4 The Natural-Killer-T-Cell (NKTC)

The NKTC binds to MHC-molecules of Type I (as illustrated in figure 3.5). These MHC-molecules are found on all cells. Their function is to bring to light any viral proteins from a virally infected cell. The NKTC then binds to the MHC-molecule of Type I and destroys not only the virally infected cell but also the NKTC itself [149].

3.4 Immunity Types

Immunity can be obtained either naturally or artificially (as illustrated in figure 3.6). In both cases immunity can be active or passive. Antigens are only encountered in active immunity and activate the immune system. The activated immune system reacts to these antigens by producing memory cells. This implies that memory cells are only produced in active immunity. Although the production of these memory cells in active immunity is much more time consuming compared to passive immunity, active immunity is permanent and passive immunity only temporary.

Passive immunity does however have an immediate reaction to encountered antigen. This section discusses the different types of immunity.

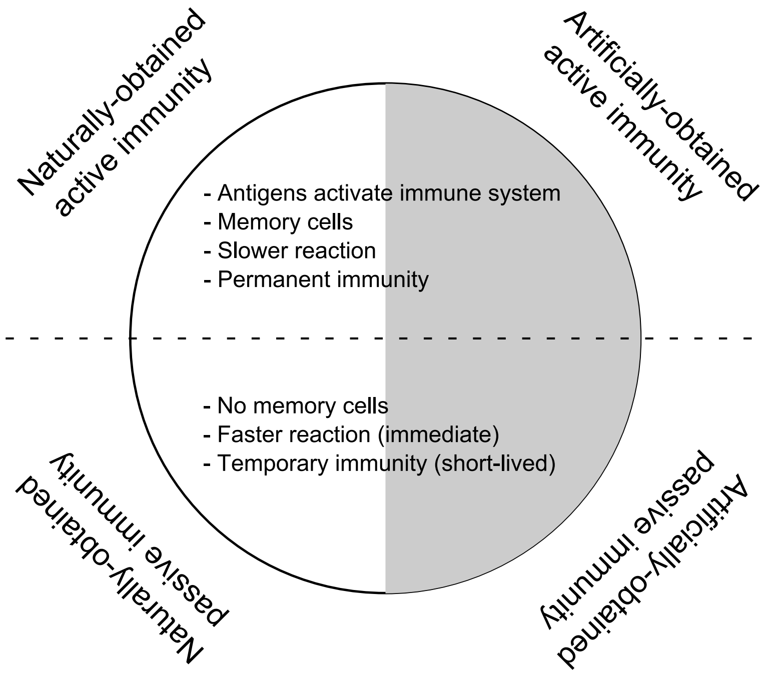

Naturally-obtained active immunity
Artificially-obtained active immunity
- Antigens activate immune system
- Memory cells
- Slower reaction
- Permanent immunity
- No memory cells
- Faster reaction (immediate)
- Temporary immunity (short-lived)
Naturally-obtained passive immunity
Artificially-obtained passive immunity

Figure 3.6 Immunity Types

Naturally-obtained active immunity: The immune system of an antigen-infected body reacts to the antigen by producing antibodies. The production of memory cells is an end-result of frequently encountered antigen. Due to memory cells, active naturally-obtained immunity is more or less permanent. This type of immunity can also develop when the body receives foreign red blood cells and actively produces antibodies to deactivate the antigen [127].

Naturally-obtained passive immunity: Naturally-obtained passive immunity is short-lived since antibodies are continuously broken down without creation of new antibodies. New antibodies are not created because the antigens did not activate the *self* immune system. The immunity type develops from IgG-antibodies that are transplanted from the mother to the baby. The secreted IgA-antibodies in mothers-milk are another example of this immunity type and protect the baby from any antigens with which the mother came into contact [127].

Artificially-obtained active immunity: Artificially-obtained active immunity develops when dead organisms or weakened organisms are therapeutically applied. The concept is that special treated organisms keep their antigens without provoking illness-reactions [127].

Artificially-obtained passive immunity: Artificially-obtained passive immunity is obtained when a specific antibody that was produced by another human or animal, is injected into the body for an emergency treatment. Since the immune system was not activated to generate antibodies and produce memory cells, immunity is short-lived and temporary [127].

3.5 The Process of Affinity Maturation

Learning in the immune system is based on increasing the population size of those lymphocytes that frequently recognise antigens. Learning by the immune system is done by a process known as affinity maturation. As illustrated in figure 3.7, affinity maturation can be broken down into two smaller processes, namely a cloning process (left side of figure 3.7) and a somatic hyper mutation process (right side of figure 3.7). The cloning process is more generally known as *clonal selection*, which is the proliferation of the lymphocytes that recognise the antigens.

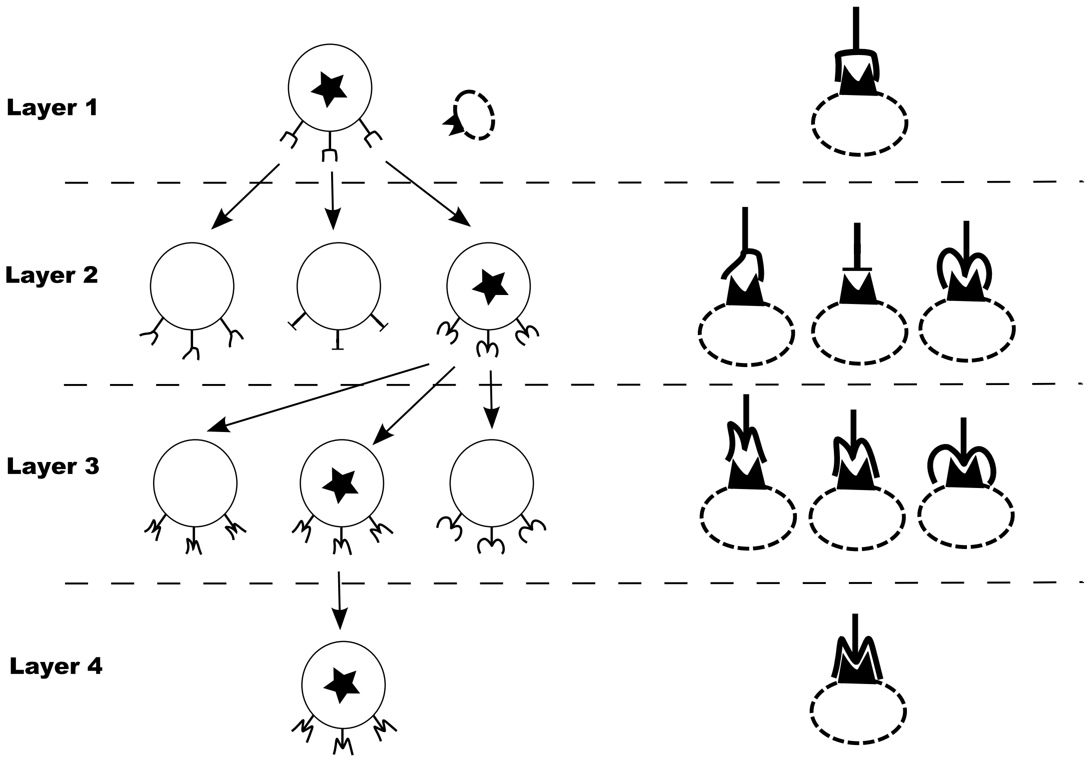


Figure 3.7 Affinity Maturation of Lymphocytes

The interaction of the lymphocyte with an antigen leads to an activation of the lymphocyte where upon the cell is proliferated and grown into a clone (activated lymphocytes are indicated with a star in figure 3.7). When an antigen stimulates a lymphocyte, the lymphocyte not only secretes antibodies to bind to the antigen but also generates mutated clones of itself in an attempt to have a higher binding affinity with the detected antigen (layer 1 to layer 2, layer 2 to layer 3, etc. in figure 3.7). The latter process is known as somatic hyper mutation. Somatic hyper mutation only occurs in the germinal centers of the different lymphoid organs, which therefore are spatially organised into different zones [123]. Thus, through repetitive exposure to the antigen, the immune system learns and adapts to the shape of the frequently encountered antigen (as illustrated in the right side of figure 3.7) and moves from a random receptor creation (layer 1 in figure 3.7) to a repertoire that represents the antigens more precisely (layer 4 in figure 3.7). Lymphocytes in a clone produce antibodies if it is a B-Cell and secrete growth factors (lymphokines) in the case of an HTC.

Since antigens determine or select the lymphocytes that need to be cloned, the process is called *clonal selection* [127]. The fittest clones are those which produce antibodies that bind to antigen best (with highest affinity). Since the total number of lymphocytes in the immune system is regulated, the increase in size of some clones decreases the size of other clones. This leads to the immune system forgetting previously learned antigens. When a familiar antigen is detected, the immune system responds with larger cloning sizes. This response is referred to as the secondary immune response [149]. Learning is also based on decreasing the population size of those lymphocytes that seldom or never detect any antigens. These lymphocytes are removed from the immune system. For the affinity maturation process to be successful, the receptor molecule repository needs to be as complete and diverse as possible to recognise any foreign shape [149].

3.6 The Network Theory

The network theory was first introduced by Jerne [97, 98] and further developed and formulated by Perelson [148]. The variable region of an antibody can be antigenic and invoke an immune response. Thus, the variable region of an antibody, responsible for binding to an antigen, has an antigenic profile. This antigenic profile is known as the *idiotype* of the antibody. The idiotype of an antibody can invoke an immune response for the creation of anti-idiotypic antibodies by a stimulated B-Cell [98]. As illustrated in figure 3.8, the idiotopic profile of an antibody consists of multiple sites in the variable region of an antibody. These sites are known as *idiotopes*.

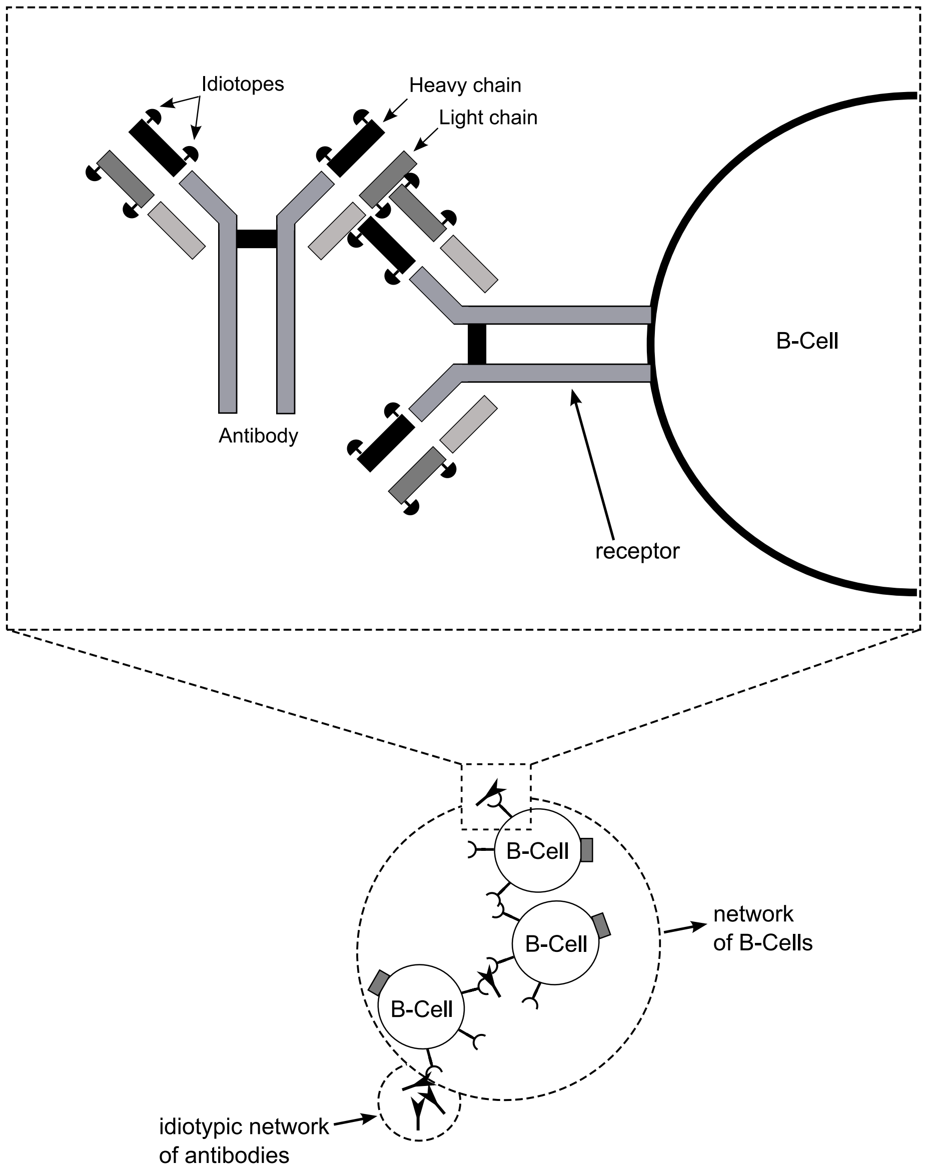


Figure 3.8 Idiotypic Network of Antibodies and B-Cells

In summary, the network theory of antibodies and B-Cells states that B-Cells are interconnected to form an idiotypic network of cells. When a B-Cell in the network responds to a foreign cell, the activated B-Cell stimulates all the other B-Cells to which it is connected in the network. Thus, a lymphocyte is not only stimulated by an antigen, but can also be stimulated or suppressed by neighbouring lymphocytes. That is, when a lymphocyte reacts to the stimulation of an antigen, the secretion of antibodies and generation of mutated clones (as discussed in section 3.5) stimulate the lymphocyte's immediate neighbours, if the neighbouring B-Cells bind to the idiotopes of the produced antibodies or receptor of the stimulated B-Cell. This implies that a neighbour lymphocyte can then in turn also react to the stimulation of the antigen-stimulated lymphocyte by generating mutated clones, stimulating the immediate group of neighbours [149] (as illustrated in figure 3.8). Therefore, lymphocytes signal (or communicate) each other across spatial distances by means of anti-idiotypic networks.

The interactions or connections between the cells in an idiotypic network determine the resultant architecture or topology of the network. The possible interactions in an idiotypic network can be presented by different network topologies which include but are not limited to the linear topology, the simple cyclic topology, the affinity matrix topology and the Caylee tree topology. These network topologies are discussed in more detail with illustrations in section 4.7.

3.7 The Danger Theory

The danger theory was introduced by Matzinger [124, 125] and is based on the co-stimulated model of Lafferty and Cunningham [120]. The main idea of the *danger theory* is that the immune system distinguishes between what is dangerous and non-dangerous in the body. The *danger theory* differs from the classical view in that the immune system does not respond to all foreign cells, but only to those foreign cells that are harmful or dangerous to the body. A foreign cell is seen to be dangerous to the body if it causes body cells to stress or die. Matzinger gives two motivational reasons for defining the new theory, which is that the immune system needs to adapt to a changing *self* and that the immune system does not always react on *foreign* or *non-self*.

Although cell death is common within the body, the immune system only reacts to those cell deaths that are not normal programmed cell death (apoptosis), i.e. non-apoptotic or necrotic deaths. When a cell is infected by a virus, the cell itself will send out a stress signal (known as signal 0) of necrotic death to activate the antigen presenting cells (APCs) (as illustrated in figure 3.9). Thus, co-stimulation of an APC to a helper T-Cell is only possible if the APC was activated with a *danger* or stress signal. Therefore, the neighbouring cells of an APC determine the APC's state. Hereon the immune reaction process is as discussed within the classical view (see section 3.1), where mature helper T-Cells are now presented with a peptide representation of the antigen and co-stimulated by an activated APC.

The different types of signals from a dying or stressed cell are unknown. According to Matzinger

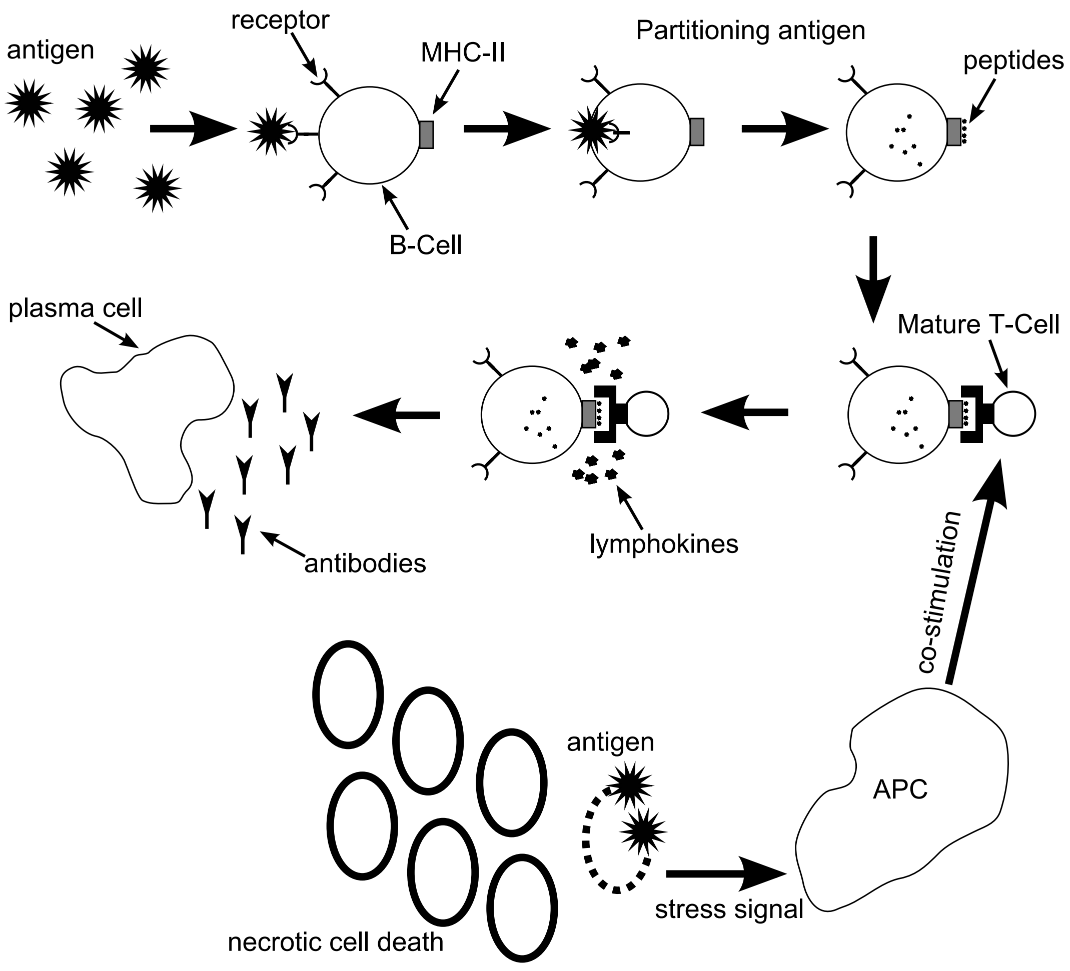


Figure 3.9 Co-Stimulation of T-Cell by an APC

these signals could either be defined as the sensing of a certain protein within a cell that leaked after the cell's death or an unexpected connection lost between connected cells after one of the cells died. Thus, if none of the above signals are fired by a cell, no immune response will be triggered by an antigen to activate the antigen presenting cells (APCs).

Thus, from a *danger* immune system perspective, a T-Cell only needs to be able to differentiate APCs from any other cells. If an APC activated a T-Cell through co-stimulation, then only will the immune system respond with a clonal proliferation of the B-Cell (as discussed in section 3.3.3). The B-Cell will then secrete antibodies to bind with the *dangerous* antigen instead of binding to all foreign harmless antigen.

3.8 The Dendritic Cell System

Dendritic cells were first identified by Banchereau and Steinman [12]. An immature dendritic cell (DC) can be generated when the appropriate cytokines are applied to blood monocytes, but is also developed in the bone marrow. A mature DC is an antigen presenting cell (APC) which initiates an immune response. A difference between immature and mature DCs is that immature DCs have a lower probability to initiate an immune response, and is more specialised in processing encountered antigens, i.e. immature DCs cannot activate T-Cells [12].

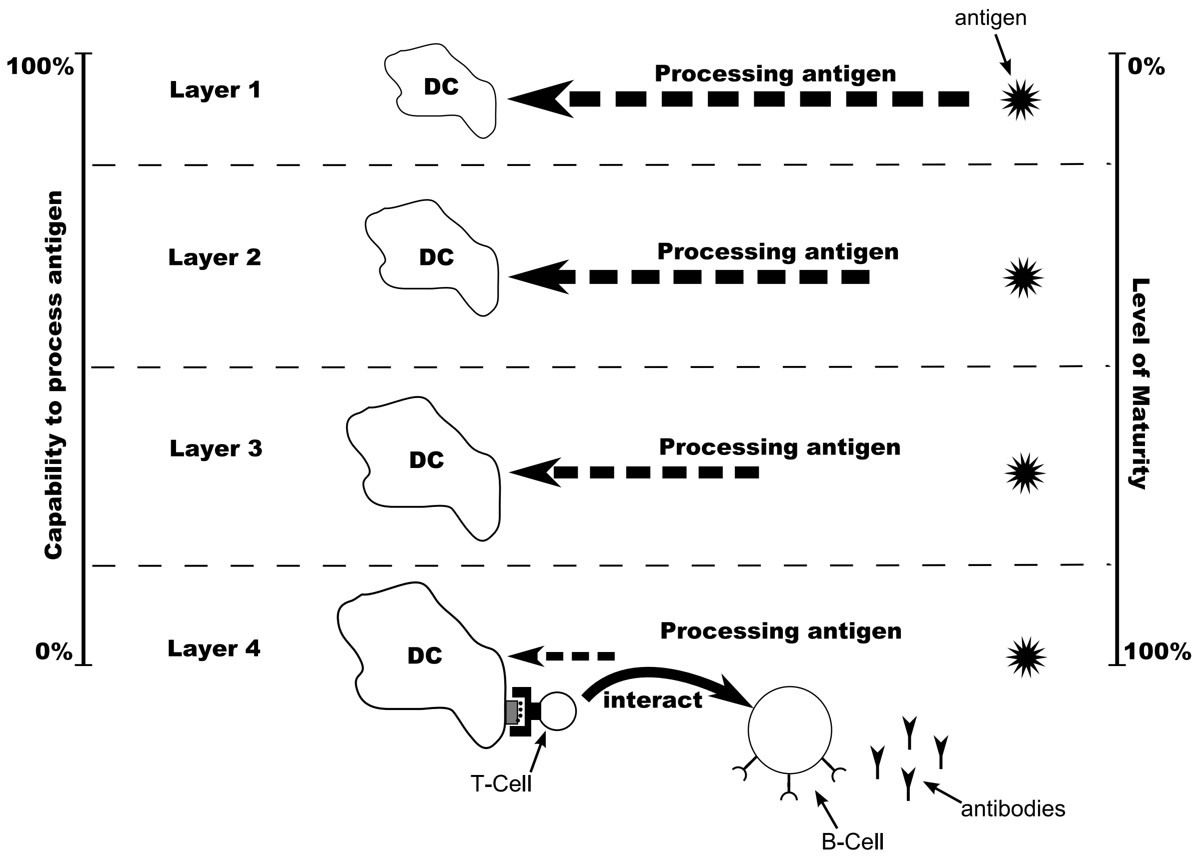


Figure 3.10 Maturation of Dendritic Cells

As illustrated in figure 3.10, initially (layer 1 in figure 3.10), immature DCs process antigen and in time become less capable in processing antigen (as indicated by a shorter dashed arrow in figure 3.10), becoming APCs which stimulate and activate T-Cells. Thus, an immature DC becomes mature by processing encountered antigen (as indicated by an enlarged DC in figure 3.10 at each layer of processing an antigen), which results in the formation of MHC-peptide complexes on the dendritic cell's surface (similar to the antigen presenting B-Cell, as discussed in section 3.3.3).

The MHC-peptide complexes are then presented to an antigen-specific T-Cell (mature T-Cell), co-stimulating and activating the T-Cell and thus initiating an immune response (similar to the APCs as discussed in section 3.7) [163]. The activated T-Cell interacts with B-Cells, which in turn produce antibodies (as discussed in section 3.3.3).

Both the B-Cell and DC are APCs, which are either directly or indirectly responsible for the secretion of antibodies [12]. DCs are responsible for Helper-T-Cell activation, which in turn promotes the proliferation of B-Cells to produce antibodies (as discussed in section 3.5). Thus, DCs *carry* antigenic *information* to lymph nodes where T-Cells reside which can react to the antigen [159].

When an MHC-peptide complex on the surface of a virally infected cell is presented to a Natural Killer T-Cell (as discussed in section 3.3.4), the Natural Killer T-Cell (NKTC) first needs to be activated by a DC before the NKTC can kill the virally infected cell. This is known as *cross-presentation* and enables the DC to initiate an immune response without getting infected by the pathogen [159, 163].

The immune system consists of many different types of dendritic cells (DC) with specialised roles. Follicular DCs (FDCs) directly maintain the growth of stimulated B-Cells [12]. FDCs are found in the lymph nodes. FDCs do not process antigen but capture antigen-antibody complexes which reside on the FDC's surface. FDCs are present in areas of antigen stimulated B-Cells. A stimulated B-Cell proliferates (the process of affinity maturation as explained in section 3.5), and when the B-Cell matches an antibody-antigen complex on the surface of an FDC with a high affinity, processes the antigen and presents the MHC-peptide complex to the T-Cell (as discussed in section 3.3.3). This ensures the survival of stimulated B-Cells with high affinities, while less stimulated B-Cells with lower affinities apoptose (normal programmed cell death) [12].

Since DCs are of crucial importance in the initiation of an immune response, research has been done on vaccines for *tuberculosis* that targets the expansion of DC populations [126]. The increased population of DCs resulted in better T-Cell activation, i.e. amplifying the level of immune activation that led to stable memory formations.

3.9 Conclusion

This chapter introduced the different theories of immunology. These are the classical view, the process of affinity maturation, the network theory, and the danger theory. With reference to the classical view, the co-operation between T-Cell and B-Cell lymphocytes to react to an encountered antigen was discussed. The binding between an antigen and antibody was also briefly discussed. The chapter also gave an overview of the different immunity types. These immunity types can either activate or assist the immune system to react to an encountered antigen. The different immunity types are: naturally-obtained active immunity, naturally-obtained passive immunity, artificially-obtained active immunity and artificially-obtained passive immunity. This chapter was concluded by a brief introduction to the dendritic cell system, with reference to immunology.

The proposed model in this thesis was mainly inspired by the network theory of immunology. The network of lymphocytes learns the structure of an antigen through the process of affinity maturation, where the activated lymphocyte proliferates by generating mutated clones which co-stimulate the immediate neighbours of the activated lymphocyte. The neighbouring lymphocytes in turn could also react and proliferate by generating mutated clones, stimulating immediate neighbours. The network topology of co-stimulated lymphocytes to adapt to the antigen structure inspired the development of the proposed model in this thesis with application to data clustering problems in stationary and non-stationary environments. The proposed model adapts a population of artificial lymphocytes through cloning and mutation operations. Furthermore, co-stimulation between neighbouring artificial lymphocytes is simulated with a pre-defined network topology. Chapter 5 introduces and discusses the proposed model in more detail.

The next chapter discusses some of the most familiar artificial immune system (AIS) models which are inspired by the different theories in the science of immunology.

Chapter 4

Artificial Immune Systems

The different theories in the science of immunology inspired the development (design) of immune inspired algorithms, collectively known as artificial immune systems (AIS), which are either based on or inspired by a specific theory on immunology or a combination of the different theories. The application areas of AIS cover a broad spectrum. AISs have been applied to different problem domains which among others include classification [30, 53, 63, 93, 179], anomaly and fraud detection [53, 60, 89, 90, 111, 121], optimisation [27, 35, 37, 55, 129], scheduling [57, 79, 130], data analysis and clustering [24, 32, 36, 133, 139, 165, 169, 187], and robotics [23, 104, 177]. This chapter discusses some of the most familiar AIS models and their applications. Since the proposed AIS model in this thesis is inspired by and mostly based on the network theory, a more detailed overview is given on existing network based AIS models within the context of data clustering.
The rest of the chapter is organised as follows:

- Section 4.1 defines a general AIS framework to highlight the basic components of an AIS model.
- Section 4.2 gives an introduction to the *shape space* model and how an artificial lymphocyte (ALC) and antigen pattern are presented in a shape space.
- Section 4.3 discusses the different measures of affinity between an ALC and antigen pattern within a specific shape space. The section also gives an overview of the different matching rules to determine whether an ALC binds to an antigen pattern.
- Section 4.4 gives an overview of AIS models which are inspired by the self-tolerant T-Cells in the natural immune system (classical view).

- Section 4.5 discusses some of the AIS models which are inspired by the clonal selection theory.
- Section 4.6 gives an introduction to some of the theoretical network based models and a detailed discussion on different network theory inspired AIS models.
- Section 4.7 discusses the different theoretical approaches to determine the possible interactions in an idiotypic network.
- Section 4.8 briefly highlights the difference between danger theory inspired AIS models and those AIS models which are inspired by the classical view of the natural immune system. The section briefly discusses some of the applications of the danger AIS models.
- Section 4.9 concludes the chapter by giving an overall summary of the chapter and comparing existing network based AIS models to the network AIS proposed in this thesis.

4.1 A Basic AIS Framework

This section defines a general AIS framework which is based upon the functional and organisational behaviour of the natural immune system (NIS) as discussed in chapter 3. In this section, the terms "cell" and "molecule" are used interchangeably. The capabilities of the NIS within each theory are summarised below:

- In some cases the NIS knows the structure of *self*/normal cells and *non-self*/*foreign* cells. In other cases the NIS only knows the structure of *self*/normal cells.
- In cases where the NIS knows the structure of *self* and *non-self* cells, the NIS is capable of recognising *non-self* associated cell structures (innate immune system).
- In cases where the NIS only knows the structure of *self* cells, the NIS needs to learn the structure of the *non-self* cells (adaptive immune system).
- A *foreign* cell is capable of causing *damage*.
- Lymphocytes are cloned and mutated to learn and adapt to the structure of the encountered *foreign* cells.
- The build-up of a *memory* on the learned structures of the *foreign* cells.

- A faster secondary response to frequently encountered *foreign* cells, due to the built-up *memory*.
- The co-operation and co-stimulation among lymphocytes to learn and react to encountered *foreign* cells can result into the formation of lymphocyte networks.
- The NIS consists of layers of defense against *foreign* cells.
- The entities within the different layers communicate in response to encountered *foreign* cells by means of signaling.

The above capabilities imply that it is the co-operation between different lymphocytes in different layers of the natural immune system which results in an active immune response to detected pathogenic material. To recapitulate the discussion on the different theories and layers of the NIS in chapter 3: Foreign cells are detected by macrophages in the first layer of defense, known as the innate immune system. If a foreign cell is not detected in the innate immune layer, mature T-Cells and B-Cells react to the encountered foreign cell in the adaptive immune system, i.e. second layer of defense. The response within the adaptive layer can be either primary or secondary.

In the primary response, B-Cells and T-Cells co-operate and co-stimulate each other in an attempt for the B-Cell to secrete antibodies with a higher affinity to the detected foreign cell. If foreign cells with a similar structure are frequently detected in the primary response, a memory of the structure is built-up in the NIS by the B-Cells that proliferate. In the secondary response, these memory cells can then have a faster reaction to the foreign cell with a similar structure, thus there is no need for a primary response to adapt to the structure of the foreign cell.

Within the primary response, B-Cells adapt to the structure of the foreign cell through the process of affinity maturation. This can result in B-Cells detecting each others structure to form an idiotypic network, co-stimulating or suppressing each other in response to an encountered foreign cell. Before a B-Cell can proliferate or undergo the affinity maturation process, the helper T-Cell needs to secrete lymphokines which either promote or suppress B-Cell growth. When a helper T-Cell receives a danger signal from the innate immune layer, indicating necrotic cell death, the T-Cell secretes lymphokines which promote B-Cell growth. This in turn proliferate the B-Cell. If a T-Cell is presented with a peptide pattern by a B-Cell without receiving a danger signal from the innate immune system, it implies that although the detected cell is foreign, it is harmless to the body and the T-Cell can secrete lymphokines to suppress the proliferation of the B-Cell.

The interaction of T-Cells, B-Cells and signaling of danger, occurs within the lymph nodes which are connected with lymph vessels. T-Cells and B-Cells *meet* each other in the lymph nodes to *exchange* antigenic information. Thus, to model an AIS, there are a few basic concepts that must be considered:

- There are trained detectors (artificial lymphocytes) that detect non-self patterns with a certain affinity.
- The artificial immune system may need a good repository of self patterns or self and non-self patterns to train the artificial lymphocytes (ALCs) to be self-tolerant.
- The affinity between an ALC and a pattern needs to be measured. The measured affinity indicates to what degree an ALC detects a pattern.
- To be able to measure affinity, the representation of the patterns and the ALCs need to have the same structure.
- The affinity between two ALCs needs to be measured. The measured affinity indicates to what degree an ALC links with another ALC to form a network.
- The artificial immune system has memory that is built-up by the artificial lymphocytes that frequently detect non-self patterns.
- When an ALC detects non-self patterns, it can be *cloned* and the clones can be mutated to have more diversity in the search space.

Using the above concepts as a guideline, the pseudo code in algorithm 4.1 is a template for the AIS algorithms considered in this thesis. Each of the algorithm's parts is briefly explained next.

1. **Initialising $\mathcal{B}$ and determining $\mathcal{A}$**: The population $\mathcal{B}$ can be populated either with randomly generated ALCs or with ALCs that are initialised with a cross section of the data set to be learned. If a cross section of the data set is used to initialise the ALCs, the complement of the data set will determine the training set $\mathcal{A}$. These and other initialisation methods are discussed for each of the AIS models in the sections to follow.
2. **Stopping condition for the *while*-loop**: In most of the discussed AIS models, the stopping condition is based on convergence of the ALC population or a preset number of iterations.

Algorithm 4.1: AIS Algorithm Template

```
Initialise a set of ALCs as population $\mathcal{B}$;
Determine the antigen patterns as training set $\mathcal{A}$;
while some stopping condition(s) not true do
    for each antigen pattern $\mathbf{a}_j \in \mathcal{A}$ do
        Select a subset of ALCs for exposure to $\mathbf{a}_j$, as population $\mathcal{S} \subseteq \mathcal{B}$;
        for each ALC, $\mathbf{b}_i \in \mathcal{S}$ do
            Calculate the antigen affinity between $\mathbf{a}_j$ and $\mathbf{b}_i$;
        end
        Select a subset of ALCs with the highest calculated antigen affinity as population
        $\mathcal{H} \subseteq \mathcal{S}$;
        Adapt the ALCs in $\mathcal{H}$ with some selection method, based on the calculated
        antigen affinity and/or the network affinity among ALCs in $\mathcal{H}$;
        Update the stimulation level of each ALC in $\mathcal{H}$;
    end
    Adapt the ALCs in $\mathcal{H}$ with a network selection method, based on the calculated
    network affinity among ALCs in $\mathcal{H}$ (optional);
end
```

3. **Selecting a subset, $\mathcal{S}$, of ALCs**: The selected subset $\mathcal{S}$ can be the entire set $\mathcal{B}$ or a number of randomly selected ALCs from $\mathcal{B}$. Selection of $\mathcal{S}$ can also be based on the stimulation level (as discussed below).

4. **Calculating the *antigen affinity***: The antigen affinity is the measurement of similarity or dissimilarity between an ALC and an antigen pattern. The most commonly used measures of affinity in existing AIS models are the Euclidean distance, r-contiguous matching rule, hamming distance and cosine similarity.

5. **Selecting a subset, $\mathcal{H}$, of ALCs**: In some of the AIS models, the selection of *highest affinity* ALCs is based on a preset affinity threshold. Thus, the selected subset $\mathcal{H}$ can be the entire set $\mathcal{S}$, depending on the preset affinity threshold.

6. **Calculating the *network affinity***: This is the measurement of *affinity* between two ALCs. The different measures of network affinity are the same as those for *antigen affinity*. A preset network affinity threshold determines whether two or more ALCs are linked to form a network.

7. **Adapting the ALCs in subset $\mathcal{H}$**: Adaptation of ALCs can be seen as the maturation process of the ALC, supervised or unsupervised. Some of the *selection* methods that can be

used are *negative selection* (or *positive selection*), *clonal selection* and/or some evolutionary technique with mutation operators. ALCs that form a *network* can influence each other to adapt to an antigen. These *selection* methods are discussed for each of the AIS models considered in this chapter.

8. **Updating the stimulation level of an ALC**: The stimulation level is calculated in different ways in existing AIS models. In some AIS models, the stimulation level is seen as the summation of antigen affinities. The stimulation level determines the resource level of an ALC. The stimulation level can also be used to determine a selection of ALCs as the *memory* set. The *memory* set contains the ALCs that most frequently match an antigen pattern, thus *memory* status is given to these ALCs. The stimulation level is discussed for the different AIS models in the chapter.

The above listed concepts to model a basic AIS can be grouped into the different layers of an AIS framework as proposed by De Castro and Timmis [34]. The proposed layered AIS framework consists of three main parts [34]:

1. **Representation**: Defining the structure to represent an antigen or receptor (ALC) in the problem domain (search space).

2. **Interaction**: Selecting an affinity function to quantify the *quality* of a structure as defined in the *representation* layer. Typically this function measures the degree of *similarity* or *dissimilarity* between an ALC's (receptor) structure and a structure representing another ALC or antigen.

3. **Adaptation**: Selecting a strategy (immune process or theory) to guide the behaviour of the model.

In the above definition of a layered AIS framework, the structures within the *representation* layer forms part of the input to the *interaction* layer for affinity calculations (quality). In turn, the calculated quality within the *interaction* layer forms part of the input to the *adaptation* layer. Therefore the selected strategy within the *adaptation* layer guides the behaviour of the model based on the calculated affinities. The sections to follow discuss the different existing AIS models within the context of the above layered AIS framework.

4.2 Representation of Antigens and Antibodies

An antigen or receptor can be represented as a string of attributes in the *search space*. Perelson and Oster defined the notion of a *shape space* which is similar to a search space [147, 149]. Assume the shape space ϖ is a bounded region of $\mathcal{R}^N$ with volume V. Each receptor's binding site can be characterised by measuring a number of features. The grouping of these features is known as the receptor's *generalised shape* [147]. If the generalised shape of a receptor is described by a *feature vector* with N features, the generalised shape of the receptor can be represented as a point in N-dimensional space, and therefore an N-dimensional ALC can be represented in shape space ϖ.

Epitopes on the surface of an antigen also have a generalised shape and can be represented in the shape space ϖ. Assume receptor **b** has an exact complementary match to epitope **a**, then the binding affinity between the receptor and epitope is at its highest level. A less exact match results in a lower binding affinity. Thus, the generalised shape of a receptor can *match* more than one epitope with different binding affinity levels [147]. The epitope in ϖ with an exact complement of the receptor's generalised shape represents a region of epitopes with different levels of binding affinities with the receptor (ALC). The region is known as a *recognition region* and all epitopes within this region's radius have an affinity higher than a certain threshold. The radius of the *recognition region* is determined by the affinity threshold. A lower threshold results in a higher radius, which implies a larger region with a less specific binding between an epitope and receptor (ALC).

Figure 4.1 illustrates the shape space ϖ with an ALC **b** and a *recognition region* for a complementary match with antigen **a**. The average volume of the region covered by a *recognition region* in shape space ϖ is defined as $v_{\phi(r)}$ where ϕ is the radius function of the affinity threshold r [149]. From the above definition of a *shape space* with different *recognition regions* for each of the receptors, Perelson and Oster concluded that an infinite number of epitopes (antigen) are recognised by a finite collection or repertoire of receptors (ALCs) [147]. Thus, the initialisation of a repertoire with p random receptors covers a total volume of $p \times v_{\phi(r)}$. With $p \times v_{\phi(r)} > V$, the volume V of *shape space* ϖ is completely covered by the repertoire of ALCs with some overlap between the different *recognition regions* [147, 149].

Measuring the affinity between an ALC and an antigen pattern as a complementary match in-

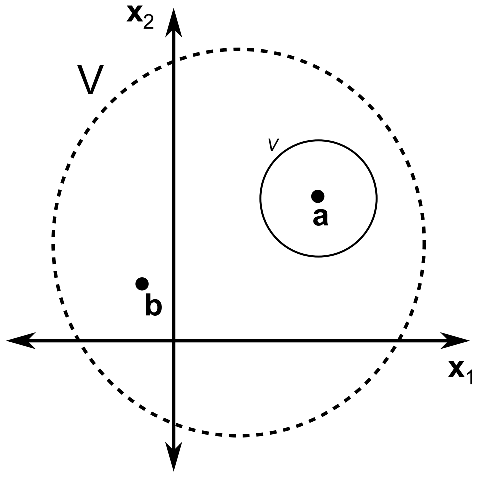


Figure 4.1 Shape space ϖ as bounded region $\mathcal{R}^2$ with volume V

dicates the *dissimilarity* between an ALC and an antigen pattern. The *similarity* between an ALC and antigen pattern can also be measured as the affinity. Thus, measuring the affinity as *similarity*, each ALC in shape space can be presented as a *recognition region* with a certain radius, i.e. affinity threshold. All antigen patterns within the area of an ALC's *recognition region* have a certain measured degree of *similarity* which adheres to the affinity threshold of the ALC.

There are different definitions and approaches to measure the *interaction* between an ALC and antigen pattern or between ALCs, i.e. affinity measurement. Different (*dis*)*similarity* measures for calculating the degree of affinity between an ALC and an antigen/ALC pattern have been proposed. The most commonly used measures of affinity are discussed in the next section.

4.3 Affinity as Quality Measure

Referring to section 3.2, an antibody/receptor binds to an antigen with a certain binding strength known as the *affinity*. The process of affinity maturation (refer to section 3.5), which consists of somatic hyper mutation and clonal selection, improves the affinity of an antibody with the detected antigen. Thus, measuring the affinity between an ALC and an antigen pattern or another ALC gives an indication of the quality (or fitness) of the ALC to *match* an antigen pattern. The adaptation of ALCs to learn the structure of the antigen patterns is guided by measuring the quality of the ALCs.

Affinity in AISs is measured as the spatial distance between an ALC and an antigen pattern or another ALC. The affinity between an ALC and an antigen pattern could be measured against an affinity threshold to determine whether the ALC *matches* the antigen pattern, i.e. whether the antigen pattern is within the radius of the ALC's *recognition region* (as defined in the shape space theory, discussed in section 4.2). Therefore, there are different matching rules based on affinity measurement and thresholding for different shape spaces (problem domains). This section discusses some of the most common matching rules proposed for nominal shape spaces. The *generalised shape* of an ALC or antigen pattern in a nominal shape space consists of features (attributes) which are nominal categories. The different categories for a nominal shape space are known as the *alphabet* of the shape space.

The different distance-based (*dis*)*similarity* measures between feature vectors in continuous shape space were discussed in section 2.2. The distance between two feature vectors in continuous shape space is also measured against an affinity threshold to determine whether the two feature vectors *match*. The most commonly used (*dis*)*similarity* measure in AISs, applied to continuous shape space problems, is the Euclidean distance (as defined in equation (2.3)).

4.3.1 A Complementary Matching Rule

The Hamming distance measures the dissimilarity between two feature vectors in nominal shape space. The Hamming distance between two feature vectors, **a** and **b**, counts the number of positions (features) which are different, as defined in equation (2.10). Therefore, a shape space alphabet of $\{0,1\}$, has feature vectors in binary space and the Hamming distance between these binary vectors counts the number of exclusive-or bits between the corresponding positions (defined as $\sigma(\mathbf{a},\mathbf{b})$ in equation (2.11)). The binary immune system was introduced by Farmer *et al.* [49]. A pattern has a complementary *match* with another pattern if the calculated Hamming distance is greater or equal to an affinity threshold, r, i.e.

$$\sigma(\mathbf{a},\mathbf{b}) \geq r \tag{4.1}$$

Thus, the affinity threshold, r, indicates the least number of differing positions for a pattern to *match* another pattern under the Hamming distance based matching rule. The reader is referred to [93] for an overview of Hamming distance based matching rules.

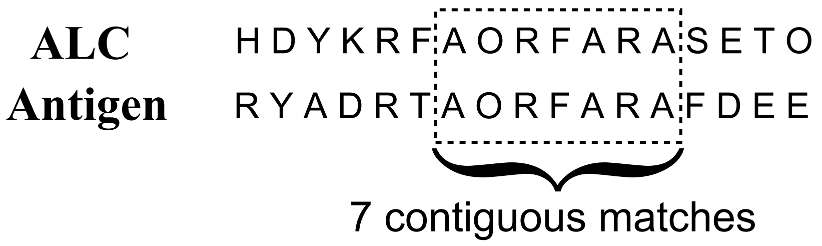

ALC
Antigen
H D Y K R F A O R F A R A S E T O
R Y A D R T A O R F A R A F D E E
7 contiguous matches

Figure 4.2 *r*-contiguous matching rule

4.3.2 The *r*-contiguous Matching Rule

The *r*-contiguous matching rule was proposed by Percus *et al.* [146]. Figure 4.2 illustrates the r-contiguous matching rule between two patterns, **a** and **b**. The r-contiguous matching rule is a partial matching rule. This means that a pattern matches another pattern if there are *r*-contiguous or more matches in the corresponding positions. *r* is the degree of affinity for a pattern to *match* another pattern. In figure 4.2 there are seven contiguous matches between the two patterns. Thus, if $r = 4$, the two patterns match each other in figure 4.2, since $7 > r$. If $r > 7$, there is no match between the patterns in the figure. A higher value of *r* indicates a stronger affinity between two patterns. As illustrated in figure 4.2, the *r*-contiguous matching rule can be seen as a window of width *r*, sliding from the left to the right over two patterns, searching for an *exact* match in the window. The *r*-contiguous matching rule is applied in [53, 183].

4.3.3 The *r*-chunks Matching Rule

The *r*-chunks matching rule is a variation of the above discussed *r*-contiguous matching rule, introduced by Balthrop *et al.* [11]. This matching rule is also known as the *r-contiguous templates* matching rule. The difference between *r*-chunks and the *r*-contiguous matching rule is that *r*-chunks generates template windows of size *r* from pattern **a**. Each template window consists of r-contiguous positions in **a**. A pattern, **b**, is matched by **a** if one of the template windows has an *exact* match by *r*-contiguous positions in **b**. The number of template windows of size *r*, generated from a pattern of size *N* is equal to $(N - r + 1)$ [11]. For example, a pattern, $\mathbf{a} = \langle 100011 \rangle$, in binary space of length 6, can be partitioned into 4 template windows of size 3. The template windows of **a** are $\langle 100 \rangle$, $\langle 000 \rangle$, $\langle 001 \rangle$ and $\langle 011 \rangle$ $(r = 3)$.

Compared to the *r*-contiguous matching rule, instead of sliding a window of size *r* over two patterns to find an exact match within the window, *r*-chunks slides pattern **b** as a window over pattern **a** to align/match r-contiguous positions between the patterns. Thus, the *r*-chunks matching rule generates detectors of length N, consisting of a template window of size r, starting at position i in the detector. An r-chunk detector, $\mathbf{x}$, generated from template window $\langle 001 \rangle$ in the above example with $i = 3$ gives $\mathbf{x} = \langle \#\#001\# \rangle$ where # is a 'no care' symbol. The detector $\mathbf{x}$ matches pattern $\mathbf{a}$. If $i = 1$, then $\mathbf{x} = \langle 001\#\#\# \rangle$ and does not match pattern $\mathbf{a}$.

All of the above matching rules have mostly been used by *negative selection* based AIS models. ALCs trained with *negative selection* (discussed in section 4.4) are self-tolerant. When the set of self patterns, Υ, does not contain all patterns of self during the *censoring* process, the set of self-tolerant ALCs, $\mathcal{B}$, represents a generalised structure which results in some patterns being unmatched by the self-tolerant ALCs [11]. These unmatched patterns are known as *holes* and occur when the above matching rules are applied [42]. Balthrop *et al.* identified and defined two types of *holes*, namely *length-limited holes* and *crossover holes* [11].

Length-limited holes: Length-limited holes occur when applying the r-contiguous matching rule [11]. A length-limited hole, $\mathbf{x}^*$, is a pattern with at least one window of size r that does not exist among the distinct template windows in a set of patterns, Υ, and for which a detector cannot be generated [11]. Let $\Upsilon = \{\langle 0010 \rangle, \langle 1000 \rangle, \langle 0100 \rangle, \langle 1100 \rangle\}$, $N = 4$, $r = 3$ and $\mathbf{h} = \langle 0101 \rangle$. There are two template windows for $\mathbf{x}^*$. These template windows are $\langle 010 \rangle$ and $\langle 101 \rangle$. Therefore a detector starts with template window $\langle 010 \rangle$ and/or ends with template window $\langle 101 \rangle$. A detector that starts with template window $\langle 010 \rangle$ matches patterns in Υ and can therefore not be generated. A detector that ends with template window $\langle 101 \rangle$ can either represent pattern $\langle 0101 \rangle$ or pattern $\langle 1101 \rangle$, which both match patterns in Υ. Therefore detector $\mathbf{x}^*$ cannot be generated.

Crossover holes: Crossover holes occur when applying the r-chunks matching rule [11]. Two template windows are adjacent if the last position of the first template window is the first position of the second template window. A crossover hole, $\mathbf{x}^*$, is a pattern which is not part of a set, Υ, but the template windows of $\mathbf{x}^*$ are adjacent to the distinct template windows of the patterns in Υ [11]. Let $W_i^{\mathbf{x}} = (x_i, x_{i+1}, \ldots, x_{i+r-1})$ be the template window of pattern $\mathbf{x}$ starting at position i in $\mathbf{x}$. A crossover hole occurs between template window $W_i^{\mathbf{x}^*}$ of pattern $\mathbf{x}^*$ and a template window $W_{i+1}^{\mathbf{a}}$ of pattern $\mathbf{a} \in \Upsilon$ if $x_j^* = a_j, \forall j : i+1 \leq j \leq i+r-1$ [11].

4.4 Negative Selection Models

One of the main features in the classical view of the natural immune system is the mature T-Cells, which are self-tolerant, i.e. mature T-Cells have the ability to distinguish between *self* cells and foreign/*non-self* cells. The original negative selection algorithm proposed by Forrest *et al.* [53] is inspired by the maturation process of immature T-Cells in the thymus.

In the original model of Forrest *et al.* [53], all patterns and ALCs are represented as strings with a fixed length, N. The attributes of each string can have any value which is selected from a pre-defined alphabet with size κ. For example, each attribute of a binary string can only have a value of 0 or 1 since the valid values in the binary alphabet is defined as $\{0,1\}$, therefore $\kappa = 2$. A string generated from an alphabet defined as {G,A,T,C} can only have combinations of {G,A,T,C} attribute-values. The number of strings with length N that can be generated from an alphabet with size κ is N^{κ}.

A set of trained ALCs in the model represents the mature T-Cells in the natural immune system. A training set of self patterns is used to train the set of ALCs with the *negative selection* technique. Algorithm 4.2 lists the pseudo code for negative selection, explained in detail below.

For each randomly generated candidate ALC of length N, the affinity between the ALC and each self pattern (also of length N) in the training set is calculated. The affinity between an ALC and a pattern is measured with the r-contiguous matching rule. If the affinity between any self pattern and an ALC is higher than the affinity threshold, r, the candidate ALC is discarded and a new candidate ALC is randomly generated. The new ALC also needs to be measured against the training set of self patterns. If the affinity between all the self patterns and a candidate ALC is lower than the affinity threshold, r, the ALC is added to the self-tolerant set of ALCs. Thus, the set of ALCs is negatively selected, which means that only those ALCs with a calculated affinity less than the affinity threshold, r, will be included in the set of self-tolerant ALCs. This phase is known as *censoring*.

Algorithm 4.2: Training ALCs with *negative selection*

```
Set counter $b$ as the number of self-tolerant ALCs to train;
Create empty set of self-tolerant ALCs as $\mathcal{B}$;
Determine the training set of self patterns as $\Upsilon$;
while size of $\mathcal{B}$ not equal to $b$ do
    Randomly generate a candidate ALC, $\mathbf{x}$;
    matched=false;
    for each self pattern $\mathbf{s} \in \Upsilon$ do
        if affinity between $\mathbf{x}$ and $\mathbf{s}$ is higher than affinity threshold $r$ then
            matched=true;
            break;
        end
    end
    if not matched then
        Add $\mathbf{x}$ to set $\mathcal{B}$;
    end
end
```

The trained, self-tolerant set of ALCs is then presented with a test set of self and non-self patterns for classification. This phase is known as *monitoring*. The affinity between each training pattern and the set of self-tolerant ALCs is calculated. If the calculated affinity is below the affinity threshold, r, the pattern is classified as a self pattern; otherwise the pattern is classified as a non-self pattern. The training set is monitored by continually testing the ALC set against the training set for changes. A number of drawbacks of the proposed negative selection model are that,

- the training set needs to have a good representation of self patterns,
- an increase in the number of self patterns exponentially increases the number of randomly generated candidate ALCs [111],
- there is an exhaustive replacement of an ALC during the censoring of the training set until the randomly generated ALC is self-tolerant, and
- there is no validation/removal of redundant ALCs.

The above listed drawbacks were also highlighted by Ayara *et al.* [7]. Alternative *censoring* approaches to generate self-tolerant ALCs have been proposed to address some of the above drawbacks of the original model. These alternative models are briefly discussed next.

Linear model: The first of these alternative *censoring* models is the *linear* model proposed by D'haeseleer *et al.* [42]. The linear model runs in linear time with respect to the size of the self set, given that the string length, N, and the matching affinity threshold, r, are fixed. Binary strings are generated from a binary alphabet $\{0,1\}$. The model generates different matching templates to determine the number of unmatched strings in the self set. A template is a string of length N, where r contiguous positions of the template are set to a value. The remaining $N-r$ positions are set to 'no care' symbols. A set of self-tolerant ALCs are then randomly selected from the unmatched templates.

Greedy model: D'haeseleer *et al.* also proposed the *greedy* model [42]. The difference between the *greedy* model and the previously discussed *linear* model is that ALCs in the self-tolerant set, $\mathcal{B}$, are not randomly selected from the set of unmatched template strings. The selected set of self-tolerant ALCs have minimal overlap among each other and maximum coverage of the non-self space.

Binary tree template and the discriminative power: The templates which are used in the *greedy* model to generate self-tolerant ALCs can be assembled to form different binary trees. This results in the formation of general subtrees (templates) which reduces the number of self-tolerant ALCs to cover most of the patterns in non-self space. The formation of binary trees by these templates was observed and proposed by Wierchon, i.e. binary tree templates [182]. As discussed in section 4.3, compared to the hamming distance, the r-contiguous matching rule is symmetric and reflexive. Thus, Wierchon investigated the discriminative power of a candidate ALC containing a template which is matched by the r-contiguous matching rule [183]. The discriminative power of a candidate ALC is defined as the number of unique strings matched by the ALC using the r-contiguous matching rule [183].

NSMutation: This model differs in the *censoring* process of candidate ALCs by not immediately discarding a candidate ALC when matched with a certain affinity to a self pattern. Instead, guided mutation is performed on the self-matching candidate ALC pattern, away from the matched self pattern. This model was proposed by De Castro and Timmis [34] and is inspired by the process of affinity maturation in the natural immune system. Since the r-contiguous matching rule is applied, the candidate ALC pattern is only mutated in the r matching positions. The probability of mutation is proportional to the affinity with the self pattern. Thus a higher affinity between the self-matching ALC and the self pattern results in a higher rate of mutation and vice versa.

Mutation is performed on a self-matching candidate ALC pattern for only a number of times, the lifetime of a candidate ALC. The mutated candidate ALC is discarded if the specified lifetime is reached with no improvement. If a mutated candidate ALC does not match any self pattern, the mutated ALC is then added to the set of self-tolerant ALCs.

Evolutionary approaches: A different approach is proposed by Kim and Bentley [110] where candidate ALCs are not randomly generated and tested with negative selection, but an evolutionary process is used to evolve ALCs towards non-self and to maintain diversity and generality among the ALCs. The model by Potter and De Jong [150] applies a co-evolutionary genetic algorithm to evolve ALCs towards the selected class of non-self patterns in the training set and further away from the selected class of self patterns. Once the fitness of the ALC set evolves to a point where all the non-self patterns and none of the self patterns are detected, the ALCs represent a description of the concept. If the training set of self and non-self patterns is noisy, the ALC set will be evolved until most of the non-self patterns are detected and as few as possible self patterns are detected. The evolved ALCs can discriminate between examples and counter-examples of a given concept. Each class of patterns in the training set is selected in turn as self and all other classes as non-self to evolve the different concepts in the training set.

Gonzalez *et al.* [60] present a negative selection method which is able to train ALCs with continuously-valued self patterns. The ALCs are evolved away from the training set of self patterns and are well separated from one another to maximise the coverage of non-self. This results in the least possible overlap among the evolved set of ALCs. A similar approach is presented in the GAIS model of Graaff and Engelbrecht [63]. All patterns are represented as binary strings and the Hamming distance is used as affinity measure. A genetic algorithm is used to evolve ALCs away from the training set of self patterns towards a maximum non-self space coverage and a minimum overlap among existing ALCs in the set. The difference to the model of Gonzalez *et al.* [60] and the original negative selection model of Forrest *et al.* [53] is that each ALC in the set has a local affinity threshold. The ALCs are trained with an adapted negative selection method as illustrated in figure 4.3.

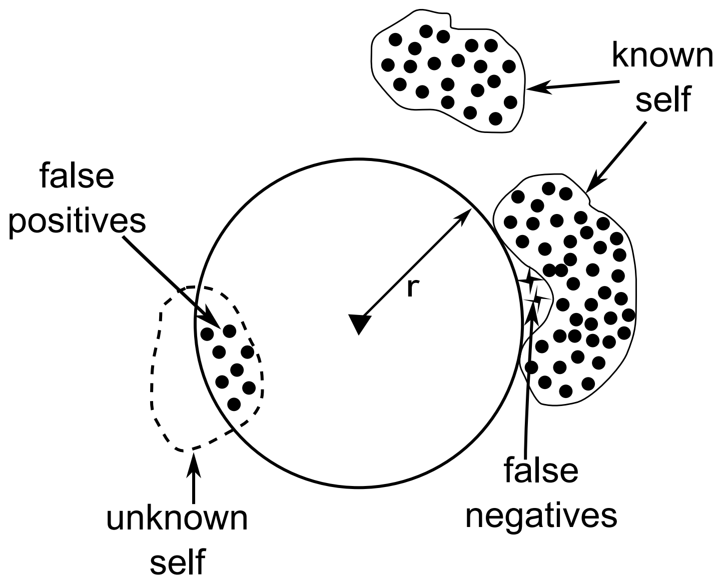


Figure 4.3 Adapted Negative Selection

With the adapted negative selection method the affinity threshold, *r*, of an ALC is determined by the distance to the closest self pattern from the ALC. The affinity threshold, *r*, is used to determine a match with a non-self pattern. Thus, if the measured affinity between a pattern and an ALC is less than the ALC's affinity threshold, *r*, the pattern is classified as a non-self pattern. The adaptive negative selection method is inspired by the definition of epitope-volumes in *shape space* (as discussed in section 4.2). Figure 4.3 also illustrates the drawback of *false positives* and *false negatives* when the ALCs are trained with the adapted negative selection method. These drawbacks are due to an incomplete static self set. The *known self* is the incomplete static self set that is used to train the ALCs and the *unknown self* is the self patterns that are not known during training. The *unknown self* can also represent self patterns which are outliers to the set of *known self* patterns.

Surely all evolved ALCs will cover non-self space, but not all ALCs will detect non-self patterns. Therefore, Graaff and Engelbrecht [62, 63] proposed a transition function, the life counter function, to determine an ALC's status. ALCs with annihilated status are removed in an attempt to have only mature and memory ALCs with optimum classification of non-self patterns.

The V-detector model proposed by Ji and Dasgupta [99] as an alternative to the model of Gonzalez *et al.* [60] for continuously valued patterns in Euclidean space is similar to the model of

Graaff and Engelbrecht [61, 63], in that each generated ALC has a local affinity threshold which is determined by calculating the distance between the generated ALC and the closest self pattern. The V-detector model generates ALCs of variable length in continuous space compared to the fixed length ALCs in Hamming space of the proposed model by Graaff and Engelbrecht [63].

Smith *et al.* derived analogies between the memory capabilities of the immune system and the method of sparse distributed memory (SDM) [161]. SDM was proposed by Kanerva as a method to store a large number of large binary patterns using a small number of physical data addresses [105]. These physical addresses are known as physical or hard locations and binary patterns are stored in these locations in such a manner that any data can be accurately recalled. An SDM is composed of a set of these hard locations. Each location has a recognition radius. A location recognises data if the distance to the data is within the recognition radius of the location.

In addition, each location also has a set of counters, each representing a bit in the location. The counters are used to determine whether a recalled bit from the memory should be set to '1' or '0'. A data pattern is distributed to all locations which recognise it. If a location recognises a data pattern, the counter of each bit is incremented by '1' if the data pattern's corresponding bit is '1' and decremented by '1' if the data pattern's corresponding bit is '0'. When a pattern needs to be recalled from the memory, the counters of locations recognising the pattern are summed. The corresponding bit of the recalled pattern is set to '1' if the summed counters of that bit is greater than or equal to zero; otherwise the bit is set to '0'.

Inspired by the analogies between SDM and the immune system, Hart *et al.* proposed a co-evolutionary SDM (COSDM) model to cluster non-stationary data [77]. In COSDM an antigen represents the data pattern that needs to be stored and an ALC represents a hard location. The recognition radius of an ALC determines the size of a cluster. COSDM consists of a number of ALC populations. A co-evolutionary genetic algorithm was used to find the set of ALCs as well as the size of their individual recognition radii, which best clustered the current available data. Some of the drawbacks of COSDM were that the algorithm was relatively slow and had difficulty to set the correct recognition radius for each ALC [78].

Hart *et al.* proposed the self-organising SDM (SOSDM) to address the drawbacks of COSDM [76, 78]. Hart *et al.* highlighted the unsuitability of Kanerva's SDM to cluster data in [77] and based the SOSDM on an alternative SDM model as proposed by Hely *et al.* [82]. In SOSDM, an

ALC binds to an antigen pattern if the binding strength between the ALC and the antigen pattern is greater than a specified threshold. The binding strength is proportional to the calculated affinity between an ALC and an antigen pattern. The affinity is measured as the Hamming distance between an ALC and an antigen pattern. The binding strength of each ALC is calculated as the ratio of the calculated affinity to the maximum affinity in the set of ALCs. Therefore, the binding strength of an ALC with maximum affinity to an antigen pattern is one. Each ALC's set of counters is updated with the binding strength to an antigen pattern, where the binding strength is proportional to the calculated affinity.

In addition, each ALC also measures an accumulated error between an ALC (hard location) and all antigens presented to the ALC. The accumulated error of an ALC is updated with each antigen pattern that binds to the ALC. After all antigen patterns have been presented to the set of ALCs, the average error of each ALC is used to self organise the set of ALCs. This is done in such a manner that ALCs move towards positions in the search space, such that the average error is minimised. An ALC's associated set of counters decays over time. The results obtained by SOSDM on clustering stationary data are scalable with the size of the data set and with the length of the antigen pattern [78]. Clustering of non-stationary data delivered promising results, but highlighted a decrease in performance with more dynamic data [78]. SOSDM is an adaptive, scalable and self-organising model.

4.5 Clonal Selection Models

The natural immune system is able to adapt to unseen antigens and capable of keeping a memory of frequently encountered antigens. This is achieved by a process of affinity maturation which consists of clonal selection with somatic hyper mutation (as discussed in section 3.5). The former is the process of selecting the most stimulated (highest affinity) lymphocytes for clonal proliferation, and the latter the mutation process on these clones. The increase in size of some clones results in a decrease in size of other previously cloned lymphocytes, since the natural immune system regulates the total number of lymphocytes in the body. Clones are mutated in an attempt to have a higher affinity with the encountered antigen.

Frequently selected lymphocytes (through clonal selection) transition into a state of memory and these memory lymphocytes are used in a faster secondary response to frequently encountered antigens (as discussed in section 3.5). Those lymphocytes which are seldom selected (or never stimulated) transition into a state of annihilation and is eventually replaced by the natural immune system with newly generated lymphocytes. Through the process of affinity maturation the natural immune system is able to learn new antigens, keep a memory of frequently encountered antigens, and integrate newly generated lymphocytes. The learning capability of the natural immune system inspired the modelling of clonal selection with somatic hyper mutation in AISs.

Clonal selection in AISs is the selection of a set of ALCs with the highest calculated affinity with an antigen pattern. The selected ALCs are then cloned and mutated in an attempt to have a higher binding affinity with the presented antigen pattern. The mutated clones compete with the existing set of ALCs, based on the calculated affinity between the mutated clones and the antigen pattern, for survival to be exposed to the next antigen pattern. This section discusses some of the AIS models inspired by the clonal selection theory.

CLONALG: The CLONALG model is a general implementation of the clonal selection theory and was introduced by De Castro and Von Zuben as an algorithm that can perform machine-learning and pattern recognition tasks [35, 38]. ALCs and antigen patterns are presented as binary strings and therefore the affinity between an ALC and an antigen pattern is measured with the Hamming distance. A lower Hamming distance implies a higher affinity.

CLONALG evolves a population of randomly initialised ALCs over a number of generations to have a higher affinity with the presented antigen patterns. The population of ALCs is partitioned into a subset of memory ALCs and a remaining subset of ALCs (non-memory ALCs). CLONALG assumes that there is an ALC in the memory subset for each antigen pattern that needs to be recognised.

In a generation, each of the antigen patterns is presented to the population of ALCs. A number of highly stimulated ALCs (those with highest affinity with the presented antigen) are then selected for cloning. The number of clones generated for an ALC is directly proportional to the calculated affinity and is calculated as [38]

$$\eta(\mathbf{b}_i) = round\left(\frac{\Theta \times |\mathcal{B}|}{i}\right) \tag{4.2}$$

where Θ is a multiplying factor, *round* is a function that rounds a floating-point value to the closest integer, and *i* is the position of the ALC in the sorted set of highly stimulated ALCs (sorted in ascending order of affinity). Each of the clones is then mutated at a rate which is inversely proportional to the affinity. This means that clones from highly stimulated ALCs are mutated less than clones from less stimulated ALCs. The mutated clone with the highest affinity with the presented antigen replaces the ALC in the memory subset of the population if its corresponding memory ALC has a lower measured affinity. A percentage of the ALC population with the lowest affinities is replaced by randomly generated ALCs.

A modified version of CLONALG has been applied to multi-modal function optimisation [35]. In the optimisation model the entire population of ALCs is seen as the memory set (no subset of memory ALCs). All the ALCs are cloned with equal size, changing equation (4.2) to [38]

$$\eta(\mathbf{b}_i) = round\left(\Theta \times |\mathcal{B}|\right) \tag{4.3}$$

The affinity of an ALC is calculated as the objective function that needs to be optimised, since there are no antigen patterns to present to the population. The ALCs with the highest affinity is selected as the population for the next generation. The population of ALCs for the next generation is selected from the population of the previous generation and the mutated clones of ALCs.

DynamiCS: In some cases the problem that needs to be optimised consists of self patterns that change through time. To address these types of problems, the dynamic clonal selection algorithm (DCS) was introduced by Kim and Bentley [114]. The dynamic clonal selection algorithm is based on the AIS proposed by Hofmeyr [85]. The basic concept in [85] is to have three different populations of ALCs, categorised into immature, mature and memory ALC populations.

Kim and Bentley explored the effect of three parameters on the adaptability of the model to changing self [114]. These parameters were the tolerisation period, the activation threshold and the life span. The tolerisation period is a threshold on the number of generations during which ALCs can become self-tolerant. The activation threshold is used as a measure to determine if a mature ALC met the minimum number of antigen matches to be able to become a memory ALC. The life span parameter indicates the maximum number of generations that a mature ALC is allowed to be in the system.

If the mature ALC's life span meets the pre-determined life span parameter value, the mature ALC is deleted from the system. Experimental results with different parameter settings indicated that an increase in the life span with a decrease in the activation threshold resulted in the model to have an increase in detecting true non-self patterns. An increase in the tolerisation period resulted in less self patterns being detected falsely as non-self patterns, only if the self patterns were stable.

With a changing self the increase in tolerisation period had no remarkable influence in the false detection of self patterns as non-self patterns. Although the DCS could incrementally learn the structure of self and non-self patterns, it lacked the ability to learn any changes in unseen self patterns. The memory ALCs in the DCS algorithm had infinite lifespan. This feature was omitted in the extended DCS by removing memory ALCs which were not self-tolerant to newly introduced self patterns [112].

DCS was further extended by introducing *hyper mutation* on the deleted memory ALCs [113]. The deleted memory ALCs were mutated to seed the immature detector population, i.e. deleted memory ALCs form part of a gene library. Since these deleted memory ALCs contain information (which was responsible for giving them memory status), applying mutation on these ALCs will retain and fine tune the system, i.e. reinforcing the algorithm with previously trained ALCs.

MARIA: A different model, though similar to the above DCS model with regards to the three populations used, is the MARIA model proposed by Knight and Timmis [116]. The model consists of multiple layers and addresses some of the shortfalls of the AINE model [169] (discussed in section 4.6). The defined layers interact to adapt and learn the structure of the presented antigen patterns. The model consists of three layers which fulfills different roles in the adaptation process.

All patterns in the training set are seen as antigens. The affinity between an antigen pattern, $\mathbf{a}_j$, and a cell within a layer is measured using the Euclidean distance, σ. The different layers in sequential order are: the free antibody layer ($\mathcal{F}$), the B-Cell layer ($\mathcal{B}$) and the memory layer ($\mathcal{M}$). Each layer has an affinity threshold and a death threshold. The affinity threshold determines whether an antigen binds to a cell within a specific layer. The death threshold is the maximum elapsed time for a cell not to be stimulated. This means that if the length of time since a cell was last stimulated exceeds the death threshold of the specific layer, the cell dies and is removed from the population in the specific layer. The B-Cell layer has an additional stimulation threshold which determines whether a cell in the specific layer is cloned.

Each antigen pattern is first presented to a random selection of cells in $\mathcal{F}$. The number of free antibodies that bind to the antigen pattern is calculated as $free\_binding$. The antigen, $\mathbf{a}_j$, is then randomly presented to the B-Cells in $\mathcal{B}$ until one of the B-Cells, $\mathbf{b}_i$, binds to the antigen pattern. $\mathbf{b}_i$ is cloned as $\mathbf{b}_i^*$ if the calculated $free\_binding$ in $\mathcal{F}$ exceeds the stimulation threshold. The clone $\mathbf{b}_i^*$ is then mutated as $\mathbf{b}_i^{'}$ and added to $\mathcal{B}$. Mutated clones of $\mathbf{b}_i$ are then added to $\mathcal{F}$. The number of mutated clones (or rather free antibodies) produced by a stimulated B-Cell is given in [116] as

$$\eta\left(\mathbf{a}_j, \mathbf{b}_i\right) = \left(\sigma_{max} - \sigma\left(\mathbf{a}_j, \mathbf{b}_i\right)\right) \times k \tag{4.4}$$

where η is the number of antibodies that are added to the free-antibody layer, σ_{max} is the maximum possible Euclidean distance between a B-Cell and an antigen pattern in the data space (i.e. lowest possible affinity), and k is some constant.

If none of the B-Cells in $\mathcal{B}$ bind to the antigen pattern, a new B-Cell is created with the same presentation as the unbinded antigen. The new B-Cell is added to $\mathcal{B}$ resulting in a more diverse coverage of antigen data. The new B-Cell also produces mutated clones as free antibodies, which are added to the free-antibody layer.

The final layer, $\mathcal{M}$, only consists of memory cells and only responds to new memory cells. The generated clone, $\mathbf{b}_i^*$, in $\mathcal{B}$ is presented as a new memory cell to $\mathcal{M}$. The memory cell with the lowest affinity to $\mathbf{b}_i^*$ is selected as $\mathbf{m}_{min}$. $\mathbf{m}_{min}$ is replaced by $\mathbf{b}_i^*$ if the affinity between $\mathbf{b}_i^*$ and $\mathbf{m}_{min}$ is lower than the affinity threshold of the specific layer, and the affinity of $\mathbf{b}_i^*$ is less than the affinity of $\mathbf{m}_{min}$ with the antigen that was responsible for the creation of the new memory cell. If the affinity between $\mathbf{b}_i^*$ and $\mathbf{m}_{min}$ is higher than the affinity threshold, $\mathbf{b}_i^*$ is added to the memory layer. The multi-layered model, compared to the SSAIS model [140] (discussed in section 4.6), obtained better compression on data while forming stable clusters.

AIRS: A supervised learning AIS algorithm, the Artificial Immune Recognition System (AIRS) [178, 179] borrowed the concept of an artificial recognition ball (ARB) population within a resource limited environment as proposed by the network based resource limited AIS (AINE) [169]. Contrary to AINE and other unsupervised network based AIS algorithms (as discussed in section 4.6), AIRS does not model any network interactions between ARBs. Furthermore, most unsupervised network based AIS models are applied to the problem of data clustering whereas AIRS is an AIS classifier.

The ARBs in AIRS also compete for resources to survive. The ARBs undergo a clonal expansion and maturation process to evolve a set of memory ARBs which represents the different classes of the training patterns (antigens). The evolved set of memory ARBs is used to classify unseen patterns into multiple classes. The definition of an ARB and the concept of a resource limited environment are discussed in more detail in the AINE paragraph in the next section.

4.6 Idiotypic Network Models

A number of different theoretical network based models have been proposed by immunologists to formulate and capture the characteristics and interactions of the natural immune network system. One of these theoretical network based models was proposed by Farmer *et al.* [49]. Farmer *et al.* exploited the fundamental concepts of the network theory as proposed by Jerne [97] and proposed a simple model to simulate the dynamics of the natural immune network system and its memory capability [49]. Perelson also proposed a model to simulate the dynamics and production of a network based immune network system [148]. The theoretical model proposed by Farmer *et al.* is discussed next, since the earliest work in artificial immune systems are based on this model. The rest of the section discusses different network theory inspired AIS models.

The theory of clonal selection as part of the process of affinity maturation assumes that all immune responses are activated by encountered antigens. As explained in section 3.5, antigens *select* those lymphocytes with which the antigens have the highest binding affinity, resulting in clonal proliferation and somatic hyper mutation of the selected lymphocytes. As a result of somatic hyper mutation on the clones, the variable regions of the clones can become antigenic and invoke an immune response from neighbouring lymphocytes (as discussed in section 3.6). The recognition of idiotopes results in interconnected neighbouring lymphocytes, forming an idiotypic network.

Thus, lymphocytes in a network co-stimulate and/or co-suppress each other in reaction to an antigen. Therefore a lymphocyte is not only stimulated by an antigen, but also by neighbouring lymphocytes (as discussed in section 3.6). This results in the annihilation of some lymphocytes and the introduction of mutated lymphocyte clones into the population of lymphocytes. Highly stimulated lymphocytes remain part of the population whereas less stimulated lymphocytes are replaced/removed from the population. The population of lymphocytes is dynamic in such a way that the concentration of antibodies/lymphocytes at different points in time differ. In order to formulate the change in concentration of the population, based on the stimulation of the lymphocytes, Farmer *et al.* [49] identified three factors which influence the stimulation of a lymphocyte. These are:

- the binding affinity with the encountered antigen,
- the stimulation received from neighbouring lymphocytes, and
- the suppression received from neighbouring lymphocytes.

The model of Farmer *et al.* defines a lymphocyte, $\mathbf{b}$, as two binary strings which represent the lymphocyte's epitope, $\mathbf{e}$, and paratope, $\mathbf{p}$. The change in concentration, ν, of a lymphocyte, $\mathbf{b}_i$, relative to time, t, is simulated with the following differential equation as proposed in [49]:

$$\frac{d\nu(\mathbf{b}_i)}{dt} = c\left[\sum_{j=1}^{|\mathcal{B}|} m_{j,i}\nu(\mathbf{b}_i)\nu(\mathbf{b}_j) - k_1\sum_{j=1}^{|\mathcal{B}|} m_{i,j}\nu(\mathbf{b}_i)\nu(\mathbf{b}_j) + \sum_{j=1}^{|\mathcal{A}|} m_{j,i}\nu(\mathbf{b}_i)\nu(\mathbf{a}_j)\right] - k_2\nu(\mathbf{b}_i) \tag{4.5}$$

where $|\mathcal{B}|$ is the number of lymphocytes and $|\mathcal{A}|$ the number of antigens. $m_{i,j}$ denotes the interaction strength between epitope $\mathbf{e}_i$ of lymphocyte $\mathbf{b}_i$ and paratope $\mathbf{p}_j$ of lymphocyte $\mathbf{b}_j$. The interaction strength (affinity) between two lymphocytes is calculated as the complementary match between the respective paratope and epitope strings. Two lymphocytes interact (bind) if their calculated interaction strength, $m_{i,j}$, is above a certain threshold.

In the above differential equation, the first term signifies the stimulation of paratope $\mathbf{p}_i$ by epitope $\mathbf{e}_j$; the second term represents the suppression of lymphocyte $\mathbf{b}_i$ whose epitope $\mathbf{e}_i$ is recognised by paratope $\mathbf{p}_j$; and the third term signifies the recognition of antigen $\mathbf{a}_j$. k_1 is a constant which regulates the inequality between stimulation and suppression and k_2 is the rate of annihilation. The constant rate c depends on the rate of lymphocyte/antibody production which is stimulated by an interaction. Therefore, c also depends on the number of interactions per time unit. Cloning and mutation of lymphocytes are proportional to the stimulation level. A higher stimulated lymphocyte produces more clones. This results in a diverse set of lymphocytes.

Hunt and Cooke developed a network based AIS model for classification of DNA strings into promoter or non-promoter classes [30, 93]. Each ALC in the model consists of a binary string which represents the ALC's paratope, a library of genes from which antibodies are generated, the DNA sequence and the level of stimulation. The antibodies are used for classification of unseen

DNA sequences. The affinity between two ALCs (or between an ALC and antigen) is calculated as the complementary match between the respective paratope and epitope strings. The model makes use of an affinity threshold to determine whether ALCs are linked to form a network or whether an ALC binds to an antigen. If the calculated affinity is greater than the affinity threshold, binding/linking occurs.

The population of ALCs is initialised by a cross-section of randomly selected DNA sequences from the training set. The remainder of the training set is used as antigen patterns. Antigen patterns are randomly selected and presented to a randomly selected ALC. The antigen is then presented to a percentage of the ALC's linked neighbours to determine whether any of the linked ALCs can bind to the antigen. If an ALC binds to the antigen, the ALC's stimulation level is calculated. The stimulation level of an ALC determines whether the ALC becomes active. If none of the linked ALCs are activated, an ALC is generated by using the presented antigen as template and added to the population of ALCs. Activated ALCs are cloned and mutated. Cloned ALCs are integrated into the network at the ALCs with which the clones have the highest affinity.

The stimulation level of an ALC, **b**, is based on the differential equation as proposed by Farmer *et al.* [49] (as defined in equation (4.5)). Cooke and Hunt adapted equation (4.5) such that

$$\vartheta(\mathbf{b}) = c\left[\sum_{j=1}^{|\mathcal{B}|} m\left(\mathbf{b},\mathbf{e}_j\right) - k_1 \sum_{j=1}^{|\mathcal{B}|} m\left(\mathbf{b},\mathbf{p}_j\right) + k_2 \sum_{j=1}^{|\mathcal{A}|} m\left(\mathbf{b},\mathbf{a}_j\right)\right] - k_3 \tag{4.6}$$

where $|\mathcal{B}|$ is the number of lymphocytes, $|\mathcal{A}|$ is the number of antigens, and m denotes the affinity between ALC **b** and the paratope **p** (or epitope **e**) of linked ALC $\mathbf{b}_j$.

The DNA classification model of Hunt and Cooke [30, 93] was improved and applied to case base reasoning [91, 92]. Each ALC in the model represents a case. ALCs are linked as a network if they represent similar cases, which could result in generalised cases. These generalised cases represented trends in data. The model was also applied to data mining [89], but there were however a few drawbacks which are discussed next.

The increasing size of the network made the model less scalable and a randomly initialised network of ALCs increased the time to built generalised cases. These drawbacks were addressed in [89] and the improved model was applied to fraud detection [89, 90, 141]. The proposed fraud detection system in [90] was called JISYS and implemented different matching techniques compared to the original model in [89]. Although JISYS delivered promising results, the model was limited to a specific domain of mortgage fraud detection.

Timmis developed a basic network based AIS model which was domain-independent [165]. The basic AIS model was based on the work of Hunt and Cooke [93]. The model performed cloning and mutation operations on a set of linked ALCs. The affinity between an ALC and an antigen pattern (or another ALC) is measured using the Euclidean distance. Two ALCs are linked if the calculated affinity between them is below the network affinity threshold (NAT). The model was able to produce three distinct clusters when applied to Fisher's Iris data set [51]. Timmis also developed a tool named aiVis to visualise the formed clusters [166].

A few drawbacks to the basic AIS model were highlighted in [169]. A drawback is that there is an exponential growth in the size of the network due to the incapability of the proposed mechanism to control the size of the ALC population. The exponential growth of the network also resulted in an unscalable model, increasing the computational complexity with each iteration. Furthermore, the formed networks are difficult to interpret. The model is also very sensitive to the NAT value. The authors proposed the Resource Limited AIS as an improvement to the basic AIS model, discussed below [169].

Another approach to enhance the model of Timmis was proposed by Wierchoń and Kużelewska [184]. The model of Timmis [165] was adapted in such a way that the set of ALCs is randomly initialised. Furthermore, the network affinity threshold in [184] is calculated as the average distance between the $|\mathcal{A}| \times k$ lowest distances in the antigen set, where $|\mathcal{A}|$ is the size of the antigen set and k some constant. The adapted model in [184] also improves on the model in [165] in that the maximum network size is limited to the number of training patterns in the training set and stable clusters are formed with a minimal number of control parameters.

Resource Limited AIS (AINE): AINE presented a new concept of artificial recognition balls (ARBs), bounded by a resource limited environment. A resource limited environment is defined as the maximum number of available B-Cells that is shared among ARBs in a population. Thus, each ARB allocates a number of resources based on the ARB's overall stimulation level. In summary, AINE consists of a population of ARBs, links between the ARBs, a set of antigen training patterns (of which a cross section is used to initialise the ARBs) and some clonal operations for learning. An ARB represents a region of antigen space that is covered by a certain type of B-Cell.

ARBs which are close to each other (similar) in antigen space are connected with weighted edges to form a number of individual network structures. The similarity (affinity) between ARBs and between the ARBs and an antigen is measured using Euclidean distance. Two ARBs are connected if the affinity between them is below the network affinity threshold (NAT). The NAT is the average distance between all the antigen patterns in the training set, calculated as [168]

$$NAT = \frac{2k}{|\mathcal{A}| \times (|\mathcal{A}| - 1)} \sum_{i=1}^{|\mathcal{A}|-1} \sum_{j=i+1}^{|\mathcal{A}|} \sigma\left(\mathbf{a}_i, \mathbf{a}_j\right) \tag{4.7}$$

where σ is the Euclidean distance and k is a constant value such that $0 \leq k \leq 1$. The value of the NAT determines the linking between ARBs and therefore influences the number of formed networks. Algorithm 4.3 lists the pseudo code for AINE.

For each iteration, all training patterns in set $\mathcal{A}$ are presented to the set of ARBs, $\mathcal{B}$. After each iteration, each ARB, $\mathbf{b}$, calculates its stimulation level, ϑ, and allocates resources (B-Cells) based on its stimulation level as defined in equation (4.12). The stimulation level, ϑ, of an ARB, $\mathbf{b}$, is calculated as the summation of the antigen stimulation, ps, the network stimulation, ns, and the network suppression, nn. The stimulation level of an ARB is defined as follows [169]

$$\vartheta(\mathbf{b}) = ps(\mathbf{b}) + ns(\mathbf{b}) + nn(\mathbf{b}) \tag{4.8}$$

$$ps(\mathbf{b}) = \sum_{i=1}^{|\alpha_{\mathbf{b}}|} 1 - \alpha_i \tag{4.9}$$

$$ns(\mathbf{b}) = \sum_{j=1}^{|\lambda_{\mathbf{b}}|} 1 - \lambda_j \tag{4.10}$$

$$nn(\mathbf{b}) = -\sum_{j=1}^{|\lambda_{\mathbf{b}}|} \lambda_j \tag{4.11}$$

where $|\alpha_{\mathbf{b}}|$ is the normalised set of affinities between an ARB, $\mathbf{b}$, and all antigen $\mathbf{a} \in \mathcal{A}$ for which $\sigma(\mathbf{a}_i, \mathbf{b}) < NAT$. The antigen stimulation, ps, is thus the sum of all antigen affinities below the NAT threshold and $0 \leq \alpha_i \leq 1$; $\alpha_i \in \alpha_{\mathbf{b}}$. The network stimulation, ns, and the network suppression, nn, are the sum of affinities between an ARB and all the ARB's connected neighbours, as defined in equation (4.10) and equation (4.11) respectively. In equations (4.10) and (4.11), $|\lambda_{\mathbf{b}}|$ is the normalised set of affinities between an ARB, $\mathbf{b}$, and all other ARBs in set $\mathcal{B}$. The ns and nn terms are based on the summation of the distances to the $|\lambda_{\mathbf{b}}|$ linked neighbours of an

Algorithm 4.3: Artificial Immune Network (AINE)

```
Normalise the training data;
Initialise the ARB population, $\mathcal{B}$, using a randomly selected cross section of the
normalised training data;
Initialise the antigen set, $\mathcal{A}$, with the remaining normalised training data;
Set the maximum number of available resources, $R_{max}$;
for each ARB, $\mathbf{b}_i \in \mathcal{B}$, at index position i in $\mathcal{B}$ do
    for each ARB, $\mathbf{b}_j \in \mathcal{B}$, at index position j in $\mathcal{B}$ do
        Calculate the ARB affinity, $\sigma\left(\mathbf{b}_i, \mathbf{b}_j\right)$;
        if $\sigma\left(\mathbf{b}_i, \mathbf{b}_j\right) < NAT$ and $i \neq j$ then
            Add $\sigma\left(\mathbf{b}_i, \mathbf{b}_j\right)$ to the set of network stimulation levels, $\lambda_{\mathbf{b}_i}$;
        end
    end
end
while not stopping condition do
    for each antigen, $\mathbf{a}_i \in \mathcal{A}$, at index position i in $\mathcal{A}$ do
        for each ARB, $\mathbf{b}_j \in \mathcal{B}$, at index position j in $\mathcal{B}$ do
            Calculate the antigen affinity, $\sigma\left(\mathbf{a}_i, \mathbf{b}_j\right)$;
            if $\sigma\left(\mathbf{a}_i, \mathbf{b}_j\right) < NAT$ then
                Add $\sigma\left(\mathbf{a}_i, \mathbf{b}_j\right)$ to the set of antigen stimulation levels, $\alpha_{\mathbf{b}_j}$;
            end
        end
    end
    Allocate resources (see algorithm 4.4) to the set of ARBs, $\mathcal{B}$;
    Clone and mutate the remaining ARBs in $\mathcal{B}$;
    Integrate mutated clones into $\mathcal{B}$;
end
```

ARB, **b**. The network suppression, *nn*, is the dissimilarity between an ARB and neighbouring ARBs in the network. Network suppression keeps the size of the ARB population under control.

Algorithm 4.4: Resource allocation in AINE

```
Set the number of allocated resources, $R_T = 0$;
for each ARB, $\mathbf{b}_i \in \mathcal{B}$, at index position $i$ in $\mathcal{B}$ do
    Allocate resources, $R(\mathbf{b}_i)$;
    $R_T = R_T + R(\mathbf{b}_i)$;
end
Sort the set of ARBs, $\mathcal{B}$, in ascending order of $R$;
if $R_T > R_{max}$ then
    $z = R_T - R_{max}$;
    for each ARB, $\mathbf{b}_i \in \mathcal{B}$, at index position $i$ in $\mathcal{B}$ do
        $q = R(\mathbf{b}_i)$;
        if $q = 0$ then
            Remove $\mathbf{b}_i$ from set $\mathcal{B}$;
        end
        else
            $q = q - z$;
            if $q \leq 0$ then
                Remove $\mathbf{b}_i$ from set $\mathcal{B}$;
                $z = -q$;
            end
            else
                $R(\mathbf{b}_i) = q$;
                break;
            end
        end
    end
end
```

Algorithm 4.4 lists the pseudo code for resource allocation to the set of ARBs. The number of resources allocated to an ARB is calculated as

$$R(\mathbf{b}) = R_k \times \left(\vartheta^{'}(\mathbf{b})^2\right) \tag{4.12}$$

where $\vartheta^{'}$ is the normalised stimulation level and R_k some constant. Since the stimulation level of the ARBs in $\mathcal{B}$ are normalised, some of the ARBs will have no resources allocated. Thus, after the resource allocation, the weakest ARBs (zero resources) are removed from the population of

ARBs. Each of the remaining ARBs in the population, $\mathcal{B}$, is then cloned and mutated if the calculated ϑ of the ARB is above a certain threshold. These mutated clones are then integrated into the population by re-calculating the network links between the ARBs in $\mathcal{B}$.

The *stopping condition* can be based on whether the maximum size of $\mathcal{B}$ has been reached. Since ARBs compete for resources (based on their stimulation level), an upper limit is set to the number of resources (B-Cells) available in the model. The specified upper limit of resources has an influence on the performance of AINE. If the number of available resources is too large, the network will become too large and difficult to interpret (as in the case of the basic network based AIS [165]). A too small value will result in small networks, which are a premature representation of the antigen patterns. The Self Stabilising AIS model was proposed by Neal [140, 142] to address the drawback of setting the upper limit of available resources in AINE. The population of ARBs is also overtaken by a few ARBs with high stimulation levels that match a small number of antigen, resulting in the premature convergence of the population of ARBs [115]. The fuzzy AINE proposed by Nasraoui *et al.* [139, 137] improves AINE on this drawback.

Self Stabilising AIS: AINE was improved and simplified by a model proposed by Neal, namely the self stabilising AIS [140]. The main difference between these two models is that the SSAIS has no shared/distributed pool with a fixed number of resources that ARBs must compete for. The resource level of an ARB is increased if the ARB has the highest stimulation for an incoming pattern. Each ARB calculates its resource level locally. After a data pattern has been presented to all of the ARBs, the resource level of the most stimulated ARB is increased by addition of the ARB's stimulation level. Algorithm 4.5 lists the pseudo code for the self stabilising AIS. The differences between algorithm 4.3 (AINE) and algorithm 4.5 (SSAIS) are discussed next.

SSAIS defines the stimulation level, ϑ, of an ARB as [140]

$$\vartheta(\mathbf{b},\mathbf{a}) = ps(\mathbf{b},\mathbf{a}) + sns(\mathbf{b}) \tag{4.13}$$

$$ps(\mathbf{b},\mathbf{a}) = 1 - \sigma(\mathbf{b},\mathbf{a}) \tag{4.14}$$

$$sns(\mathbf{b}) = \frac{ns(\mathbf{b})}{|\lambda_{\mathbf{b}}|} \tag{4.15}$$

where ns is defined in equation (4.10) and $\sigma(\mathbf{b},\mathbf{a})$ is the Euclidean distance between an ARB, $\mathbf{b}$, and a training pattern, $\mathbf{a}$, in normalised data space, i.e. $0 \leq \sigma(\mathbf{b},\mathbf{a}) \leq 1$. The *sns* term is based on the average of the summation of the distances to the $|\lambda_{\mathbf{b}}|$ linked neighbours of an ARB, $\mathbf{b}$.

Algorithm 4.5: Self Stabilising AIS

```
Normalise the training data;
Initialise the ARB population, $\mathcal{B}$, using a cross section of the normalised training data;
Initialise the antigen set, $\mathcal{A}$, with the remaining normalised training data;
for each antigen, $\mathbf{a} \in \mathcal{A}$ do
    Present $\mathbf{a}$ to each $\mathbf{b} \in \mathcal{B}$;
    Calculate stimulation level, $\vartheta$, for each ARB, $\mathbf{b}$;
    Select the ARB with the highest calculated stimulation level, $\vartheta$, as $\mathbf{h}$;
    Increase the resource level of $\mathbf{h}$;
    for each ARB, $\mathbf{b} \in \mathcal{B}, \mathbf{b} \neq \mathbf{h}$ do
        Deplete resources of $\mathbf{b}$;
    end
    Remove ARBs with the number of allocated resources less than the mortality
    threshold, $R_\Lambda$;
    Generate $\eta$ clones of $\mathbf{h}$ and mutate;
    Integrate clones (mutated or not) into $\mathcal{B}$;
end
```

The *nn* term defined in equation (4.11) is discarded to prevent premature convergence of ARBs to dominating training patterns.

For each training pattern, $\mathbf{a} \in \mathcal{A}$, presented to the network of ARBs, $\mathcal{B}$, the resource level of each ARB, $\mathbf{b}$, that does not have the highest calculated ϑ is geometrically decayed by the following function [140]

$$R(\mathbf{b}, \mathbf{a}_i) = R_\gamma \times R(\mathbf{b}, \mathbf{a}_{i-1}) \tag{4.16}$$

where $R(\mathbf{b}, \mathbf{a}_i)$ is the number of resources for an ARB, $\mathbf{b}$, after being presented to i training patterns. R_γ is the decaying rate of resources for an ARB. All ARBs with a resource level less than the fixed predefined mortality threshold, R_Λ, are culled from the network. Resources are only allocated by the ARB, $\mathbf{h}$, with the highest calculated stimulation level, ϑ. The number of resources allocated to ARB $\mathbf{h}$ with the highest ϑ is calculated as [140]

$$R(\mathbf{h}, \mathbf{a}_i) = R_\gamma \times (R(\mathbf{h}, \mathbf{a}_{i-1}) + \vartheta(\mathbf{h}, \mathbf{a}_i)) \tag{4.17}$$

where $\vartheta(\mathbf{h}, \mathbf{a}_i)$ is the stimulation level of ARB $\mathbf{h}$ after being presented to i training patterns.

The highest stimulated ARB, **h**, generates η clones. The number of clones generated is given as [140]

$$\eta = \frac{R(\mathbf{h}, \mathbf{a}_i)}{R_\Lambda \times 10}$$

where R_Λ is the mortality threshold. Thus, the number of clones generated by an ARB is proportional to the resource level of the ARB. The generated clones are mutated with a fixed mutation rate. If a clone is mutated, the clone is assigned $R_\Lambda \times 10$ resources from the ARB's resource level. Clones (mutated or not) are integrated with the network of ARBs, $\mathcal{B}$.

The SSAIS resulted in a model that can adapt to continuously changing data sets and a genuinely stable AIS. A drawback to SSAIS is that the final networks that are formed have poor data compression and the SSAIS model has a time lag to adapt to the introduction of a new region of data due to the lack of diversity of the network of ARBs. The stable memory artificial immune network (SMAIN) was proposed by Neal as a simplification of the SSAIS model [142] and is explained next.

Stable Memory Artificial Immune Network (SMAIN): The main difference between SSAIS and SMAIN is the elimination of the mutation operator [142]. The ARB population, $\mathcal{B}$, is initialised with a cross section, $\mathcal{B}_{init}$, of the training data. Each ARB is initialised with R_{init} resources. Furthermore, cloning in SMAIN is only performed on the ARB, **h**, with the closest distance to an antigen pattern, **a**, if the measured distance is greater than the NAT threshold. Whenever an antigen triggers the cloning of an ARB, the antigen is initialised as a cloned ARB. Half of the parent ARB's resources is then assigned to the cloned ARB and the clone is integrated with the ARB population. There is no mutation operator on the clone. Algorithm 4.6 lists the pseudo code for SMAIN. The differences between algorithm 4.5 (SSAIS) and algorithm 4.6 (SMAIN) are further discussed. The stimulation level, ϑ, in equation (4.13) is redefined in SMAIN as [142]

$$\vartheta(\mathbf{b}, \mathbf{a}) = ps(\mathbf{b}, \mathbf{a}) + sns(\mathbf{b}) \tag{4.18}$$

where

$$ps(\mathbf{b}, \mathbf{a}) = \frac{1}{1 + \sigma(\mathbf{b}, \mathbf{a})} \tag{4.19}$$

and

$$sns(\mathbf{b}) = \sum_{j=1}^{|\lambda_{\mathbf{b}}|} \lambda_j \tag{4.20}$$

Algorithm 4.6: Stable Memory Artificial Immune Network

```
Initialise the ARB population, $\mathcal{B}$, using a cross section of the training data;
Initialise the antigen set, $\mathcal{A}$, with the remaining training data;
for each antigen, $\mathbf{a} \in \mathcal{A}$ do
    Present $\mathbf{a}$ to each $\mathbf{b} \in \mathcal{B}$;
    Calculate stimulation level, $\vartheta$, for each ARB, $\mathbf{b} \in \mathcal{B}$;
    Increase the resource level of each $\mathbf{b} \in \mathcal{B}$;
    Select the ARB with the lowest measured distance as $\mathbf{h}$;
    if $\sigma(\mathbf{h}, \mathbf{a}) \geq NAT$ then
        Initialise $\mathbf{a}$ as a clone of ALC $\mathbf{h}$;
        Add clone to $\mathcal{B}$;
    end
    for each ARB, $\mathbf{b} \in \mathcal{B}$ do
        Deplete resources of $\mathbf{b}$;
        Remove $\mathbf{b}$ from $\mathcal{B}$ if the number of allocated resources are less than the mortality
        threshold, $R_\Lambda$;
    end
end
```

where *sns* is still based on the neighbours of an ARB but simplified by removing the need to normalise the distances to the neighbours as in equation (4.15). The resource level of an ARB is calculated as [142]

$$R(\mathbf{b}, \mathbf{a}_i) = R(\mathbf{b}, \mathbf{a}_{i-1}) + \left(R_k \times \left(R_{max} - R_{decay}\right)\right) \times \vartheta(\mathbf{b}, \mathbf{a}_i) \tag{4.21}$$

where $R_k \in (0,1)$ is a constant, R_{max} is the maximum number of resources an ARB can allocate and R_{decay} is the decayed resource level of an ARB as defined in equation (4.16). Similar to SSAIS, all ARBs with a resource level less than the mortality threshold, R_Λ, are culled from the network. SMAIN generates stable memory networks which represents structures inherent in complex data sets.

Fuzzy Artificial Immune Network (Fuzzy AINE): Another enhancement to the AINE model is the fuzzy AINE proposed by Nasraoui *et al.* [136, 139]. The fuzzy AINE was applied to the clustering (profiling) of session patterns for a specific web site. ARBs in the fuzzy AINE are referred to as fuzzy ARBs, since each training pattern is grouped with all fuzzy ARBs to a certain degree of membership (similar to the Fuzzy C-means algorithm which was explained in section 2.3.2). Compared to an ARB in AINE, a fuzzy ARB represents a single pattern as center to a set of member antigen patterns which is within the fuzzy ARB's influence radius. The membership function (or degree of membership) between fuzzy ARB $\mathbf{b}_i$ and antigen pattern $\mathbf{a}_j$ is calculated as [136, 139]

$$m_{ij} = \exp\left(-\frac{\sigma\left(\mathbf{b}_i, \mathbf{a}_j\right)^2}{2\phi_i^2}\right) \tag{4.22}$$

where σ is the distance between $\mathbf{b}_i$ and $\mathbf{a}_j$. The radius of influence, ϕ, is similar to the NAT threshold in [168], but local to each fuzzy ARB. Thus, each fuzzy ARB has a different radius of influence. The membership function decreases with an increase in distance between the fuzzy ARB and the antigen pattern. This results in the gradual exclusion of distant antigen patterns from the fuzzy ARB, resulting in a robust weight function which decreases the influence of outliers. The radius of influence, ϕ_i^2, of each fuzzy ARB $\mathbf{b}_i$ is updated after each iteration using [139]

$$\phi_{i,J}^2 = \frac{\sum_{j=1}^{J} m_{ij}\sigma\left(\mathbf{b}_i, \mathbf{a}_j\right)^2 - \beta(t)\sum_{k=1}^{|\mathcal{B}|} m_{ik}\sigma\left(\mathbf{b}_i, \mathbf{b}_k\right)^2}{\sum_{j=1}^{J} m_{ij} - \beta(t)\sum_{k=1}^{|\mathcal{B}|} m_{ik}} \tag{4.23}$$

where $J = |\mathcal{A}|$. The second term in both the numerator and denominator denote the suppression of similar fuzzy ARBs with overlapping radii of influence. Therefore the stimulation level of a fuzzy ARB consists of the density of the antigen patterns surrounding the fuzzy ARB (as antigen stimulation) and the density of neighbouring fuzzy ARBs (as penalty for suppression). The stimulation level, ϑ, of $\mathbf{b}_i$ is calculated as [139]

$$\vartheta(\mathbf{b}_i) = s_i(\mathcal{A}, |\mathcal{A}|) - \beta(t)\, s_i(\mathcal{B}, |\mathcal{B}|) \tag{4.24}$$

where

$$s_i(X, J) = \frac{\sum_{j=1}^{J} m_{ij}}{\phi_{i,J}^2} \tag{4.25}$$

$s_i(X, |X|)$ calculates the density of patterns within set X surrounding $\mathbf{b}_i$; s_i calculates the antigen stimulation for $X = \mathcal{A}$ and the suppression for $X = \mathcal{B}$. Algorithm 4.7 provides the pseudo code for fuzzy AINE.

Algorithm 4.7: Fuzzy Artificial Immune Network (Fuzzy AINE)

```
Normalise the training data;
Initialise the fuzzy ARB population, $\mathcal{B}$, using a randomly selected cross section of the
normalised training data;
Initialise $\phi^2$ for each fuzzy ARB;
Initialise the antigen set, $\mathcal{A}$, with the remaining normalised training data;
Set the maximum number of available resources, $R_{max}$;
while not stopping condition do
    for each antigen, $\mathbf{a}_j \in \mathcal{A}$, at index position $j$ in $\mathcal{A}$ do
        for each ARB, $\mathbf{b}_i \in \mathcal{B}$, at index position $i$ in $\mathcal{B}$ do
            Update membership function $m_{ij}$ of fuzzy ARB $\mathbf{b}_i$;
        end
    end
    for each ARB, $\mathbf{b}_i \in \mathcal{B}$, at index position $i$ in $\mathcal{B}$ do
        Calculate the simulation level $\vartheta(\mathbf{b}_i)$;
        Update the radius influence $\phi_i^2$ of fuzzy ARB $\mathbf{b}_i$;
    end
    Allocate resources (see algorithm 4.4) to the set of fuzzy ARBs, $\mathcal{B}$;
    Clone and mutate remaining fuzzy ARBs in $\mathcal{B}$;
    Integrate mutated clones into $\mathcal{B}$;
end
```

To avoid the premature convergence of the population of fuzzy ARBs, the number of resources allocated to a fuzzy ARB is calculated as [136, 137]

$$R(\mathbf{b}_i) = R_k \times \left(\log\left[\vartheta'(\mathbf{b}_i)\right]\right) \tag{4.26}$$

where ϑ' is the normalised stimulation level of a fuzzy ARB and R_k some constant. This modification to the number of resources allocated to a fuzzy ARB will limit the influence of those fuzzy ARBs with high stimulation to slowly overtake the population.

A cloned fuzzy ARB also inherits the radius of influence ϕ^2 value of the parent fuzzy ARB. After the integration of the mutated clones, the fuzzy ARBs with the same B-Cell representation are merged into one fuzzy ARB. The merging of identical fuzzy ARBs limit the high rate of population growth. The merging of two fuzzy ARBs, $\mathbf{b}_i$ and $\mathbf{b}_j$, is done through a crossover operator on the fuzzy ARBs' attributes, defined as [137, 139]

$$\mathbf{b}_{k,n} = avg\left(\mathbf{b}_{i,n}, \mathbf{b}_{j,n}\right)$$

where $\mathbf{b}_{k,n}$ is the value for attribute n of merged ARB, $\mathbf{b}_k$, and avg is the average value between the n-th attributes in $\mathbf{b}_i$ and $\mathbf{b}_j$ respectively.

In the context of data clustering, fuzzy AINE maintains a diverse set of fuzzy ARBs to represent the different clusters within a data set, compared to a few good ARBs in AINE which dominates the population of ARBs, and therefore the population prematurely converges. The fuzzy AINE proved to be scalable and diverse in profiling web usage session patterns. The Euclidean distance in AINE (affinity measurement) was replaced by the calculation of the cosine similarity between two session patterns in fuzzy AINE [137, 139]. The cosine similarity between two vectors is defined in equation (2.8).

A drawback to fuzzy AINE in the context of web usage profiling is the assumption that all web usage sessions are available beforehand. This is a disadvantage in environments with limited resources (like system memory), making the fuzzy AINE model less scalable. Another drawback, which is common to most clustering algorithms, is that any change in web usage sessions results in the re-application of the model to cluster the data. Nasraoui *et al.* highlighted these drawbacks in [134, 138] and improved the fuzzy AINE with the Dynamic Weighted B-Cell AIS model which is able to cluster streaming non-stationary web usage sessions [134].

Dynamic Weighted B-Cell AIS: Nasraoui *et al.* proposed a scalable AIS model which can be applied to the clustering of non-stationary data [134, 135, 138]. The model is similar to fuzzy AINE in that each training pattern is grouped with all ARBs to a certain degree of membership. An ARB in this model is known as a dynamic weighted B-Cell (DWB-cell). The membership function of fuzzy AINE (as defined in equation (4.22)) is adapted in [135] to be more applicable for non-stationary environments. The adapted membership function includes the time when an antigen pattern was presented to the network of DWBs. The membership function for DWB, $\mathbf{b}_i$, is calculated as [135]

$$m_{ij} = \exp\left[-\left(\frac{\sigma\left(\mathbf{b}_i, \mathbf{a}_j\right)^2}{2\phi_i^2} + \frac{j}{\tau}\right)\right] \tag{4.27}$$

where σ is the distance between $\mathbf{b}_i$ and the j-th antigen pattern, $\mathbf{a}_j$. Thus the antigen index j increases monotonically with time. τ controls the rate at which previously presented antigens contribute to the degree of membership as well as the relevance of the network. The above membership function not only decreases with an increase in distance between a DWB-cell and an antigen pattern as in fuzzy AINE, but also with the time since an antigen pattern has been presented to the network of DWBs. Therefore the most recent antigen patterns presented to a DWB-cell have a higher degree of membership compared to less current antigen patterns.

The DWB population, $\mathcal{B}$, has a maximum of $\mathcal{B}_{max}$ DWB-cells and is initialised with the first $\mathcal{B}_{max}$ of incoming antigen training patterns. The radius of influence, ϕ^2, of each DWB-cell is initialised with ϕ_{init}. The antigen stimulation level of a DWB-cell after J antigen patterns is given as [135]

$$s_{i,J} = s_i(\mathcal{A}, J) \tag{4.28}$$

where $s_i(\mathcal{A}, J)$ is defined in equation (4.25) and the radius of influence, ϕ^2, is given as

$$\phi^2_{i,J} = \frac{\sum_{j=1}^{J} m_{ij}\sigma\left(\mathbf{b}_i, \mathbf{a}_j\right)^2}{2\sum_{j=1}^{J} m_{ij}} \tag{4.29}$$

In order to calculate a DWB-cell's stimulation level and radius of influence after each antigen pattern has been presented to the DWB-cell, the following derivatives from the above equations are given in [135] as approximate incremental updates for the stimulation level and radius of influence of $\mathbf{b}_i$ respectively:

$$s_{i,J} = \frac{\exp\left(-\frac{1}{\tau}\right) M_{i,J-1} + m_{iJ}}{\phi^2_{i,J}} \tag{4.30}$$

where

$$\phi^2_{i,J} = \frac{\exp\left(-\frac{1}{\tau}\right) \phi^2_{i,J-1} M_{i,J-1} + m_{iJ}\sigma\left(\mathbf{b}_i, \mathbf{a}_J\right)^2}{2\left[\exp\left(-\frac{1}{\tau}\right) M_{i,J-1} + m_{iJ}\right]} \tag{4.31}$$

and

$$M_{i,J-1} = \sum_{j=1}^{J-1} m_{ij} \tag{4.32}$$

Algorithm 4.8 lists the pseudo code for the dynamic weighted B-Cell model. The model also proposed the incorporation of a dynamic stimulation/suppression factor into the stimulation level of a DWB-cell to control the proliferation and redundancy of DWB-cells in the network. Thus, old sub-nets die if not re-stimulated by current incoming antigen patterns. The total stimulation

Algorithm 4.8: Dynamic Weighted B-Cell

```
Set the maximum size of the DWB population as $\mathcal{B}_{max}$;
Initialise the DWB population, $\mathcal{B}$, using the first $\mathcal{B}_{max}$ of incoming antigen training patterns in set $\mathcal{A}$;
Initialise $\phi^2 = \phi_{init}$ for each DWB;
Compress the population of DWBs into $k_{compress}$ sub-nets using K-means clustering;
for each antigen, $\mathbf{a}_j \in \mathcal{A}$, at index position $j$ in $\mathcal{A}$ do
    Present $\mathbf{a}_j$ to the centroid $\mathbf{c}_k$ of each sub-net $C_k$ and calculate the weight $m_{kj}$ and update $\phi_k^2$ using equations (4.27) and (4.31) respectively;
    Select the sub-net with the maximum $m_{kj}$ as the most activated subnet $C_w$;
    if $\forall \mathbf{b}_i \in C_w$ the weight $m_{ij} < m_{min}$ then
        Create new DWB-cell $\mathbf{x} = \mathbf{a}_j$ and set the new cell's $\phi^2 = \phi_{init}$;
        Add new DWB-cell to the population;
    end
    else
        for each DWB, $\mathbf{b}_i \in C_w$ do
            if $m_{ij} \geq m_{min}$ then
                Reset $age_i = 0$ of DWB $\mathbf{b}_i$;
            end
            else
                Increment $age_i = age_i + 1$ of DWB $\mathbf{b}_i$;
            end
            Calculate the stimulation level of $\mathbf{b}_i$ using equation (4.34);
            Update the radius of influence, $\phi_i^2$, using equation (4.35);
        end
    end
    Clone and mutate DWB-cells;
    if $|\mathcal{B}| > \mathcal{B}_{max}$ then
        Sort population of DWBs in ascending order of their stimulation levels;
        Remove top $|\mathcal{B}| - \mathcal{B}_{max}$ DWB-cells from the sorted population;
    end
    Compress the population of DWBs every $A$ antigens into $k_{compress}$ sub-nets using K-means clustering with the previous centroids as initial centroids;
end
```

of a DWB-cell consists of the antigen stimulation as well as the co-stimulation and suppression from other DWB-cells in the network. The total stimulation of $\mathbf{b}_i$ is given as [135]

$$\vartheta(\mathbf{b}_i) = s_{i,J} + \alpha(age_i)\frac{\sum_{n=1}^{|\mathcal{B}|} m_{in}}{\phi_{i,J}^2} - \beta(age_i)\frac{\sum_{n=1}^{|\mathcal{B}|} m_{in}}{\phi_{i,J}^2} \tag{4.33}$$

where $\alpha(age_i) = \left(1 + \frac{age_i}{\tau_\alpha}\right)^{-1}$ is the co-stimulation coefficient, $\beta(age_i) = \left(1 + \frac{age_i}{\tau_\beta}\right)^{-1}$ is the network suppression coefficient and age_i records the age of $\mathbf{b}_i$.

Another drawback of existing immune network based learning models is that the number of interactions between the B-Cells in the network and a specific antigen are immense. The model of [135] clusters the DWB-cells into $k_{compress}$ sub-nets using K-means clustering (as discussed in section 2.3.2) to decrease the number of interactions between an antigen pattern and the DWB-cells in the network. The centroids of each of these formed clusters (or sub-nets) are used to represent the sub-nets and interact with the presented antigen pattern. Therefore the network of DWB-cells is compressed in different sub-nets and the total stimulation level and radius of influence for each DWB-cell in a specific sub-net C_k is calculated as [135]

$$\vartheta(\mathbf{b}_i) = s_{i,J} + \alpha(age_i)\frac{\sum\limits_{\forall \mathbf{b}_n \in C_k} m_{in}}{\phi_{i,J}^2} - \beta(age_i)\frac{\sum\limits_{\forall \mathbf{b}_n \in C_k} m_{in}}{\phi_{i,J}^2} \tag{4.34}$$

where $\mathbf{b}_i \in C_k$ and

$$\phi_{i,J}^2 = \frac{D_{i,J}^2 + \alpha(age_i)\sum\limits_{\forall \mathbf{b}_n \in C_k} m_{in}\sigma(\mathbf{b}_i, \mathbf{b}_n)^2 - \beta(age_i)\sum\limits_{\forall \mathbf{b}_n \in C_k} m_{in}\sigma(\mathbf{b}_i, \mathbf{b}_n)^2}{2\left[M_{i,J} + \alpha(age_i)\sum\limits_{\forall \mathbf{b}_n \in C_k} m_{in} - \beta(age_i)\sum\limits_{\forall \mathbf{b}_n \in C_k} m_{in}\right]} \tag{4.35}$$

where

$$D_{i,J}^2 = \exp\left(-\frac{1}{\tau}\right)\phi_{i,J-1}^2 M_{i,J-1} + m_{iJ}\sigma(\mathbf{b}_i, \mathbf{a}_J)^2 \tag{4.36}$$

$$M_{i,J} = \exp\left(-\frac{1}{\tau}\right)M_{i,J-1} + m_{iJ} \tag{4.37}$$

A DWB-cell is only cloned if it reached maturity and is activated by an antigen pattern. A

DWB-cell is activated when $m_{ij} > m_{min}$, where m_{min} is the minimum degree to become activated. Maturity of a DWB-cell is reached when the cell's age age_i is within a specified range, $a_{min} \leq age_i \leq a_{max}$. Cloning of a DWB-cell is proportional to the cell's stimulation level. If a DWB-cell's age exceeds the maximum time threshold ($age_i > a_{max}$), cloning of the cell is prevented, thus increasing the probability to clone newer DWB-cells. The number of clones generated for an activated and mature DWB-cell, $\mathbf{b}_i$, is given as [135]

$$\eta(\mathbf{b}_i) = k_{clone} \times \frac{\vartheta(\mathbf{b}_i)}{\sum_{n=1}^{|\mathcal{B}|} \vartheta(\mathbf{b}_n)} \qquad \text{if} \quad a_{min} \leq age_i \leq a_{max} \tag{4.38}$$

Whenever the maximum size, $\mathcal{B}_{max}$, of the network of DWB-cells has been reached, the DWB-cells are sorted in ascending order of their stimulation levels and starting from the top, the DWB-cells with the lowest stimulation levels are removed until the size of the network is equal to the maximum size, $\mathcal{B}_{max}$.

The mechanism of somatic hyper mutation is computationally expensive and replaced in the DWB-model by a different concept, namely *dendritic injection*. When the immune network encounters an antigen that the network cannot react to, the specific antigen is initialised as a DWB-cell. Thus, new information is *injected* into the immune network, i.e. *dendritic injection*. The dendritic cell system was discussed in section 3.8. The DWB-model has proven to be robust to noise, adaptive, and scalable in learning antigen structures in a non-stationary environment.

aiNet: De Castro and Von Zuben proposed a novel network based AIS model which evolves a population of linked memory ALCs through clonal selection [32, 36]. The model is applied to the problem of data clustering. A typical ALC network consists of ALC nodes (the B-Cells or antibodies) which are connected by edges to form node pairs. A weight value (connection strength) is assigned to each edge to indicate the similarity between two nodes. Thus, the ALC network that is formed during training is presented by an edge-weighted graph. Edges in the ALC network are pruned by measuring the weight of each edge against a *similarity* threshold. Pruning the ALC network results in the formation of a number of sub-networks. Each of the formed sub-networks represents a cluster within the data set. Thus, the evolved population of linked memory ALCs contains a number of ALC networks, each representing a cluster in the data set.

The data patterns in the training set, $\mathcal{A}$, are seen as antigens. Training of the memory ALC population, $\mathcal{B}$, follows an iterative two phase approach. The first phase implements the process of clonal selection as proposed by the CLONALG model of De Castro and Von Zuben [35]. The second phase is the *network formation* with *network suppression* between the evolved memory ALCs. During training, each pattern is presented to the population of memory ALCs. The affinity between an antigen pattern and an ALC is inversely proportional to the measured Euclidean distance between the antigen pattern and the ALC. Thus, a higher measured Euclidean distance results in a lower affinity measurement and vice-versa. Euclidean distance was defined in equation (2.3). The affinity between an ALC, **b**, and an antigen pattern, **a**, is calculated as [36]

$$f(\mathbf{b},\mathbf{a}) \quad = \quad \frac{1}{\sigma(\mathbf{b},\mathbf{a})} \tag{4.39}$$

where σ is the Euclidean distance. After the affinity to each ALC is calculated, a subset of the n highest affinity ALCs is selected for cloning. The number of clones to generate for an ALC is proportional to the ALC's affinity to the antigen pattern, **a**. Therefore, a higher affinity results in more clones. The number of clones for the n-th selected ALC is defined as [36]

$$\eta(\mathbf{b}_n) \quad = \quad round(|\mathcal{B}| - \sigma(\mathbf{b}_n,\mathbf{a}) \times |\mathcal{B}|) \tag{4.40}$$

where $|\mathcal{B}|$ is the cardinality of the ALC set $\mathcal{B}$ and σ is the Euclidean distance between antigen **a** and the n-th selected ALC, $\mathbf{b}_n$. Each of the generated clones are then mutated. An ALC clone, $\mathbf{b}^*$, is mutated as [32, 36]

$$\mathbf{b}^{'} \quad = \quad \mathbf{b}^* - \varsigma(\mathbf{b}^* - \mathbf{a}) \tag{4.41}$$

where ς is the mutation rate on ALC clone $\mathbf{b}^*$. The mutation rate of a clone is inversely proportional to the affinity of the clone's parent ALC. Thus, a higher affinity level results in a smaller mutation rate. The affinity between the antigen pattern and each of the mutated clones is then calculated.

A clonal memory set is then selected from the mutated clones. The clonal memory set contains ζ% of the mutated clones with the highest affinity. The Euclidean distance between the antigen pattern and each memory clone in the clonal memory set is measured against a death threshold, ε_{death}. Memory clones with a measured Euclidean distance above ε_{death} are removed from the clonal memory set. The Euclidean distance between each of the remaining memory clones is then calculated as the network distance. Clones with a network distance below the network suppression threshold, $\varepsilon_{network}$, are also removed from the clonal memory set, i.e. clonal suppression. The remaining clonal memory set is then concatenated with the set of memory ALC networks, $\mathcal{B}$.

After all the antigen patterns have been presented to the memory set of ALC networks, the Euclidean distance between ALCs in the memory set is measured against $\varepsilon_{network}$. ALCs with a measured Euclidean distance below the $\varepsilon_{network}$ threshold are removed from the memory set, i.e. network suppression. A percentage (φ%) of the lowest affinity (dissimilar) ALCs in $\mathcal{B}$ is replaced with randomly generated ALCs. The remaining memory set is then used in the next iteration. Algorithm 4.9 provides the pseudo code of the aiNet model.

The *stopping condition* of the *while*-loop can be one of the following [32, 36]:

1. **Setting a *loop* counter**: A counter can be set to determine the number of loops.
2. **Setting the maximum size of the network**: The while-loop can be stopped when the size of the network reaches a maximum.
3. **Testing for convergence**: The loop terminates when the average error between the training patterns in $\mathcal{A}$ and ALCs in $\mathcal{B}$ rises after a number of consecutive loops.

The final network of memory ALCs, $\mathcal{B}$, is partitioned with an agglomerative hierarchical clustering technique with single linkage (as discussed in section 2.3.1) to determine [36]

- the number of clusters presented by the network of memory ALCs $\mathcal{B}$,
- the spatial distribution of these clusters, and
- which ALCs belong to the same cluster.

De Castro and Von Zuben also proposed the minimal spanning tree as an alternative technique to determine the above goals [36]. Since the network of memory ALCs can be presented as a graph with weighted edges, a minimal spanning tree is generated as a sub-graph of the network of memory ALCs, such that the summed weights of the tree is minimised.

As discussed in chapter 2, each cluster can be represented by a centroid. De Castro and Von Zuben proposed the application of a fuzzy membership function to determine the degree of membership between the centroids and each of the memory ALCs (similar to the fuzzy C-means clustering algorithm as explained in section 2.3.2).

Algorithm 4.9: aiNet Learning Algorithm

```
Determine the antigen patterns as training set $\mathcal{A}$ and initialise the set of memory ALCs, $\mathcal{B}$;
while stopping condition not true do
    for each antigen pattern, $\mathbf{a}_j \in \mathcal{A}$ do
        for each ALC, $\mathbf{b}_i \in \mathcal{B}$ do
            Calculate the antigen affinity $f\left(\mathbf{b}_i, \mathbf{a}_j\right)$;
        end
        Select $n$ of the highest affinity ALCs as set $\mathcal{H}$;
        for each $\mathbf{b}_n \in \mathcal{H}$ do
            Create $\eta\left(\mathbf{b}_n\right)$ mutated clones of $\mathbf{b}_n$ and add to set $\mathcal{H}'$;
        end
        for each $\mathbf{b}'_n \in \mathcal{H}'$ do
            Calculate the antigen affinity, $f\left(\mathbf{b}'_n, \mathbf{a}_j\right)$;
        end
        Select $\zeta\%$ of the highest affinity antibodies as set $\mathcal{M}$;
        for each $\mathbf{y}_m \in \mathcal{M}$ do
            if $f_a\left(\mathbf{a}_j, \mathbf{y}_m\right) > \varepsilon_{death}$ then
                Remove $\mathbf{y}_m$ from $\mathcal{M}$;
            end
        end
        for each $\mathbf{y}_{m_1} \in \mathcal{M}$ do
            for each $\mathbf{y}_{m_2} \in \mathcal{M}$ do
                if the network affinity $f_a\left(\mathbf{y}_{m_1}, \mathbf{y}_{m_2}\right) < \varepsilon_{network}$ then
                    Remove $\mathbf{y}_{m_1}$ and $\mathbf{y}_{m_2}$ from $\mathcal{M}$;
                end
            end
        end
        $\mathcal{B} = \mathcal{B} \cup \mathcal{M}$;
    end
    for each $\mathbf{b}_{i_1} \in \mathcal{B}$ do
        for each $\mathbf{b}_{i_2} \in \mathcal{B}$ do
            if the network affinity $f_a\left(\mathbf{b}_{i_1}, \mathbf{b}_{i_2}\right) < \varepsilon_{network}$ then
                Remove $\mathbf{b}_{i_1}$ and $\mathbf{b}_{i_2}$ from $\mathcal{B}$;
            end
        end
    end
    Replace $\varphi\%$ of the lowest affinity ALCs in $\mathcal{B}$ with randomly generated ALCs;
end
```

Drawbacks of the aiNet model include the number of parameters that need to be specified and that the cost of computation increases as the dimension of the training patterns increases. The minimum spanning tree will also have difficulty in determining the network clusters if there are intersections between the clusters in the training data set. The aiNet is capable of reducing data redundancy and obtaining a compressed representation of the data.

There exists many different document clustering techniques, but these techniques have the main drawback that they directly apply the clustering techniques to the raw data (collection of documents). Larger collections contain more noise in the data which result in the formation of clusters with inferior quality. Tang and Vemuri [164] used the aiNet as a data preprocessing algorithm since aiNet is capable of reducing data redundancy and obtaining a compressed representation of the data. Each document is treated as an antigen in aiNet.

Principle component analysis (PCA) is also introduced in the proposed framework of Tang and Vemuri [164] to reduce the dimension of the vectors in the data after which the aiNet algorithm is applied to the compressed vectors. PCA resulted in a speedup of the compression of the data and further reduced the noise in the data. The result of the aiNet (compressed representation of the data) is then either clustered with K-means clustering or Hierarchical Agglomerative Clustering (HAC). Experimental results in [164] have shown that aiNet as a data preprocessing and compression algorithm obtained better clustering results (more compact clusters) than directly clustering the raw data with K-means clustering or HAC.

Another model which is based on the multipopulation aspect of aiNet is the proposed multi-objective multipopulation artificial immune network (MOM-aiNet) model by Coelho *et al.* [28]. Further investigation into MOM-aiNet led to an improved model, MOM-aiNet+ [29]. Contrary to aiNet, MOM-aiNet+ not only keeps the best individual of each subpopulation but several within each subpopulation. MOM-aiNet+ was applied to the biclustering problem which is a multi-objective optimisation problem. The biclustering technique is capable of finding several subsets (biclusters) in a data set in such a way that each subset contains patterns with a certain shared similarity [25, 80]. The quality of each bicluster can be measured by the volume of the bicluster (number of patterns $\times$ number of features) and the degree of similarity among the patterns within the bicluster. Both of these measurements need to be maximised. Furthermore, the number of patterns in a data set which is covered by the different biclusters and the degree of overlap between the biclusters also need to be measured.

Minor refinements to the aiNet model led to versions of the model which were respectively applied to the initialisation of centers of a radial basis function neural network [37] and the optimisation of multi-modal functions [33, 55]. The refined aiNet model is known as opt-aiNet. The dynamic opt-aiNet (dopt-aiNet) was recently proposed as an improvement to opt-aiNet for non-stationary environments [39, 40]. Both of these models are discussed next.

opt-aiNet: The aiNet model was adapted to solve multi-modal function optimisation problems and is known as *opt-aiNet* [33]. A few observations on the originally proposed opt-aiNet model in [33] were summarised in [167]. Some of the features of opt-aiNet listed are among others a dynamic population size, the capability to explore and exploit the search space and the capability to maintain multiple optima solutions [167]. Algorithm 4.10 lists the pseudo code of the opt-aiNet model with minor modifications as proposed in [167] (assuming minimisation of the objective) and is discussed next. The differences to the original model are also highlighted.

Algorithm 4.10: opt-aiNet Learning Algorithm

```
Randomly initialise a population of ALCs, $\mathcal{B}$, with size $\mathcal{B}_{init}$;
while stopping condition not true do
    Determine the fitness, $f$, of each ALC, $\mathbf{b}$, in $\mathcal{B}$;
    Normalise the fitnesses of population $\mathcal{B}$;
    repeat
        Generate $\eta$ ALC clones for each $\mathbf{b}$;
        Mutate each ALC clone proportionally to the normalised fitness of its parent ALC;
        Determine the fitness of each mutated clone, $\mathbf{b}'$, and select the mutated clone with
        the lowest fitness as $\mathbf{b}^*$;
        if $\mathbf{b}^*$ has a lower fitness than $\mathbf{b}$ then
            Replace $\mathbf{b}$ with $\mathbf{b}^*$;
        end
        Determine the fitness, $f$, of each ALC, $\mathbf{b}$, in $\mathcal{B}$;
        Normalise the fitnesses of population $\mathcal{B}$;
    until the difference in average fitness of $\mathcal{B}$ is less than a pre-defined threshold $\varepsilon_{fitness}$;
    Determine the network affinity between each pair of ALCs in $\mathcal{B}$;
    If the calculated network affinity is below the network suppression threshold $\varepsilon_{network}$,
    remove the ALC with the lowest fitness from $\mathcal{B}$;
    Add $\varphi$% (of the size of $\mathcal{B}$) of randomly generated ALCs to $\mathcal{B}$;
end
```

The network affinity between two ALCs is calculated as the Euclidean distance, as defined in equation (2.3). The fitness of an ALC, $\mathbf{b}$, is calculated using the fitness function, f, which is the objective that needs to be optimised. An ALC clone is mutated proportionally to the normalised fitness, f^*, of its parent $\mathbf{b}$ as [33]

$$\mathbf{b}' = \mathbf{b} + \left(\frac{1}{\varsigma}\right) \exp\left[-\left(1 - f^*\left(\mathbf{b}\right)\right)\right] N\left(0,1\right) . \tag{4.42}$$

where $N(0,1)$ is a Gaussian random variable with zero mean and standard deviation of one. ς controls the decay of the inverse exponential function. The above mutation results in that highly fit ALCs are mutated less than less fit ALCs. Therefore poor ALCs are mutated more to explore the search space and good ALCs are mutated less to exploit the search space. The *stopping condition* of the *while*-loop is set to a maximum number of iterations, t_{max}.

The differences between the original opt-aiNet model and algorithm 4.10 are [167]:

- the assumption in the original model that the average fitness always decreases (assuming minimisation of the objective). The degree of similarity between the current average fitness and the previous average fitness is measured against a threshold value. A disadvantage of this assumption is that the degree of similarity will always be less than the threshold if the previous average fitness is greater than the current average fitness. This holds true even if there is a large difference between the previous and current average fitness. This drawback is addressed in algorithm 4.10 by calculating the difference between the previous and current average fitness and measuring the result against a similarity threshold value.

- the suppression of ALCs in the original model always results in the first ALC to be removed whenever the calculated network affinity (Euclidean distance) between two ALCs are below the network suppression threshold. A drawback of this approach is that the removal of an ALC is not based on its fitness, which can result in the removal of a potential optimum solution. Therefore the fitness of ALCs in algorithm 4.10 are first evaluated and the ALC with the worst fitness is removed.

In the context of data clustering as an optimisation problem, each ALC in the population represents a possible partitioning of the data set (similar to Clustering PSO as discussed in section 2.7.1). Thus, an ALC represents K centroids, one for each cluster. An ALC is defined as $\mathbf{b}_i = (\mathbf{c}_{i,1}, \mathbf{c}_{i,2}, \ldots, \mathbf{c}_{i,K})$ where $\mathbf{c}_{i,k}$ is the cluster centroid of the k-th cluster, $C_{i,k}$, represented by the *i*-th ALC in the population. The objective function that needs to be optimised is the quantization error as defined in equation (2.75) and is thus the fitness function of the ALCs, i.e. $f = J_{PSO}$.

Dynamic opt-aiNet: The dynamic opt-aiNet (dopt-aiNet) model was proposed by de França *et al.* [39, 40] and improved the opt-aiNet model to be more suitable for non-stationary environments. Two drawbacks of the opt-aiNet model are the large number of function evaluations to find good solutions and the possibility of an excessive increase in the size of the population over time. The recommended modifications in [40] to address these drawbacks and enhance opt-aiNet are discussed next.

In dopt-aiNet the population size is preset to a maximum. Whenever the size of the population reaches this maximum a percentage of the ALCs with the worst fitness are removed from the population. Another recommendation is to keep a separate memory population. The memory population contains ALCs which have not been replaced by their mutated clones for a certain peroid of time. Each ALC therefore needs to be initialised with a rank value. The rank value is incremented each time a mutated ALC clone replaces its parent ALC and decremented if not. An ALC is replaced by its mutated ALC clone if the mutated ALC clone improves the parent ALC's fitness. An ALC is moved to the memory population when the rank value reaches zero. The memory ALC then receives a new rank value which follows the same process. When the rank of a memory ALC reaches zero, it does not undergo any mutation.

The ς parameter for the Gaussian mutation in opt-aiNet sometimes require pre-analysis of the function landscape to be set properly. Small ς-values may lead to slower convergence whereas too large ς-values may lead to a mutated ALC clone which diverges even further from an optimum solution. The *golden section* technique in [13] is recommended to find the optimal value for ς. The golden section technique divides a search space into two and selects the interval with the best fitness. The selected interval is then again divided and the sub-interval with the best fitness is selected for division. The process of division is recursive and applied to the selected interval until the interval reaches a given length. Each interval is divided with the *golden ratio* which is found on many nature structures. The golden section is only guaranteed for continuous, convex and unimodal objective functions. Since the model has no prior knowledge of the objective function, the initial search interval is divided into four segments. The golden section technique is then applied to each of these segments. Algorithm 4.11 lists the pseudo code of the dopt-aiNet model with the recommended modifications and additional mutation operators as proposed in [40] (assuming minimisation of the objective) and is discussed next.

Two new mutation operators are proposed and applied to ALCs with a rank value that is greater than zero in the current ALC population and the memory population. These mutation operators are:

- One-dimensional mutation where each dimension of an ALC is individually mutated. This means that an N-dimensional ALC will generate N mutated clones, each clone mutated in exactly one dimension. Two additional ALC clones are also mutated in the direction of the unitary vectors $\mathbf{1}$ and $-\mathbf{1}$, respectively. Algorithm 4.12 lists the pseudo code for one dimensional mutation. Matrix $\mathbf{D}^{(N+2)\times N}$ contains the identity matrix of size N and two rows with the unitary vectors $\mathbf{1}$ and $-\mathbf{1}$, respectively. ς is calculated with the golden section technique.

- Gene duplication is inspired by the duplication of genes in nature whenever a chromosome is read [86, 144]. A dimension of an ALC is randomly selected and its value is copied into another randomly selected dimension if the fitness of the ALC is improved. Algorithm 4.13 lists the pseudo code for gene duplication mutation.

Another drawback of opt-aiNet is the network suppression threshold, $\varepsilon_{network}$. Since the Euclidean distance between ALCs determine the network affinity, some knowledge of the fitness landscape or pre-analysis should be done to adjust $\varepsilon_{network}$ to an optimal and appropriate value. The cell line supression technique is recommended as an enhancement to the Euclidean distance measure. Algorithm 4.14 lists the pseudo code for cell line suppression between two ALCs. The middle point, $\mathbf{p}_m$, of the line segment from $\mathbf{p}_i$ to $\mathbf{p}_j$ is calculated. The middle point is then projected onto the line segment to calculate the nearest point, $\mathbf{p}_{projection}$, to the line segment. If $\mathbf{p}_{projection}$ is inside the line segment then the network affinity is calculated as $\sigma(\mathbf{p}_m, \mathbf{p}_n)$, which means that the nearest point is at the point $\mathbf{p}_n$ where $\overline{\mathbf{p}_m\mathbf{p}_n} \perp \overline{\mathbf{p}_j\mathbf{p}_i}$. If $\mathbf{p}_{projection}$ is outside the line segment then the network affinity is calculated between $\mathbf{p}_m$ and $\mathbf{p}_i$ (in the case where $\mathbf{d}_m \otimes \mathbf{d}_j \leq 0$) or $\mathbf{p}_j$ (in the case where $\left|\mathbf{d}_j\right| \leq \mathbf{d}_m \otimes \mathbf{d}_j$).

Other Network Based Models: Gaspar and Collard proposed the *Simple Artificial Immune System* (SAIS) which is inspired by the network formation and adaptability of the immune system to foreign antigens [58], specifically the primary and secondary responses to foreign antigens (as discussed in section 3.3.3). SAIS is applied to problems within a binary space and therefore measures the affinity between an ALC and an antigen pattern using the Hamming distance (as

Algorithm 4.11: dopt-aiNet Learning Algorithm

```
Randomly initialise a population of ALCs, $\mathcal{B}$, with size $\mathcal{B}_{max}$;
while stopping condition not true do
    Determine the fitness, $f$, of each ALC, b, in $\mathcal{B}$;
    Normalise the fitnesses of population $\mathcal{B}$;
    repeat
        Generate $\eta$ ALC clones for each b;
        Gaussian mutate each ALC clone proportionally to the normalised fitness of its
        parent ALC;
        Determine the fitness of each mutated clone, $\mathbf{b}'$, and select the mutated clone with
        the lowest fitness as $\mathbf{b}^*$;
        if $\mathbf{b}^*$ has a lower fitness than b then
            Replace b with $\mathbf{b}^*$;
            Increment the rank value of $\mathbf{b}^*$ by one;
        end
        else
            Decrement the rank value of b by one;
        end
        if the rank value of b equals zero then
            Remove b from $\mathcal{B}$ and add to the memory population, $\mathcal{M}$;
        end
        Apply one-dimensional mutation on each ALC in $\mathcal{B}$ (Algorithm 4.12);
        Apply gene duplication on each ALC in $\mathcal{B}$ (Algorithm 4.13);
        Apply one-dimensional mutation on each memory ALC in $\mathcal{M}$ (Algorithm 4.12);
        Apply gene duplication on each memory ALC in $\mathcal{M}$ (Algorithm 4.13);
        for each memory ALC do
            if the memory ALC has improved after mutation then
                Increment the rank value of the memory ALC by one;
            end
            else
                Decrement the rank value of the memory ALC by one;
            end
        end
        Determine the fitness, $f$, of each ALC, b, in $\mathcal{B}$;
        Normalise the fitnesses of population $\mathcal{B}$;
    until the difference in average fitness of $\mathcal{B}$ is less than a pre-defined threshold $\varepsilon_{fitness}$;
    Determine the network affinity between each pair of ALCs in $\mathcal{B}$ and suppress the ALC
    network using the cell line suppression as listed in algorithm 4.14;
    Add $\varphi$% (of $\mathcal{B}_{max}$) of randomly generated ALCs to $\mathcal{B}$;
    if $|\mathcal{B}| > \mathcal{B}_{max}$ then
        Remove $(\mathcal{B}_{max} - |\mathcal{B}|)$ ALCs with the worst fitness from population $\mathcal{B}$;
    end
end
```

Algorithm 4.12: One-dimensional Mutation Algorithm

```
Generate $n+2$ ALC clones for ALC $\mathbf{b}$;
for each ALC clone $\mathbf{b}^*$ do
    for each row $\mathbf{d}$ in matrix $\mathbf{D}$ do
        Generate a mutated clone $\mathbf{b}'$ from $\mathbf{b}^*$ using $\mathbf{b}' = \mathbf{b}^* + \mathbf{d} \times \varsigma$;
    end
end
Determine the fitness of each mutated clone, $\mathbf{b}'$, and select the mutated clone with the
lowest fitness as $\mathbf{b}^*$;
if $\mathbf{b}^*$ has a lower fitness than $\mathbf{b}$ then
    Replace $\mathbf{b}$ with $\mathbf{b}^*$;
end
```

Algorithm 4.13: Gene Duplication Mutation Algorithm

```
Initialise gene to the value of a randomly selected dimension of ALC $\mathbf{b}$;
for each dimension $n$ of ALC $\mathbf{b}$ do
    $oldval = \mathbf{b}_n$;
    $\mathbf{b}_n = gene$;
    if the fitness of ALC $\mathbf{b}$ does not improve then
        $\mathbf{b}_n = oldval$;
    end
end
```

Algorithm 4.14: Cell Line Suppression Algorithm

```
Select two ALCs $\mathbf{b}_i$ and $\mathbf{b}_j$;
$\mathbf{p}_i = [\mathbf{b}_i, f(\mathbf{b}_i)]$;
$\mathbf{p}_j = [\mathbf{b}_j, f(\mathbf{b}_j)]$;
$\mathbf{p}_m = [\mathbf{b}_i + 0.5(\mathbf{b}_j - \mathbf{b}_i), f(\mathbf{b}_i + 0.5(\mathbf{b}_j - \mathbf{b}_i))]$;
$\mathbf{d}_j = \mathbf{p}_j - \mathbf{p}_i$;
$\mathbf{d}_m = \mathbf{p}_m - \mathbf{p}_i$;
$\mathbf{p}_{projection} = \frac{\mathbf{d}_m \otimes \mathbf{d}_j}{|\mathbf{d}_j|}$ (where $\otimes$ is the dot product);
if $\mathbf{d}_m \otimes \mathbf{d}_j \leq 0$ then
    $netaff = \sigma(\mathbf{p}_m, \mathbf{p}_i)$;
end
else if $|\mathbf{d}_j| \leq \mathbf{d}_m \otimes \mathbf{d}_j$ then
    $netaff = \sigma(\mathbf{p}_m, \mathbf{p}_j)$;
end
else
    $\mathbf{p}_n = \mathbf{p}_i + \mathbf{p}_{projection} \otimes \mathbf{d}_j$;
    $netaff = \sigma(\mathbf{p}_m, \mathbf{p}_n)$ (since $\overline{\mathbf{p}_m\mathbf{p}_n} \perp \overline{\mathbf{p}_j\mathbf{p}_i}$);
end
if $netaff < \varepsilon_{network}$ then
    Remove the ALC with the worst fitness;
end
```

discussed in section 4.3.1). The algorithm initially has a randomly generated population of ALCs which at each generation goes through three different phases to adapt to the training patterns (antigens). These phases are evaluation, cloning and recruitment, discussed next.

The evaluation phase differentiates the ALCs into exogenic activated or endogenic activated ALCs. Exogenic ALCs are activated by antigens in the current environment (at time t). Endogenic ALCs are not activated by the current environment and are equally reinforced by their individual densities. A number of the best exogenic ALCs are then selected in the cloning phase for cloning and mutation. The remaining endogenic ALCs are cloned without mutation. Mutated exogenic ALCs which do not improve the activation level of their corresponding parent exogenic ALCs are discarded.

In the final recruitment phase, the new population of ALCs consists of all the cloned endogenic ALCs and a selection of mutated exogenic ALCs. The latter selection process is based on a tournament selection where a number of mutated exogenic ALCs challenge each ALC in the previous generation's population.

The SAIS model is applied to time dependent optimisation problems to test the adaptability of the model in non-stationary environments. SAIS is able to track changing optima as well as memorising previously encountered optima. Scenarios do occur where previously encountered optima are forgotten. Thus the memory of SAIS is unstable and the authors proposed *Yet Another SAIS* (YASAIS) as an enhancement [58].

In YASAIS the population of ALCs are partitioned into sub-populations. In YASAIS there are no endogenic ALCs. Instead, the remaining ALCs (non-exogenic) are preserved within their respective sub-populations to the next generation. A single exogenic ALC is selected from each sub-population which goes through the same cloning and recruitment phase as in SAIS. The YASAIS did not deliver the expected improved results and was further enhanced in [59].

4.7 Idiotypic Network Topologies

The formation of idiotypic networks between lymphocytes (or their corresponding antibodies) can be defined by different network topologies. In the preceding section on network based artificial immune systems, the network interaction or network formation between artificial lymphocytes is either determined by a proximity matrix of network affinities or the grouping of similar artificial lymphocytes in sub-networks. The former is normalised with a network affinity threshold to determine the network links between the artificial lymphocytes and the latter utilises a clustering algorithm. There are, however, alternative and less familiar network topologies to determine the possible interactions in an idiotypic network of lymphocytes. The different theoretical approaches to determine the possible interactions in an idiotypic network are discussed next. Each of these network topologies specifies the interconnections between lymphocytes and the binding strength of these connections.

The Linear Topology: Lymphocytes in the linear topology are positioned as a sequence of different idiotypic levels of interaction. The linear topology was introduced by Richter and proposed as a *chain-reaction* between lymphocytes at different idiotypic levels [152, 153]. Figure 4.4 illustrates the linear topology of an idiotypic network with l idiotypic levels.

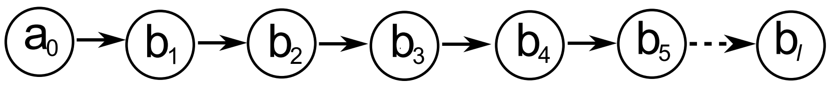


Figure 4.4 Linear Network Topology

The antigen, $\mathbf{a}_0$, is positioned at idiotypic level 0. Lymphocytes, $\mathbf{b}_i$, at idiotypic level i interact with lymphocytes at idiotypic levels $i-1$ and $i+1$ by either stimulating or suppressing neighbouring lymphocytes in the sequence. In figure 4.4, lymphocytes in idiotypic layer i stimulate the lymphocytes in layer $i+1$, which in turn stimulate the lymphocytes in layer $i+2$ and so forth. Lymphocytes in an idiotypic layer also suppresses the layer of lymphocytes responsible for its stimulation. Thus, suppression between idiotypic layers follows a *chain-reaction* in the reverse order to that of stimulation between layers. Lymphocytes in idiotypic layer l suppress the lymphocytes in layer $l-1$, which in turn suppress the lymphocytes in layer $l-2$ and so forth.

The Simple Cyclic Topology: Hiernaux discovered that the dynamical behaviour of the linear topology is dependent on whether l is odd or even [84]. Hiernaux converted the linear topology into a cyclic topology, as illustrated in figure 4.5.

The Affinity Matrix Topology: Figure 4.6 illustrates an example of an affinity matrix. Each element in the matrix specifies the interaction between two lymphocytes, while the magnitude of the element specifies the strength (binding affinity) between two lymphocytes.

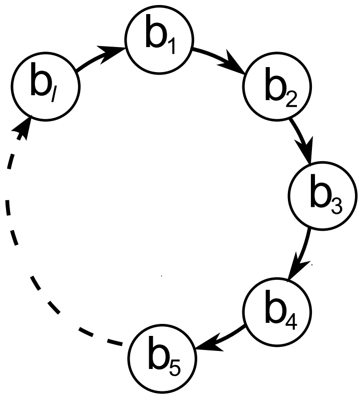
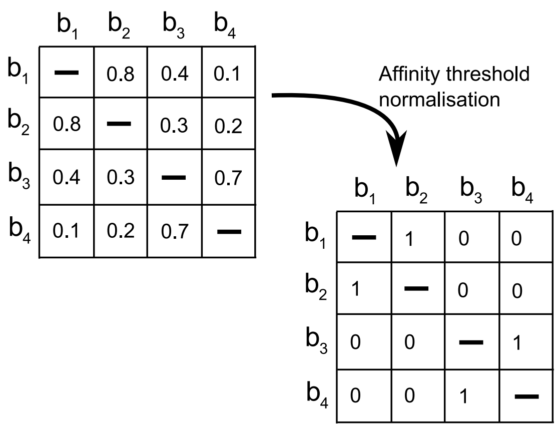


Figure 4.5 Simple Cyclic Network Topology



	b_1	b_2	b_3	b_4
b_1	—	0.8	0.4	0.1
b_2	0.8	—	0.3	0.2
b_3	0.4	0.3	—	0.7
b_4	0.1	0.2	0.7	—

	b_1	b_2	b_3	b_4
b_1	—	1	0	0
b_2	1	—	0	0
b_3	0	0	—	1
b_4	0	0	1	—

Figure 4.6 Affinity Matrix Topology with Normalisation

Thus, for a set of numbered lymphocytes, the element at position $(1,2)$ in the affinity matrix, specifies the binding affinity between lymphocyte number 1 and lymphocyte number 2 in the set [18]. Each of the elements in the affinity matrix can be measured against an affinity threshold value, normalising the affinity value of each element into either 0 or 1. A value of 1 implies idiotypic interaction between the lymphocytes and a value of 0 implies no interaction.

The Cayley Tree Topology: A Cayley tree is a loop-less tree. The node which contains the lymphocyte with the highest affinity with an antigen forms the root node of a Cayley tree [149, 181]. Only the root node, $\mathbf{b}_1$, reacts to the antigen, $\mathbf{a}$. The number of idiotypic network connections, z, is the number of connected neighbours for each node. z determines the complexity of the Cayley tree topology. Figure 4.7 illustrates the Cayley tree topology for an idiotypic network of lymphocyte nodes with $z = 3$ [181].

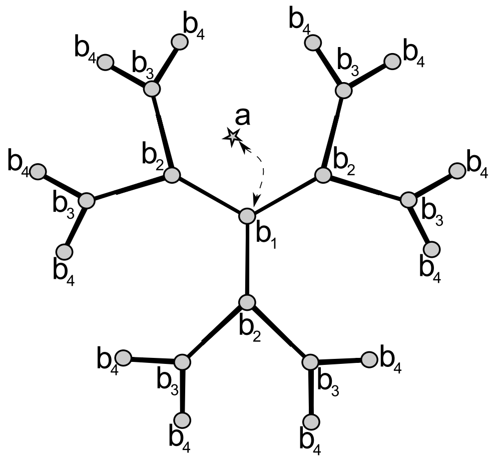


Figure 4.7 Cayley Tree Network Topology

Connectivity between the root node and the remaining lymphocyte nodes are determined with the affinity matrix topology (as discussed above). The distance between the root node and each of the connected lymphocyte nodes determines the idiotypic level for each of the connected lymphocyte nodes. Thus, with the root node at idiotypic layer 1 ($\mathbf{b}_1$), connected lymphocytes in idiotypic layer 2 ($\mathbf{b}_2$) have a closer distance to $\mathbf{b}_1$ than lymphocytes in idiotypic layer 3 ($\mathbf{b}_3$). The lymphocytes in $\mathbf{b}_3$ have a closer distance to $\mathbf{b}_1$ than lymphocytes in $\mathbf{b}_4$, etc. [149]. Therefore, the nodes are organised in a hierarchical manner, based on the measured distance to the root node. The total stimulation received by lymphocytes within a node at idiotypic layer i, is defined as [181]

$$\vartheta^i = \nu^{(i-1)} + (z-1)\,\nu^{(i+1)} \tag{4.43}$$

where ν^j denotes the concentration of lymphocytes in idiotypic layer j.

4.8 Danger Theory Models

The classical view of natural immunity is able to distinguish between *self* and *non-self* cells (as discussed in section 3.1). This ability is realised through the maturation process of T-Cells to become self-tolerant. In contrast to the classical theory, the danger theory further distinguishes the *non-self* cells as *dangerous* or *non-dangerous* (as discussed in section 3.7). The danger theory considers a cell to be *dangerous* if the cell instigates a danger signal of necrotic cell death to activate the antigen presenting cells. The activated antigen presenting cells co-stimulate the mature T-Cells, which in turn stimulate the B-Cells to react to the *foreign* cell. One of the motivations for the danger theory is that the natural immune system is able to adapt to a changing *self*, since the natural immune system only reacts to *dangerous non-self* cells. This inspired the modelling of the danger theory in AIS.

The main difference of the danger AIS models to those AIS models inspired by the classical view (as discussed in section 4.4), is the inclusion of a *signal* to determine whether a *non-self* pattern is *dangerous* or not. Therefore, a danger AIS model needs to define a signal of *death* or *danger* which is problem specific. The remainder of this section briefly discusses some of the applications of the danger AIS models to highlight the significance of the *danger* signal.

A familiar application of classical AIS models is in the field of network intrusion detection [53, 111, 114]. Intrusion detection AIS models create profiles of the *normal* incoming traffic at different nodes in the network. These *normal* profiles are seen as *self*. Through the application of the negative selection technique, abnormal traffic detectors (self-tolerant detectors) are generated from these *normal* profiles. The incoming traffic at each node is then monitored and the model signals an alarm of intrusion whenever an abnormal traffic detector is activated. A major drawback to this kind of traffic profiling is the assumption that *normal* traffic patterns never change. The fact that *normal* traffic patterns do change over time, results in outdated profiles with obsolete detectors. Therefore, an intrusion detection system needs to be adaptable.

An alternative approach to adapt to changes in *normal* traffic flow is to only signal an alarm of intrusion when the monitored host senses *danger*. In this context, *danger* can be defined as the sensing of any abnormal CPU load, memory usage, excessive I/O reads and writes, or security attacks. Whenever an abnormal traffic detector is activated without a danger signal from the host, the profile of *normal* traffic is adapted to accommodate the detected *normal* traffic pattern, resulting in an adaptive intrusion detection system. Danger AIS models as adaptive intrusion detection systems are proposed in [2, 4].

In a network with a dynamic topology, a change in *normal* traffic can also occur whenever a node is removed or added to the network, or when a node *misbehaves*. A mobile ad-hoc network is an example of such a network. A mobile ad-hoc network consists of terminal nodes, each with a radio as communication device to transmit information to other terminal nodes in the network, i.e. no infrastructure between nodes. Thus, nodes not only function as terminals, but also as relays of the transmitted information in the network. A node can misbehave whenever the node does not relay information to neighbouring nodes, or the node experiences hardware failures, or malicious software (like viruses) on the node try to overthrow the network. Sarafijanovic and Le Boudec proposed a danger theory inspired AIS to detect misbehaving nodes in a mobile ad-hoc network [155]. In this context a misbehaving node is dangerous.

Each node monitors the traffic from neighbouring nodes as *normal/self* and generates self-tolerant detectors using negative selection. Whenever a detector detects an incoming self pattern from a neighbouring node, the detector is replaced by a newly generated self-tolerant detector. Each node keeps a buffer of incoming traffic patterns from neighbouring nodes and self-tolerant detectors are frequently generated from the buffer. If a source node experiences danger (misbehaving node due to packet loss), the source node will generate an observation with a danger signal along the route where the packet loss was experienced. The action taken by the neighbouring nodes is to discard the observation from the buffered observations through correlation with the danger signal (also observed). This prevents the generation of detectors on *non-self* observations.

A similar buffering approach of *normal* patterns is taken in [158]. The danger theory inspired AIS by Secker *et al.* [158] simulates an adaptive mailbox. The proposed AIS classifies *interesting* from *uninteresting* emails. Initially, the user's actions on the incoming mail is monitored. If an email is deleted by the user, a detector is generated to detect the deleted email and added to a set of detectors. After adding a new detector to the set, the existing detectors in the set are cloned and mutated to improve the generalisation of the set. Thus, the set represents *non-self*/uninteresting email. The process continues until the size of the detector set reached a certain maximum.

The detector set adapts to the changing behaviour patterns of the user by buffering deleted emails as a set of *non-self* emails. As soon as the buffered set reaches a specific size, it is represented to the detector set of uninteresting emails. The detector set adapts to the buffered set through clonal selection.

Danger in this model is defined and measured as the number of unread emails in the inbox. Danger is signalled when the number of unread emails reaches a limit. When the model receives a danger signal, the unread emails are presented to the set of detectors for detection of *uninteresting* emails. The *uninteresting* classified emails are then moved to a temporary folder or deleted.

Table 4.1 List of Immunological Terms Mapped to Data Analysis Terms

Immunology	**Data Analysis**
Antigen set	Data set of patterns that needs to be clustered
ALC population	Clustered data set
Affinity	Measure of (dis)similarity between patterns
Network of ALCs	Cluster of patterns
Mean vector of ALC network (or representative ALC in ALC network)	Centroid of a cluster

4.9 Conclusion

This chapter highlighted the basic components of an AIS model. These components were categorised within an AIS framework. These categories are the representation of an ALC and antigen structure within a search space, the interaction between these structures (affinity measures), and the adaptation of the ALC structures through a selection strategy. The categories of the framework were then discussed with reference to proposed theoretical AIS models.

The chapter continued with an overview of the *shape space* model wherein the structure of an ALC and/or antigen can be defined and represented. In order to determine the affinity between the structure representing an ALC and the structure representing an antigen in *shape space*, the chapter gave a discussion on the different affinity measures within different *shape spaces*. Three of the most familiar affinity matching rules in a binary *shape space* were highlighted with their drawbacks of *holes*. After the discussion of affinity measures, the different selection strategies to adapt the ALCs structures were discussed.

Each selection strategy gave an overview of existing AIS models based on the specific strategy with a more detailed overview of AIS models which are based on the idiotypic network formation strategy. The discussion of the idiotypic network formation also introduced different theoretical approaches/network topologies to determine the possible interactions in an idiotypic network.

Table 4.1 provides a list of immunological terms which are mapped to data analysis terms. In the context of data clustering, the data set that needs to be clustered by an AIS model is seen as the set of antigen patterns. The clustered data set is represented by the population of ALCs. The measure of (dis)similarity between feature vectors (as discussed in section 2.2) determines the affinity between an ALC and an antigen pattern or another ALC (in the case of network based AIS models). In network based AIS models, each ALC network is a potential cluster in the data set (antigen set). The centroid of the cluster (ALC network) is calculated as the mean vector of ALCs or is a representative ALC in the ALC network.

The next chapter proposes and presents a novel network theory inspired artificial immune system.

Chapter 5

A Local Network Neighbourhood Artificial Immune System with Application to Unsupervised Data Clustering

The co-operation and co-stimulation or suppression between lymphocytes to respond and adapt to invading antigens can result in the formation of lymphocyte network structures in the natural immune system, according to the network theory of immunology. An antigen stimulated lymphocyte not only secretes antibodies but also proliferates by generating mutated clones to adapt to the antigen structure. The proliferation of a lymphocyte stimulates the immediate neighbouring lymphocytes, which in turn might also proliferate to adapt to the antigen structure and stimulate neighbouring lymphocytes. Thus, a network of lymphocytes *learns* the structure of an antigen by co-stimulating each other. The network topology of co-stimulated lymphocytes inspired the modelling of the local network neighbourhood artificial immune system (LNNAIS). The different parts of the LNNAIS algorithm are discussed in sections 5.1 to 5.4. The differences and similarities between existing network based AIS models and the proposed LNNAIS are discussed in section 5.5.

5.1 The Algorithm

The proposed LNNAIS algorithm is given in pseudo code in algorithm 5.1 and consists of seven high level steps to respond to an antigen/training pattern. Figure 5.1 shows a flow chart for the steps in the LNNAIS algorithm. These steps are:

1. Initialise the ALC population

2. Present an antigen to each ALC in the population and return the ALC with the highest calculated binding *affinity* with the antigen.

3. The returned highest affinity ALC reacts to the antigen pattern by initialising the antigen pattern as an antigen mutated clone and binds to the clone.

4. If the highest affinity ALC *activates*, the activated ALC spawns a mutated clone.

5. The spawned clone then binds to those antigen mutated clones of the activated ALC with which the spawned clone has a higher binding affinity than the activated ALC.

6. The mutated clone or activated ALC then *co-stimulates* ALCs which is within the *local neighbourhood* of the activated ALC.

7. Co-stimulation of neighbouring ALCs can result in co-suppression and/or the non-proliferation of other ALCs in the population.

The first step initialises the ALC population. The second and third step simulate the *affinity maturation* of a lymphocyte in the natural immune system. The second step models the *clonal selection* of the natural immune system. The antigen pattern selects the ALC with which the antigen has the highest binding affinity for cloning. The third step models the *proliferation* of a lymphocyte in the natural immune system. When a lymphocyte reaches a certain level of proliferation (clone size), the lymphocyte activates and spawns a mutated clone (*somatic hyper mutation* in the fourth step). The fifth and sixth steps simulate the network theory of co-stimulation and/or suppression, and the final step the non-proliferation of other lymphocyte clones due to the proliferation of neighbouring lymphocytes. The above high level steps are grouped into four phases, namely *initialise*, *react*, *adapt* and *suppress*. Each of these phases are explained next.

5.2 Initialising an ALC and the ALC population

The ALC population, $\mathcal{B}$, in LNNAIS is initialised as an empty set. The ALC population expands to a maximum size, $\mathcal{B}_{max}$, over time. The patterns in data set, $\mathcal{A}$, that needs to be partitioned are seen as antigen patterns and are randomly presented to the ALC population. The ALCs and antigen mutated clones in LNNAIS are encoded with the same structure as the antigen patterns in $\mathcal{A}$. If patterns in the data set are real-valued (or binary) vectors then the ALCs and antigen mutated clones are also real-valued (or binary) vectors. ALCs with antigen mutated clones are used in LNNAIS to adapt to the antigen patterns to form network structures and eventually cluster

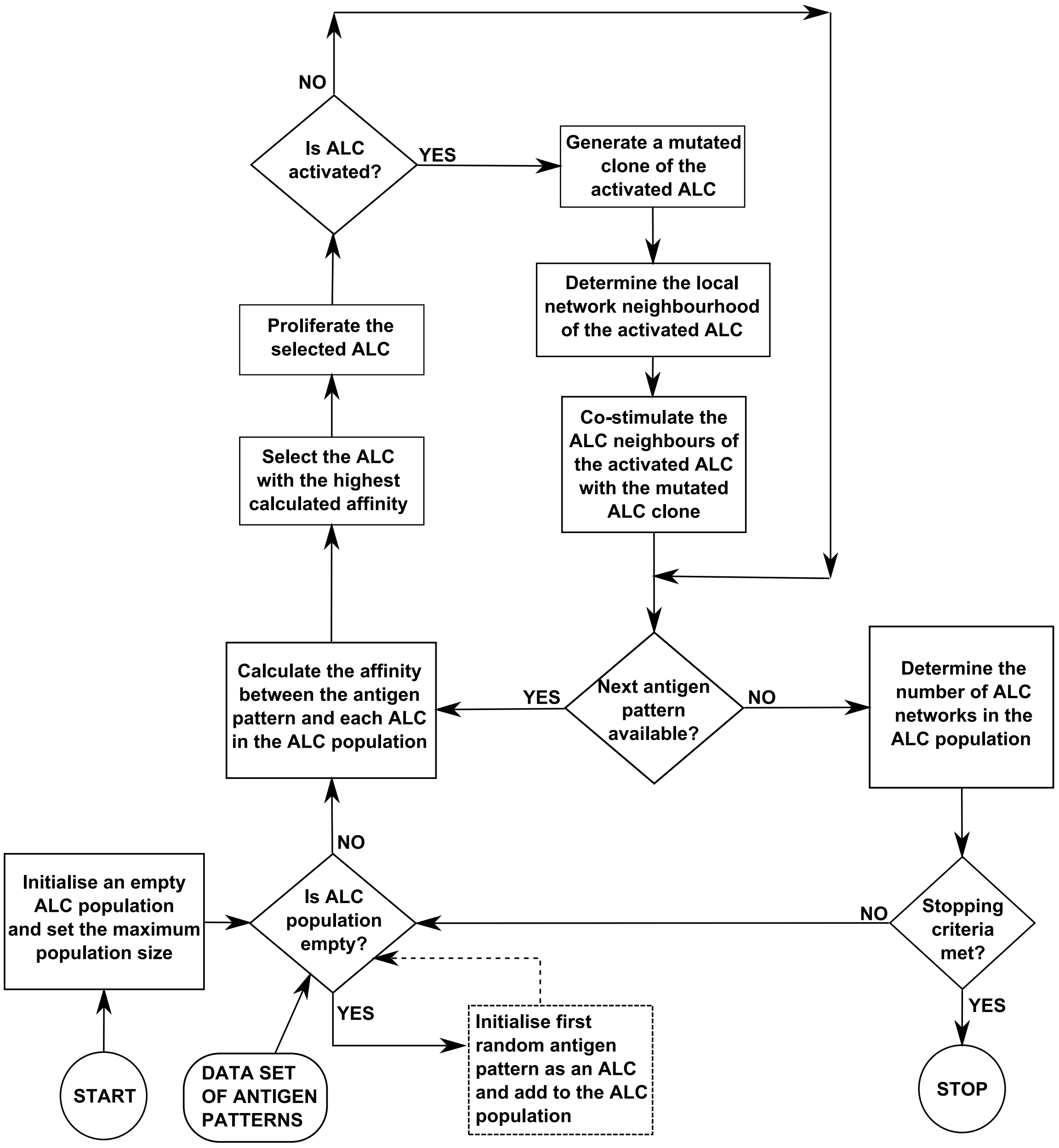

NO
Is ALC activated?
YES
Generate a mutated clone of the activated ALC
Determine the local network neighbourhood of the activated ALC
Co-stimulate the ALC neighbours of the activated ALC with the mutated ALC clone
Proliferate the selected ALC
Select the ALC with the highest calculated affinity
Calculate the affinity between the antigen pattern and each ALC in the ALC population
YES
Next antigen pattern available?
NO
Determine the number of ALC networks in the ALC population
NO
Initialise an empty ALC population and set the maximum population size
Is ALC population empty?
NO
Stopping criteria met?
YES
Initialise first random antigen pattern as an ALC and add to the ALC population
YES
START
DATA SET OF ANTIGEN PATTERNS
STOP

Figure 5.1 Flow chart of LNNAIS algorithm

Algorithm 5.1: High Level LNNAIS Algorithm

```
Set the maximum size of the ALC population as $\mathcal{B}_{max}$;
Initialise an empty set of ALCs as population $\mathcal{B}$;
for each antigen, $\mathbf{a}_j \in \mathcal{A}$, at index position $j$ in $\mathcal{A}$ do
    if $|\mathcal{B}| = 0$ (empty population of ALCs) then
        Initialise a new ALC, $\mathbf{b}$, with the same structure as pattern $\mathbf{a}_j$;
        $\mathcal{B} = \mathcal{B} \cup \mathbf{b}$;
    end
    Calculate the antigen affinity between $\mathbf{a}_j$ and each $\mathbf{b}_i \in \mathcal{B}$ using equation (2.3);
    Select $\mathbf{b}_h \in \mathcal{B}$, at index $h$, as the ALC with highest calculated antigen affinity;
    Proliferate $\mathbf{b}_h$ as discussed in section 5.3.2;
    if $\mathbf{b}_h$ is activated ($|\mathcal{C}_h| > \varepsilon_{clone}$) then
        Generate a mutated clone, $\mathbf{b}_h^{'}$, using equation (5.4);
        Secrete an antibody, $\mathbf{b}^*$, as discussed in section 5.3.4;
        Determine the local network neighbourhood of $\mathbf{b}_h$ using equation (5.5);
        Co-stimulate the local network neighbourhood of $\mathbf{b}_h$ with $\mathbf{b}^*$, as discussed in
        section 5.4.3;
    end
end
```

the data set. The initialisation of antigen mutated clones and the insertion of initialised ALCs into $\mathcal{B}$ are discussed next.

5.3 Reacting to an Antigen

The high level steps of the *react* phase are basically the steps responsible for calculating the affinity levels between the ALCs in population $\mathcal{B}$ and an antigen, selecting the ALC with the highest affinity and proliferating the selected ALC. The sections to follow explain and define each of these aspects.

5.3.1 Calculating the Affinity

The affinity between an antigen pattern, $\mathbf{a}$, and an ALC, $\mathbf{b}$, is known as the antigen affinity and is calculated as the Euclidean distance between $\mathbf{b}$ and $\mathbf{a}$. Euclidean distance is defined in equation (2.3) and is also used to measure the network affinity between two ALCs. The affinity determines the binding strength between an ALC and an antigen pattern or neighbouring ALC. Therefore, a lower Euclidean distance implies a higher affinity (stronger binding) between an

ALC and an antigen pattern or neighbouring ALC, and vice versa.

5.3.2 Proliferating the Clonal Selected ALC

The ALC with the highest binding affinity with an antigen pattern is selected as $\mathbf{b}_h$, where h is the index position of the selected ALC in $\mathcal{B}$. The antigen pattern $\mathbf{a}$ is then initialised as an antigen mutated clone $\mathbf{a}^{'}$. The antigen mutated clone $\mathbf{a}^{'}$ is grouped with $\mathbf{b}_h$ by inserting $\mathbf{a}^{'}$ at the first index position of the clonal set $\mathcal{C}_h$. Each ALC, $\mathbf{b}_i$, at index position i in $\mathcal{B}$, contains a set of antigen mutated clones, $\mathcal{C}_i$. Inserting an antigen mutated clone into $\mathcal{C}_i$ increases the clonal level of $\mathbf{b}_i$. Whenever the clonal level, $|\mathcal{C}|$, of an ALC exceeds the clonal level threshold, ε_{clone}, the ALC activates and generates a mutated ALC clone. When an antigen mutated clone is inserted at the first index of $\mathcal{C}$ and $|\mathcal{C}| > \varepsilon_{clone}$, the antigen mutated clone at the last index position $|\mathcal{C}|$, is removed from $\mathcal{C}$. This gives more current antigen mutated clones a higher probability to survive and influence the generation of the mutated ALC clone. The sections to follow discuss different definitions used to generate a mutated ALC clone.

5.3.3 Normalising the Affinity of an Antigen Mutated Clone

The normalised affinity between an antigen mutated clone, $\mathbf{a}^{'} \in \mathcal{C}_i$, and an ALC $\mathbf{b}_i$, is defined as

$$\sigma^{*}\left(\mathbf{b}_i, \mathbf{a}^{'}, \mathcal{C}_i\right) = 1.0 - \frac{\sigma\left(\mathbf{b}_i, \mathbf{a}^{'}\right)}{\sigma_{max} + 1.0} \tag{5.1}$$

where

$$\sigma_{max} = max_{c=1,\ldots,|\mathcal{C}_i|}\left\{\sigma\left(\mathbf{b}_i, \mathbf{a}^{'}_c\right)\right\} \tag{5.2}$$

and $\mathbf{a}^{'}_c$ is an antigen mutated clone at index position c in clonal set $\mathcal{C}_i$ of ALC $\mathbf{b}_i$. In the above definition, σ^{*} calculates the normalised affinity between an antigen mutated clone, $\mathbf{a}^{'}_c \in \mathcal{C}_i$, and an ALC, $\mathbf{b}_i$, with respect to the lowest affinity (highest Euclidean distance) in the set of antigen mutated clones, $\mathcal{C}_i$. A lower affinity between an antigen mutated clone and an ALC will result in a lower normalised affinity and vice versa. Thus the higher an ALC's affinity towards an antigen mutated clone, the more the ALC's clone will be mutated towards the antigen mutated clone, as explained in the next section.

5.3.4 Generating a Mutated Clone of an Activated ALC

The vector difference between two vectors $\mathbf{q}$ and $\mathbf{r}$ is defined as:

$$\theta(\mathbf{r}, \mathbf{q}) = \mathbf{q} - \mathbf{r} \tag{5.3}$$

The above function, θ, returns a vector with the same number of attributes (components) as $\mathbf{q}$. These attributes are calculated by subtracting each attribute in $\mathbf{r}$ from the corresponding attribute in $\mathbf{q}$. The set of antigen mutated clones, $\mathcal{C}_i$, which is contained by an ALC $\mathbf{b}_i$ determines the mutated clone which will be generated when an ALC is activated. The mutated clone, $\mathbf{b}_i^{'}$, is calculated using

$$\mathbf{b}_i^{'} = \mathbf{b}_i + \frac{\sum_{c=1}^{|\mathcal{C}_i|} \sigma^* \left(\mathbf{b}_i, \mathbf{a}_c^{'}, \mathcal{C}_i\right) \theta\left(\mathbf{b}_i, \mathbf{a}_c^{'}\right)}{\sum_{c=1}^{|\mathcal{C}_i|} \sigma^* \left(\mathbf{b}_i, \mathbf{a}_c^{'}, \mathcal{C}_i\right)} \tag{5.4}$$

In the above definition, $\mathbf{b}_i$ is mutated by adding a calculated average vector (second term in equation (5.4)) to $\mathbf{b}_i$. The numerator of the fraction in the second term contains the product of the normalised affinity between $\mathbf{b}_i$ and an antigen mutated clone, and the vector difference between $\mathbf{b}_i$ and the applicable antigen mutated clone. The normalised affinity between an ALC and an antigen mutated clone was discussed in section 5.3.3. The influence of the vector difference between $\mathbf{b}_i$ and an antigen mutated clone is therefore weighted by the normalised affinity. The numerator is thus calculated as the sum of weighted vector differences for all the antigen mutated clones contained by $\mathbf{b}_i$. Antigen mutated clones in $\mathcal{C}_i$ with a higher binding affinity with ALC $\mathbf{b}_i$ have a higher influence on the mutation of the clone in comparison with antigen mutated clones with a lower binding affinity. The result is that the ALC clone is mutated more towards higher affinity antigen mutated clones in $\mathcal{C}_i$. The calculated sum of weighted vector differences (numerator) is then divided by the sum of the normalised affinities to obtain an average vector for mutating $\mathbf{b}_i$.

5.3.5 Secreting an Antibody for Co-stimulation

The antigen mutated clones in $\mathcal{C}_i$ with which $\mathbf{b}_i^{'}$ has a higher affinity than the parent ALC $\mathbf{b}_i$, is added to the clonal set of $\mathbf{b}_i^{'}$ (bind to $\mathbf{b}_i^{'}$). If more than half of the number of antigen mutated clones in $\mathcal{C}_i$ bind to $\mathbf{b}_i^{'}$, the parent ALC $\mathbf{b}_i$ is added as an antigen mutated clone to the clonal set of $\mathbf{b}_i^{'}$. The parent ALC is then replaced by $\mathbf{b}_i^{'}$ in $\mathcal{B}$ and secreted as a co-stimulating antibody to neighbouring ALCs. If less than half of the number of antigen mutated clones in $\mathcal{C}_i$ bind to $\mathbf{b}_i^{'}$, the parent ALC $\mathbf{b}_i$ is suppressed by removing all of the antigen mutated clones in $\mathcal{C}_i$. This prevents frequently activated ALCs from dominating the population. The mutated ALC clone, $\mathbf{b}_i^{'}$, is then inserted into $\mathcal{C}_i$; not only to co-stimulate the parent ALC, but also to preserve the memory of the antigen structure. The mutated ALC clone is secreted as a co-stimulating antibody to neighbouring ALCs. The following section discusses the co-stimulation of neighbouring ALCs within a local network neighbourhood.

5.4 Adapting the ALCs in a Local Network Neighbourhood

The co-stimulating antibody which is secreted during the activation of a proliferated ALC is presented to the immediate ALC neighbour(s) in the local network neighbourhood of the activated ALC. The neighbouring ALCs within a local network neighbourhood adapt to the antibody as it would react to an antigen (as explained in section 5.3). The following sections discuss the manner in which a local network neighbourhood of an activated ALC is determined.

5.4.1 Determining the Local Network Neighbourhood of an Activated ALC

An ALC's neighbourhood, $\mathcal{N}$, is determined by a network neighbourhood window of size, ρ, and the highest average network affinity between the potential neighbouring ALCs. The neighbourhood, $\mathcal{N}_{i,\rho}$, of an ALC, $\mathbf{b}_i \in \mathcal{B}$, is defined as

$$\mathcal{N}_{i,\rho} = \left\{ \forall \mathbf{b}_j \in \mathcal{B} : \min_{j=i-(\rho-1),\ldots,i} \{\mu(j, j+(\rho-1))\} \right\} \tag{5.5}$$

where

$$\rho \leq |\mathcal{B}| \tag{5.6}$$

$$\mathcal{N}_{i,\rho} \subseteq \mathcal{B} \tag{5.7}$$

$$\mathbf{b}_i \in \mathcal{N}_{i,\rho} \tag{5.8}$$

and μ calculates the average network affinity between ALCs in the population from index position i to $i+(\rho-1)$; μ is defined in section 5.4.2. The above definition is a network window of size ρ which starts at position $i-(\rho-1)$, sliding over the ALC population in search of the highest average network affinity (minimum average distance). Figure 5.2 illustrates a local network neighbourhood where $\rho = 5$ and the network with the highest average network affinity starts at index position $h-2$.

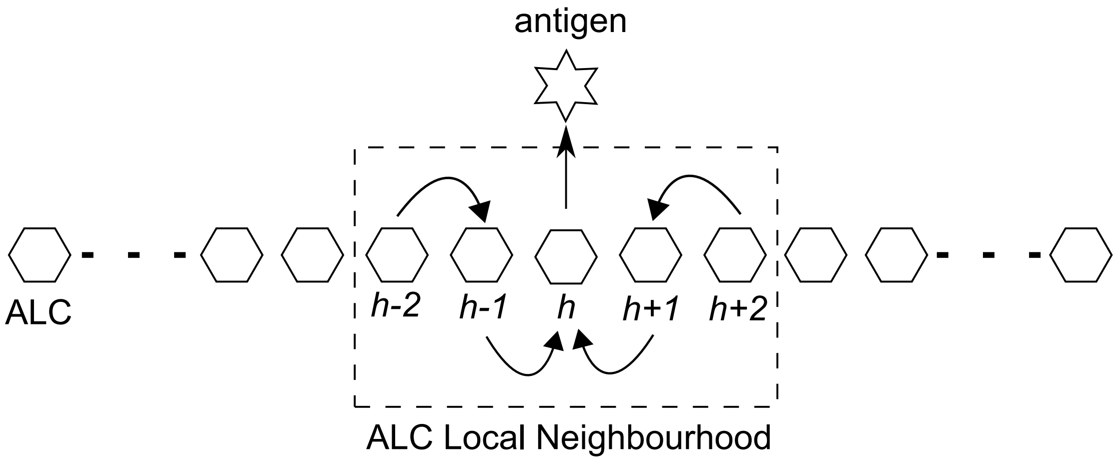


Figure 5.2 Adapting an ALC Network Neighbourhood

5.4.2 Average Network Affinity in a Local Network Neighbourhood

The average network affinity level of a network of ALCs starting at index position x to y, is defined as

$$\mu(x,y) = \frac{\sum_{i=x}^{y-1} \sigma(\mathbf{b}_i, \mathbf{b}_{i+1})}{y-x} \tag{5.9}$$

where σ is the Euclidean distance (as defined in equation (2.3)).

5.4.3 Co-stimulating the Local Network Neighbourhood

The neighbouring ALCs within a local network neighbourhood, $\mathcal{N}_{i,\rho}$, adapt to the secreted antibody of its predecessor in the neighbourhood. Figure 5.2 illustrates a local network neighbourhood with $\rho = 5$ adapting to an antigen. In this figure, ALC $\mathbf{b}_h$ is selected by the antigen for cloning and proliferation (as explained in section 5.3.2). As a result of proliferating $\mathbf{b}_h$, the ALC became active ($|\mathcal{C}_h| > \varepsilon_{clone}$) and secreted an antibody for co-stimulation of the immediate neighbours of $\mathbf{b}_h$. The immediate neighbours of $\mathbf{b}_h$ at indices $h-1$ and $h+1$ react to the secreted antibody by adding the clonal set of the antibody to $\mathcal{C}_{h-1}$ and $\mathcal{C}_{h+1}$, respectively. If either or both of the neighbouring ALCs, $\mathbf{b}_{h-1}$ and $\mathbf{b}_{h+1}$ becomes activated, either or both will secrete antibodies (as explained in section 5.3.4), which will co-stimulate their immediate ALC neighbours at indices $h-2$ and $h+2$, respectively. If a neighbouring ALC is not activated by the co-stimulation of a predecessor's antibody, the antibody is inserted into the local network at the index of the neighbouring ALC, increasing the population size through *clonal expansion* (discussed in section 5.4.4). The neighbouring ALCs with the highest network affinity in the population, which are not within the local network neighbourhood, are merged to stabilise the population size. Merging of ALCs simulate the non-proliferation of other ALC clones in the population (discussed in section 5.4.5). The process of co-stimulation continues until the ALCs on the boundary of the local network neighbourhood are co-stimulated or until a neighbouring ALC is not activated by the co-stimulation of a predecessor's antibody. Algorithm 5.2 lists the pseudo code for adapting the ALCs in a local network neighbourhood.

5.4.4 Clonal Expansion of a Local Network Neighbourhood

A local network neighbourhood is clonally expanded whenever a neighbouring ALC, $\mathbf{b}_i$, is not activated by the co-stimulation of a predecessor's secreted antibody. The secreted antibody, $\mathbf{b}^*$, is inserted at position i^* which is defined as

$$i^*\left(\mathbf{b}^*, \mathbf{b}_i\right) = \begin{cases} i & \text{if } \frac{\sigma(\mathbf{b}^*,\mathbf{b}_{i-1})+\sigma(\mathbf{b}^*,\mathbf{b}_i)}{2} < \frac{\sigma(\mathbf{b}^*,\mathbf{b}_i)+\sigma(\mathbf{b}^*,\mathbf{b}_{i+1})}{2} \\ i+1 & \text{otherwise} \end{cases} \tag{5.10}$$

The secreted antibody is inserted at the index position where the average network affinity is the highest between the secreted antibody and its potential neighbouring ALCs.

5.4.5 Non-proliferation of the ALC Population

The maximum ALC population size, $\mathcal{B}_{max}$, is exceeded whenever clonal expansion occurs in a local network neighbourhood. Therefore, the non-proliferation and suppression of other ALCs in the population keeps the size of the ALC population stable. Non-proliferation (suppression) is simulated by merging two ALCs in the population which are not within the clonally expanded local network neighbourhood, and which have the highest network affinity in the population.

5.5 Similarities and Differences with Other Network based AIS Models

This section discusses some of the differences and similarities between the proposed algorithm and existing network based AIS models.

Algorithm 5.2: Adapting the Neighbourhood, $\mathcal{N}_{h,\rho}$, to an Activated ALC, $\mathbf{b}_h$

```
Let $\mathbf{b}^*$ be the secreted antibody of the activated ALC $\mathbf{b}_h$;
$l = h - 1; r = h + 1$;
Let $\mathbf{b}_l^* = \mathbf{b}^*$ and $\mathbf{b}_r^* = \mathbf{b}^*$ be the secreted antibodies for co-stimulation of neighbouring
ALCs $\mathbf{b}_l$ and $\mathbf{b}_r$, respectively;
Activated=true;
Costimulated=false;
for $\mathbf{b}_l \in \mathcal{N}_{h,\rho}$ and Activated do
    Add antigen mutated clones of $\mathbf{b}_l^*$ to clonal set $\mathcal{C}_l$ of neighbouring ALC $\mathbf{b}_l$;
    if $\mathbf{b}_l$ is activated (i.e. $|\mathcal{C}_l| > \varepsilon_{clone}$) then
        Generate a mutated clone, $\mathbf{b}_l^{'}$, using equation (5.4);
        Secrete an antibody $\mathbf{b}_l^*$ from $\mathbf{b}_l$, as discussed in section 5.3.4;
        $l = l - 1$;
        Costimulated=true;
    end
    else
        Activated=false;
        Insert $\mathbf{b}_l^*$ into $\mathcal{N}_{h,\rho}$ at position $i^*\left(\mathbf{b}_l^*, \mathbf{b}_l\right)$ (as defined in equation (5.10));
        Merge two ALCs in the population with the highest network affinity, as discussed
        in section 5.4.5;
    end
end
Activated=true;
for $\mathbf{b}_r \in \mathcal{N}_{h,\rho}$ and Activated do
    Add antigen mutated clones of $\mathbf{b}_r^*$ to clonal set $\mathcal{C}_r$ of neighbouring ALC $\mathbf{b}_r$;
    if $\mathbf{b}_r$ is activated (i.e. $|\mathcal{C}_r| > \varepsilon_{clone}$) then
        Generate a mutated clone, $\mathbf{b}_r^{'}$, using equation (5.4);
        Secrete an antibody $\mathbf{b}_r^*$ from $\mathbf{b}_r$, as discussed in section 5.3.4;
        $r = r + 1$;
        Costimulated=true;
    end
    else
        Activated=false;
        Insert $\mathbf{b}_r^*$ into $\mathcal{N}_{h,\rho}$ at position $i^*\left(\mathbf{b}_r^*, \mathbf{b}_r\right)$ (as defined in equation (5.10));
        Merge two ALCs in the population with the highest network affinity, as discussed
        in section 5.4.5;
    end
end
if not Costimulated and $|\mathcal{B}| < \mathcal{B}_{max}$ then
    Insert $\mathbf{b}^*$ into $\mathcal{N}_{h,\rho}$ at position $i^*\left(\mathbf{b}^*, \mathbf{b}_h\right)$ (as defined in equation (5.10));
end
```

5.5.1 Training Data

Although the proposed LNNAIS model can be trained on normalised data, the normalisation of training data is not a prerequisite for LNNAIS. Similar to other network based AIS models, LNNAIS sees all training patterns as antigen patterns.

5.5.2 Population of ALCs

The population of ALCs can be initialised with a number of randomly initialised ALCs or a number of randomly selected training patterns as ALCs, i.e. a cross section of the training data is used to initialise the ALCs. The initial population of ALCs in LNNAIS is an empty set. The first randomly selected training pattern is initialised as an ALC and added to the population of ALCs. This concept is known as *dendritic injection* in the natural immune system. The population of ALCs are grown and pruned in LNNAIS. The *growth* of the population of ALCs in LNNAIS is based on the process of *affinity maturation*. When an activated ALC of a local network neighbourhood does not adapt to the presented antigen pattern, the clonal level of the ALC is penalised and a mutated clone of the ALC is inserted into the local network of ALCs.

5.5.3 ALC Presentation

An ALC in LNNAIS is presented by a continuous-valued array with the same dimension as the antigen patterns in the training set, as is the case for other network based AIS models.

5.5.4 Affinity Measurement

The affinity between an antigen pattern and an ALC is measured using the Euclidean distance as defined in section 5.3.1. The affinity between two ALCs, referred to as network affinity, is also measured using the Euclidean distance. Some of the existing network based AIS models also measure antigen and network affinity using Euclidean distance. The difference between LNNAIS and the existing network based AIS models is that LNNAIS has no threshold to determine whether two ALCs are linked to form a network. LNNAIS introduces a new concept of an ALC network neighbourhood size, as defined in section 5.4.1 and proposed by Graaff and Engelbrecht [64].

5.5.5 Learning the Antigen Structure

Another similarity between existing network based AIS models and the proposed LNNAIS is that some ALCs are cloned and mutated to adapt to antigen patterns. LNNAIS also models the process of *affinity maturation* to introduce new ALCs into the population as discussed in section 5.4.3. LNNAIS also models the non-proliferation of ALCs, as discussed in section 5.4.3. The difference between LNNAIS and existing network based AIS models is that *expansion* of the ALC population is done on a per local network neighbourhood bases. LNNAIS models the *idiotopic network theory* of ALCs. This means that the insertion of new ALCs into a population will be done within a local network neighbourhood (as discussed in section 5.4.3). Non-proliferation on the other hand is only done on ALCs which do not form part of the *activated* local network neighbourhood. This means that only ALCs outside a network neighbourhood will be non-proliferated in the ALC population (as discussed in section 5.4.3). This approach penalises the population of ALCs by *non-proliferating* the population but also reinforces the network neighbourhood by *clonal expansion*.

5.5.6 Determining the Number of Clusters

The number of ALC networks formed in existing network based AIS models represent potential clusters in the data set. In most of the existing network based AIS models the number of ALC networks in a population is determined by a network affinity threshold or a hybrid approach is taken by clustering the ALC population into sub-nets (as discussed in section 4.6). The thresholding technique uses a proximity matrix of network affinities between the ALCs in the population. The ALCs with a network affinity below the threshold value are allowed to be linked and form networks. Therefore the specified value of the network affinity threshold determines the number of ALC networks and it can be a formidable task to specify the correct network affinity threshold to obtain the correct or required number of clusters. A potential drawback of the hybrid approach is that the clusters (sub-nets) might contain ALCs which do not have a *good* or generic representation of the data. Both of these techniques are also computationally expensive.

The proposed LNNAIS model has the advantage that an ALC can only link to its immediate neighbours to form an ALC network. This is due to the network topology and an index based neighbourhood technique. Therefore, there is no need for a network affinity threshold and/or a proximity matrix of network affinities to determine the number of ALC networks in LNNAIS. It is also not necessary to follow a hybrid approach of clustering the ALC population. Determining the number of clusters in LNNAIS is thus computationally less expensive and is explained next.

In order to obtain a specified number of clusters, K, the network affinities between neighbouring ALCs in the population need to be calculated. The boundaries of each cluster are then determined by pruning the network links between the K lowest calculated network affinities. Figure 5.3 illustrates this technique where $K = 3$. The edges between ALCs have an associated network affinity. The K edges that forms the boundaries between the ALCs (dotted lines) have the lowest network affinity in the ALC population, i.e. highest Euclidean distance. The centroid of each of the formed ALC networks (illustrated as clouds) is calculated using equation (2.18).

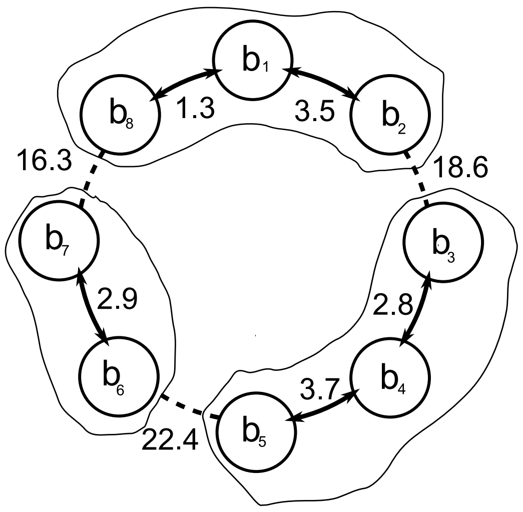


Figure 5.3 Determining the Number of Clusters in LNNAIS

5.5.7 The Number of Parameters

Focusing on existing network based AIS models which are used in the experimental work of this chapter, there is also a significant difference in the number of parameters that need to be specified for each of the models. The DWB model has a total of 12 parameters, SMAIN has a total of seven parameters and Opt-aiNet a total of six parameters. The proposed LNNAIS model has only three parameters which are the maximum population size, $\mathcal{B}_{max}$, the neighbouring radius, ρ, and the activation level for ALC cloning, ε_{clone}.

5.6 Time Complexity of LNNAIS

The time complexity of LNNAIS is based on the complexity of partitioning $\mathcal{A}$, sorting the network affinities between the ALCs in the ALC population and pruning K boundaries between the ALCs in the ALC population of size $\mathcal{B}_{max}$ to obtain K ALC networks (clusters). The time complexity of partitioning $\mathcal{A}$ is based on presenting $\mathcal{A}$ to ALC population $\mathcal{B}$ and adapting the ALC population. Assume that t_1 is the number of iterations taken by LNNAIS to converge. The worst case of time complexity for LNNAIS to partition $\mathcal{A}$ is when there is always an activated ALC in $\mathcal{B}$ and when the network neighbourhood size of the activated ALC is the entire ALC population ($\mathcal{N} = \mathcal{B}$). Then the time complexity of partitioning $\mathcal{A}$ is $O\left(t_1 |\mathcal{A}| (\mathcal{B}_{max})^2 N \chi_1\right)$ where $|\mathcal{A}|$ is the size of the data set that needs to be partitioned and N is the number of dimensions of $\mathcal{A}$. The χ_1 parameter is the time complexity for an activated ALC to generate an antibody for co-stimulation of neighbouring ALCs. The t_1, $\mathcal{B}_{max}$, N and χ_1 parameters are fixed in advance and usually $\mathcal{B}_{max} << |\mathcal{A}|$ and $\left|\mathcal{N}\right| << \mathcal{B}_{max}$. If $t_1 \mathcal{B}_{max} N \chi_1 << |\mathcal{A}|$ then the time complexity of partitioning $\mathcal{A}$ is $O(|\mathcal{A}|)$. If however, $\mathcal{B}_{max} \approx |\mathcal{A}|$ and $\left|\mathcal{N}\right| \approx \mathcal{B}_{max}$ then the time complexity of partitioning $\mathcal{A}$ is $O\left(|\mathcal{A}|^2\right)$. The maximum number of boundaries in an ALC population of size $\mathcal{B}_{max}$ is $\mathcal{B}_{max}$. The time complexity of sorting the $\mathcal{B}_{max}$ network affinities depends on the sorting algorithm used. Assume the time complexity of the sorting algorithm is some constant, χ_2. The worst case of time complexity for LNNAIS to determine K ALC networks is when $K = \mathcal{B}_{max}$, giving a time complexity of $O(\mathcal{B}_{max})$.

5.7 Experimental Results and Analysis

This section discusses and compares the clustering results obtained by K-means, CPSO, SMAIN, DWB, Opt-aiNet and LNNAIS. Furthermore, a sensitivity analysis of LNNAIS is done on the different data sets.

5.7.1 Data clustering problems

Table 5.1 lists the selection of data sets used to benchmark the clustering performance and quality of the proposed LNNAIS model against the clustering quality of existing clustering methods like K-means clustering and CPSO (as discussed in sections 2.3.2 and 2.7.1, respectively) and network based AIS models for data clustering like SMAIN, DWB-AIS and Opt-aiNet (as discussed in section 4.6). The characteristics of each data set are also listed in the table. These are the number of patterns in the dataset ($|P|$), the number of features per pattern in the data set (N

Table 5.1 List of Eleven Benchmarking Data Sets for Clustering

Category	Data set name	$\|P\|$	N	σ_{max}	K	Overlap?
Group 1	Iris	150	4	7.7	3	Y
	Two-spiral	190	2	3.045	12	Y
	Hepta	212	3	13.383	7	N
Group 2	Engytime	4096	2	14.806	2	Y
	Chainlink	1000	3	4.383	6	Y
	Target	770	2	8.627	5	Y (outliers)
Group 3	Ionosphere	351	34	11.358	2	Y
	Glass	214	9	16.449	6	Y
Group 4	Image Segmentation	2310	19	1775.117	7	Y
	Spambase	4601	57	18758.75	2	Y
	Letter Recognition	20000	16	60	26	Y

- number of dimensions), the maximum distance between the patterns in the data set (σ_{max}), the number of clusters selected for partitioning the data set (K) and whether there are any overlapping patterns in the data set. The two-spiral, hepta, engytime, chainlink and target data sets are part of a fundamental clustering problems suite [95]. The other data sets were collected from the UCI Machine Learning repository [6].

The data sets in table 5.1 can be categorised into four groups:

- Group 1 (small number of features / small number of patterns): The data sets within this group have a small number of features and a small number of patterns. The iris data set, two-spiral problem and hepta data set form part of this group. All of these data sets have less than 500 patterns and less than five features per pattern.
- Group 2 (small number of features / large number of patterns): The data sets within this group also have a small number of features but a larger number of patterns in comparison to the data sets in group 1. The engytime data set, chainlink data set and the target data set (to a lesser extent) form part of this group. All of these data sets have more than 500 patterns and less than five features per pattern.
- Group 3 (large number of features / small number of patterns): This group contains data sets with a larger number of features in comparison to groups 1 and 2, but a small number of patterns. The ionosphere data set and the glass data set form part of this group and both have less than 500 patterns, with each pattern having more than eight features.

- Group 4 (large number of features / large number of patterns): The last group contains data sets with a larger number of features (compared to groups 1 and 2) and a larger number of patterns (compared to groups 1 and 3). The image segmentation data set, spambase data set and letter recognition data set form part of this group. All of these data sets have more than 500 patterns and more than eight features.

Taken as a whole, the data sets listed in table 5.1 represent a good distribution of data clustering problems with the number of patterns in the range $[150, 20000]$ and the number of features in the range $[2, 57]$. All the data sets have overlapping patterns except the hepta data set. The target data set also contains outlier patterns.

5.7.2 Experimental setup and methodology

All experimental results in this chapter are averages taken over 50 runs, unless stated otherwise. The stopping criteria for all algorithms was set to 1000 iterations ($t_{max} = 1000$). Populations/Swarms in the respective algorithms were initialised by randomly selecting patterns from the data set. The patterns in a data set were randomly presented to each model. None of the data sets were normalised for training. All algorithms were implemented using the Java 6 framework which interfaced to a MySQL 5 database for collection of data sets and exporting of results. Algorithms were executed on a 24 core Sun Grid Engine. Tables 5.2 to 5.6 summarise the parameter values used by the respective algorithms for each data set. All parameter values for the respective algorithms were found empirically to deliver the best performance for clustering the applicable data set. The Q_{ratio} validity index (defined in equation (2.49)), intra error distance (J_{intra} as defined in equation (2.17)) and inter error distance (J_{inter} as defined in equation (2.16)) are used as performance measures to determine the clustering quality of the different models. These clustering performance measures were discussed in sections 2.3.2 and 2.4, respectively.

The following sections investigate whether there is a difference between the clustering quality, Q_{ratio}, of two models for a specific data set or not. The hypothesis is defined as

- *Null* hypothesis, H_0: There is no difference in Q_{ratio}.
- *Alternative* hypothesis, H_1: There is a difference in Q_{ratio}.

The above hypothesis was tested with a non-parametric Mann-Whitney U hypothesis test (0.95 confidence interval, i.e. $\alpha = 0.05$) between the clustering quality of LNNAIS and the clustering

Table 5.2 CPSO Parameter Values

Data set	K	$\lvert S\rvert$	d	w	c_1	c_2	δ
Iris	3	6	3	0.82	1.33	1.218	0.301
Two-spiral	12	9	4	0.558	0.656	1.94	0.326
Hepta	7	44	4	0.697	1.696	0.963	0.62
Engytime	2	63	11	0.641	0.719	0.156	0.359
Chainlink	6	11	5	0.234	0.656	1.969	0.266
Target	5	23	2	0.789	0.422	1.658	0.258
Ionosphere	2	45	8	0.683	1.518	1.207	0.961
Glass	6	13	6	0.914	1.344	1.246	0.115
Image Segmentation	7	10	5	0.77	0.875	1.545	0.312
Spambase	2	42	18	0.812	0.125	1.152	0.938
Letter Recognition	26	49	3	0.836	0.828	1.641	0.055

Table 5.3 SMAIN Parameter Values

Data set	K	$\mathcal{B}_{init}$	R_γ	R_Λ	NAT	R_k	R_{max}	R_{init}
Iris	3	0.25	0.836	3	1.115	0.422	238	37
Two-spiral	12	0.182	0.516	91	0.039	0.656	975	92
Hepta	7	0.191	0.938	38	0.259	0.375	900	88
Engytime	2	0.019	0.672	35	2.322	0.469	725	36
Chainlink	6	0.2	0.859	23	0.038	0.094	425	91
Target	5	0.049	0.824	22	0.077	0.852	819	31
Ionosphere	2	0.157	0.637	34	0.099	0.727	319	68
Glass	6	0.157	0.637	34	0.015	0.727	319	68
Image Segmentation	7	0.29	0.926	2	24.618	0.898	281	76
Spambase	2	0.123	0.805	33	6.571	0.359	388	43
Letter Recognition	26	0.051	0.93	24	8.595	0.109	988	68

Table 5.4 DWB Parameter Values

Data set	K	$\mathcal{B}_{max}$	ϕ_{init}	m_{min}	A	a_{min}	a_{max}	k_{clone}	ς	τ	τ_α	τ_β	$k_{compress}$
Iris	3	39	0.362	0.087	37	49	49	1.688	0.24	14	5	10	6
Two-spiral	12	46	0.668	0.025	55	6	6	2.438	0.913	12	13	13	5
Hepta	7	47	0.959	0.959	78	35	54	1.625	0.592	1	6	15	2
Engytime	2	39	0.485	0.209	24	24	86	3.188	0.852	6	6	1	4
Chainlink	6	40	0.592	0.102	41	72	91	2.125	0.714	7	11	2	2
Target	5	47	0.554	0.982	36	62	62	3.844	0.89	13	12	11	3
Ionosphere	2	17	0.561	0.929	44	7	68	1.25	0.929	5	7	9	3
Glass	6	46	0.845	0.018	13	11	11	4.031	0.569	2	5	9	7
Image Segmentation	7	47	0.201	0.538	16	2	2	2.906	0.477	12	14	7	1
Spambase	2	46	0.27	0.546	71	9	9	4.562	0.025	3	8	4	4
Letter Recognition	26	45	0.148	0.423	9	27	46	3.062	0.148	8	10	13	3

Table 5.5 Opt-aiNET Parameter Values

Data set	K	$\mathcal{B}_{init}$	η	$\varepsilon_{network}$	$\varepsilon_{fitness}$	φ	$\frac{1}{\varsigma}$
Iris	3	44	10	0.186	1.317	0.131	0.356
Two-spiral	12	14	1	0.324	0.902	0.219	0.169
Hepta	7	39	1	0.297	1.54	0.491	0.459
Engytime	2	29	2	0.037	0.412	0.403	0.322
Chainlink	6	7	22	0.178	0.723	0.306	0.283
Target	5	12	3	0.362	1.109	0.338	0.412
Ionosphere	2	28	3	0.477	1.97	0.294	0.144
Glass	6	35	1	0.155	0.961	0.456	0.431
Image Segmentation	7	14	5	0.021	1.184	0.316	0.134
Spambase	2	6	5	0.32	1.985	0.409	0.191
Letter Recognition	26	45	2	0.416	1.258	0.444	0.394

Table 5.6 LNNAIS Parameter Values

Data set	K	$\mathcal{B}_{max}$	ρ	ε_{clone}
Iris	3	14	3	8
Two-spiral	12	39	3	6
Hepta	7	29	3	6
Engytime	2	10	3	22
Chainlink	6	24	3	8
Target	5	28	3	6
Ionosphere	2	10	3	17
Glass	6	24	3	8
Image Segmentation	7	20	2	27
Spambase	2	10	5	22
Letter Recognition	26	104	3	10

quality of each of the other models. The result is statistical significant if the calculated probability (p-value is the probability of H_0 being true) is less than α. The results for each data set group are discussed next.

5.7.3 Testing for statistical significance - data group 1

Table 5.7 summarises the results obtained for data group 1 using the applicable parameter values in tables 5.2-5.6 for each of the data sets. The corresponding statistical hypothesis tests between LNNAIS and the remaining models for each of the data sets in group 1 are summarised in table 5.8 (based on the clustering quality, Q_{ratio}). The Mann-Whitney U statistical hypothesis test accepts H_0 that the means are the same at a 0.05 level of significance between LNNAIS and Opt-aiNet and between LNNAIS and CPSO for data set hepta. The remainder of the Mann-Whitney U statistical hypothesis tests showed a significant difference in performance between LNNAIS and the other clustering algorithms. LNNAIS tends to deliver clusters of a higher quality when compared to K-means, CPSO, DWB and Opt-aiNet for data sets iris and hepta. Although SMAIN tends to deliver clusters of a higher quality when compared to LNNAIS for all data sets in group 1, LNNAIS delivers more compact clusters for the iris data set. Also, K-means tends to deliver clusters of a higher quality for data set two-spiral (refer to table 5.7). SMAIN tends to find clusters in the data sets of group 1 with a higher quality, followed by LNNAIS.

5.7.4 Testing for statistical significance - data group 2

The results obtained for data group 2 with the applicable parameter values in tables 5.2-5.6 are summarised in table 5.9. The Mann-Whitney U statistical hypothesis test accepts H_0 that the mean clustering quality, Q_{ratio}, are the same between LNNAIS and DWB for data set chainlink; and rejects H_0 for all other cases (as summarised in table 5.10). Referring to table 5.9, LNNAIS tends to deliver clusters of a higher quality when compared to CPSO, DWB and Opt-aiNet for all data sets in group 2. K-means tends to deliver clusters of a higher quality when compared to LNNAIS for data sets chainlink and target, but of lower quality for data set engytime. SMAIN also tends to deliver clusters of a higher quality for all data sets in group 2, followed by LNNAIS.

5.7.5 Testing for statistical significance - data group 3

The results of the Mann-Whitney U statistical hypothesis test accepts H_0 that the mean clustering quality, Q_{ratio}, are the same between LNNAIS and DWB, and LNNAIS and CPSO for data set

Table 5.7 Descriptive Statistics: Data Group 1

Data set	Algorithm	J_{intra}	J_{inter}	Q_{ratio}
Iris	K-means	0.689 (± 0.073)	3.269 (± 0.201)	0.509 (± 0.268)
	CPSO	0.725 (± 0.089)	2.964 (± 0.201)	0.658 (± 0.354)
	SMAIN	0.766 (± 0.041)	3.705 (± 0.207)	0.295 (± 0.021)
	DWB	0.753 (± 0.152)	3.103 (± 0.282)	0.547 (± 0.304)
	Opt-aiNet	0.887 (± 0.021)	2.977 (± 0.095)	0.882 (± 0.168)
	LNNAIS	0.738 (± 0.054)	3.546 (± 0.309)	0.333 (± 0.048)
Two-spiral	K-means	0.212 (± 0.005)	1.014 (± 0.021)	0.521 (± 0.102)
	CPSO	0.251 (± 0.025)	0.829 (± 0.079)	1.648 (± 0.978)
	SMAIN	0.213 (± 0.004)	1.096 (± 0.013)	0.433 (± 0.015)
	DWB	0.241 (± 0.010)	0.988 (± 0.065)	1.094 (± 0.501)
	Opt-aiNet	0.279 (± 0.027)	0.813 (± 0.105)	2.740 (± 3.020)
	LNNAIS	0.233 (± 0.009)	1.030 (± 0.041)	0.847 (± 0.296)
Hepta	K-means	0.976 (± 0.232)	4.041 (± 0.147)	0.999 (± 0.465)
	CPSO	0.893 (± 0.355)	3.930 (± 0.344)	1.095 (± 1.748)
	SMAIN	0.641 (± 0.001)	4.147 (± 0.005)	0.219 (± 0.001)
	DWB	1.187 (± 0.260)	3.990 (± 0.238)	1.254 (± 0.618)
	Opt-aiNet	1.179 (± 0.462)	3.681 (± 0.499)	1.643 (± 1.353)
	LNNAIS	0.748 (± 0.102)	4.140 (± 0.099)	0.345 (± 0.206)

Table 5.8 Statistical Hypothesis Testing between LNNAIS and Other Models based on Q_{ratio}: Data Group 1 ($\alpha = 0.05$; with continuity correction; unpaired; non-directional)

Data set	Algorithm	z	p	Outcome
Iris	K-means	4.539	< 0.001	Reject H_0
	CPSO	5.958	< 0.001	Reject H_0
	DWB	5.115	< 0.001	Reject H_0
	SMAIN	3.726	< 0.001	Reject H_0
	Opt-aiNet	6.646	< 0.001	Reject H_0
Two-spiral	K-means	5.773	< 0.001	Reject H_0
	CPSO	4.361	< 0.001	Reject H_0
	DWB	2.21	0.027	Reject H_0
	SMAIN	6.646	< 0.001	Reject H_0
	Opt-aiNet	6.246	< 0.001	Reject H_0
Hepta	K-means	3.726	< 0.001	Reject H_0
	CPSO	1.331	0.183	Accept H_0
	DWB	5.892	< 0.001	Reject H_0
	SMAIN	6.646	< 0.001	Reject H_0
	Opt-aiNet	1.804	0.071	Accept H_0

ionosphere, and rejects H_0 for all other cases (as summarised in table 5.11). LNNAIS tends to deliver clusters of a higher quality for all data sets in group 3 when compared to K-means, CPSO and DWB (refer to table 5.12). However, SMAIN and Opt-aiNet tend to deliver clusters of a higher quality for data set ionosphere when compared to cluster quality of LNNAIS. SMAIN also tend to deliver clusters of a higher quality for the data sets in group 3, followed by LNNAIS. LNNAIS does however deliver more compact clusters than SMAIN for the glass data set.

5.7.6 Testing for statistical significance - data group 4

Table 5.13 summarises the results obtained for data group 4. The corresponding statistical hypothesis tests between LNNAIS and the remaining models for each of the data sets in group 4 are summarised in table 5.14. The Mann-Whitney U statistical hypothesis test accepts H_0 that the means are the same between LNNAIS and K-means for data set image segmentation, and between LNNAIS and Opt-aiNet for data set letter recognition. The Mann-Whitney U statistical hypothesis test rejects H_0 for all other cases (as summarised in table 5.14). In most cases LNNAIS tends to deliver clusters of a higher quality except for data set image segmentation and letter recognition where SMAIN tends to deliver clusters of a higher quality (refer to table 5.13).

Table 5.9 Descriptive Statistics: Data Group 2

Data set	Algorithm	J_{intra}	J_{inter}	Q_{ratio}
Engytime	K-means	1.431 (± 0.000)	2.998 (± 0.000)	0.477 (± 0.000)
	CPSO	1.435 (± 0.001)	2.935 (± 0.012)	0.489 (± 0.002)
	SMAIN	2.097 (± 0.103)	5.975 (± 0.670)	0.355 (± 0.039)
	DWB	1.599 (± 0.120)	3.057 (± 0.526)	0.540 (± 0.115)
	Opt-aiNet	1.435 (± 0.001)	2.932 (± 0.025)	0.490 (± 0.004)
	LNNAIS	1.944 (± 0.281)	4.557 (± 1.043)	0.438 (± 0.069)
Chainlink	K-means	0.488 (± 0.006)	1.550 (± 0.049)	0.517 (± 0.031)
	CPSO	0.592 (± 0.053)	1.412 (± 0.150)	1.092 (± 0.667)
	SMAIN	0.487 (± 0.007)	1.643 (± 0.039)	0.471 (± 0.023)
	DWB	0.538 (± 0.025)	1.506 (± 0.074)	0.751 (± 0.320)
	Opt-aiNet	0.646 (± 0.059)	1.363 (± 0.185)	1.352 (± 0.554)
	LNNAIS	0.535 (± 0.021)	1.493 (± 0.116)	0.640 (± 0.118)
Target	K-means	0.544 (± 0.030)	2.393 (± 0.244)	0.337 (± 0.032)
	CPSO	0.749 (± 0.077)	2.340 (± 0.556)	1.133 (± 0.578)
	SMAIN	1.008 (± 0.000)	5.794 (± 0.000)	0.238 (± 0.001)
	DWB	0.649 (± 0.059)	2.058 (± 0.319)	0.752 (± 0.285)
	Opt-aiNet	0.792 (± 0.050)	2.706 (± 0.494)	1.750 (± 1.491)
	LNNAIS	0.804 (± 0.124)	2.985 (± 0.525)	0.559 (± 0.155)

Table 5.10 Statistical Hypothesis Testing between LNNAIS and Other Models based on Q_{ratio}: Data Group 2 ($\alpha = 0.05$; with continuity correction; unpaired; non-directional)

Data set	Algorithm	z	p	Outcome
Engytime	K-means	3.097	0.002	Reject H_0
	CPSO	3.4	< 0.001	Reject H_0
	DWB	3.888	< 0.001	Reject H_0
	SMAIN	4.931	< 0.001	Reject H_0
	Opt-aiNet	3.4	< 0.001	Reject H_0
Chainlink	K-means	4.886	< 0.001	Reject H_0
	CPSO	3.748	< 0.001	Reject H_0
	DWB	0.85	0.395	Accept H_0
	SMAIN	6.32	< 0.001	Reject H_0
	Opt-aiNet	5.759	< 0.001	Reject H_0
Target	K-means	6.513	< 0.001	Reject H_0
	CPSO	4.517	< 0.001	Reject H_0
	DWB	2.964	0.003	Reject H_0
	SMAIN	6.646	< 0.001	Reject H_0
	Opt-aiNet	4.657	< 0.001	Reject H_0

Table 5.11 Statistical Hypothesis Testing between LNNAIS and Other Models based on Q_{ratio}: Data Group 3 ($\alpha = 0.05$; with continuity correction; unpaired; non-directional)

Data set	Algorithm	z	p	Outcome
Ionosphere	K-means	2.24	0.025	Reject H_0
	CPSO	1.833	0.067	Accept H_0
	DWB	1.582	0.114	Accept H_0
	SMAIN	6.646	< 0.001	Reject H_0
	Opt-aiNet	3.837	< 0.001	Reject H_0
Glass	K-means	4.664	< 0.001	Reject H_0
	CPSO	6.513	< 0.001	Reject H_0
	DWB	6.291	< 0.001	Reject H_0
	SMAIN	4.916	< 0.001	Reject H_0
	Opt-aiNet	6.646	< 0.001	Reject H_0

Table 5.12 Descriptive Statistics: Data Group 3

Data set	Algorithm	J_{intra}	J_{inter}	Q_{ratio}
Ionosphere	K-means	2.302 ($\pm$ 0.125)	3.192 ($\pm$ 0.486)	0.728 ($\pm$ 0.045)
	CPSO	2.806 ($\pm$ 0.221)	4.197 ($\pm$ 1.306)	0.778 ($\pm$ 0.387)
	SMAIN	2.767 ($\pm$ 0.000)	6.047 ($\pm$ 0.000)	0.458 ($\pm$ 0.000)
	DWB	2.632 ($\pm$ 0.168)	3.488 ($\pm$ 0.888)	0.799 ($\pm$ 0.195)
	Opt-aiNet	2.781 ($\pm$ 0.068)	4.623 ($\pm$ 1.086)	0.662 ($\pm$ 0.275)
	LNNAIS	2.807 ($\pm$ 0.207)	3.962 ($\pm$ 0.576)	0.725 ($\pm$ 0.127)
Glass	K-means	1.035 ($\pm$ 0.038)	4.557 ($\pm$ 0.464)	0.901 ($\pm$ 0.309)
	CPSO	1.581 ($\pm$ 0.120)	3.017 ($\pm$ 1.121)	1.685 ($\pm$ 0.674)
	SMAIN	1.709 ($\pm$ 0.003)	7.663 ($\pm$ 0.038)	0.381 ($\pm$ 0.007)
	DWB	1.198 ($\pm$ 0.089)	3.716 ($\pm$ 0.899)	1.458 ($\pm$ 0.471)
	Opt-aiNet	1.446 ($\pm$ 0.170)	3.256 ($\pm$ 1.179)	2.188 ($\pm$ 0.701)
	LNNAIS	1.358 ($\pm$ 0.149)	5.367 ($\pm$ 0.423)	0.541 ($\pm$ 0.113)

Table 5.13 Descriptive Statistics: Data Group 4

Data set	Algorithm	J_{intra}	J_{inter}	Q_{ratio}
Image Segmentation	K-means	65.274 (± 0.523)	356.964 (± 32.396)	0.694 (± 0.033)
	CPSO	77.522 (± 7.161)	177.950 (± 24.600)	1.493 (± 0.598)
	SMAIN	126.990 (± 0.283)	787.028 (± 1.906)	0.400 (± 0.001)
	DWB	71.657 (± 3.074)	245.495 (± 133.903)	1.060 (± 0.301)
	Opt-aiNet	74.457 (± 6.321)	174.931 (± 28.219)	1.621 (± 0.990)
	LNNAIS	87.984 (± 9.635)	597.456 (± 116.260)	0.989 (± 1.015)
Spambase	K-means	216.058 (± 0.000)	2003.263 (± 0.000)	0.108 (± 0.000)
	CPSO	301.660 (± 19.617)	136.613 (± 30.941)	2.301 (± 0.452)
	SMAIN	239.369 (± 27.139)	1599.789 (± 831.832)	0.194 (± 0.096)
	DWB	185.926 (± 22.246)	1216.169 (± 1509.055)	0.236 (± 0.120)
	Opt-aiNet	247.833 (± 18.812)	71.578 (± 37.181)	5.586 (± 6.421)
	LNNAIS	432.734 (± 221.003)	6720.659 (± 2691.210)	0.074 (± 0.046)
Letter Recognition	K-means	5.383 (± 0.012)	11.121 (± 0.157)	1.090 (± 0.043)
	CPSO	6.571 (± 0.121)	11.028 (± 0.764)	1.480 (± 0.225)
	SMAIN	7.297 (± 0.238)	17.299 (± 0.455)	0.751 (± 0.029)
	DWB	6.562 (± 0.108)	12.268 (± 0.704)	1.758 (± 0.662)
	Opt-aiNet	6.419 (± 0.108)	11.778 (± 0.630)	1.367 (± 0.179)
	LNNAIS	6.072 (± 0.080)	12.601 (± 0.331)	1.351 (± 0.202)

Table 5.14 Statistical Hypothesis Testing between LNNAIS and Other Models based on Q_{ratio}: Data Group 4 ($\alpha = 0.05$; with continuity correction; unpaired; non-directional)

Data set	Algorithm	z	p	Outcome
Image Segmentation	K-means	1.922	0.055	Accept H_0
	CPSO	5.093	< 0.001	Reject H_0
	DWB	3.6	< 0.001	Reject H_0
	SMAIN	6.646	< 0.001	Reject H_0
	Opt-aiNet	5.064	< 0.001	Reject H_0
Spambase	K-means	3.984	< 0.001	Reject H_0
	CPSO	6.646	< 0.001	Reject H_0
	DWB	5.603	< 0.001	Reject H_0
	SMAIN	5.5	< 0.001	Reject H_0
	Opt-aiNet	6.646	< 0.001	Reject H_0
Letter Recognition	K-means	5.404	< 0.001	Reject H_0
	CPSO	2.144	0.032	Reject H_0
	DWB	3.053	0.002	Reject H_0
	SMAIN	6.646	< 0.001	Reject H_0
	Opt-aiNet	0.288	0.773	Accept H_0

Also, K-means tends to deliver clusters of a higher quality for data set letter recognition.

The experimental results show that, in general, LNNAIS delivers clusters of similar or higher quality than classical data clustering models like K-means and CPSO, and network based AIS models like DWB and Opt-aiNet. Overall, SMAIN tends to deliver clusters of a higher quality for all data sets, followed by LNNAIS. Although SMAIN tends to deliver clusters of a higher quality than LNNAIS, a cursory assessment indicates that SMAIN tends to utilise a larger ALC population than LNNAIS. This might indicate an overfit of the data which results in superior clustering quality of SMAIN. A disadvantage of SMAIN when compared to LNNAIS is that SMAIN follows a hybrid approach to determine the number of ALC networks (clusters) and is therefore computationally more expensive than LNNAIS. Furthermore, LNNAIS has less user specified parameters. The next section compares and discusses the ALC population sizes of SMAIN, DWB and LNNAIS to elaborate on the cursory assessment of overfitting the data. This is then followed by a sensitivity analysis of the LNNAIS parameters on the clustering quality of the model.

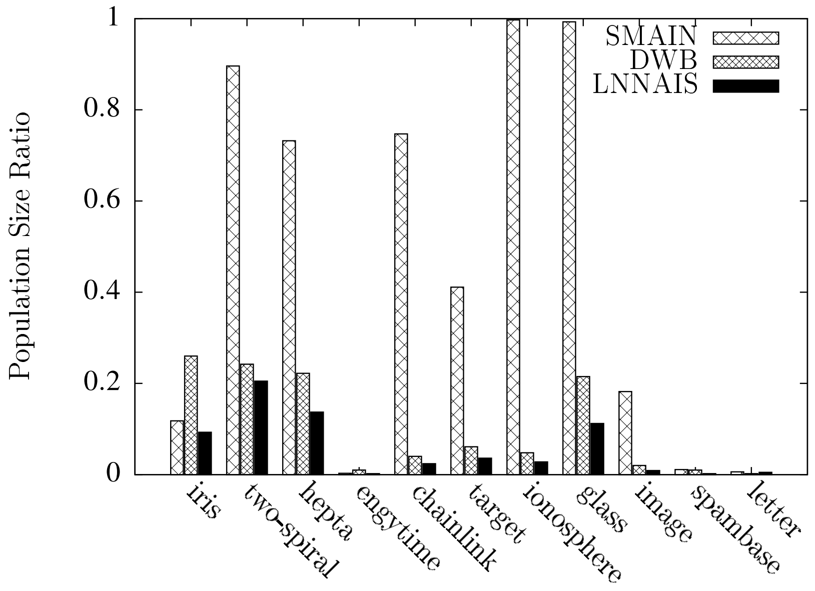


Figure 5.4 ALC Population Size Ratios of SMAIN, DWB and LNNAIS

5.7.7 ALC Population Size - Overfitting the Data

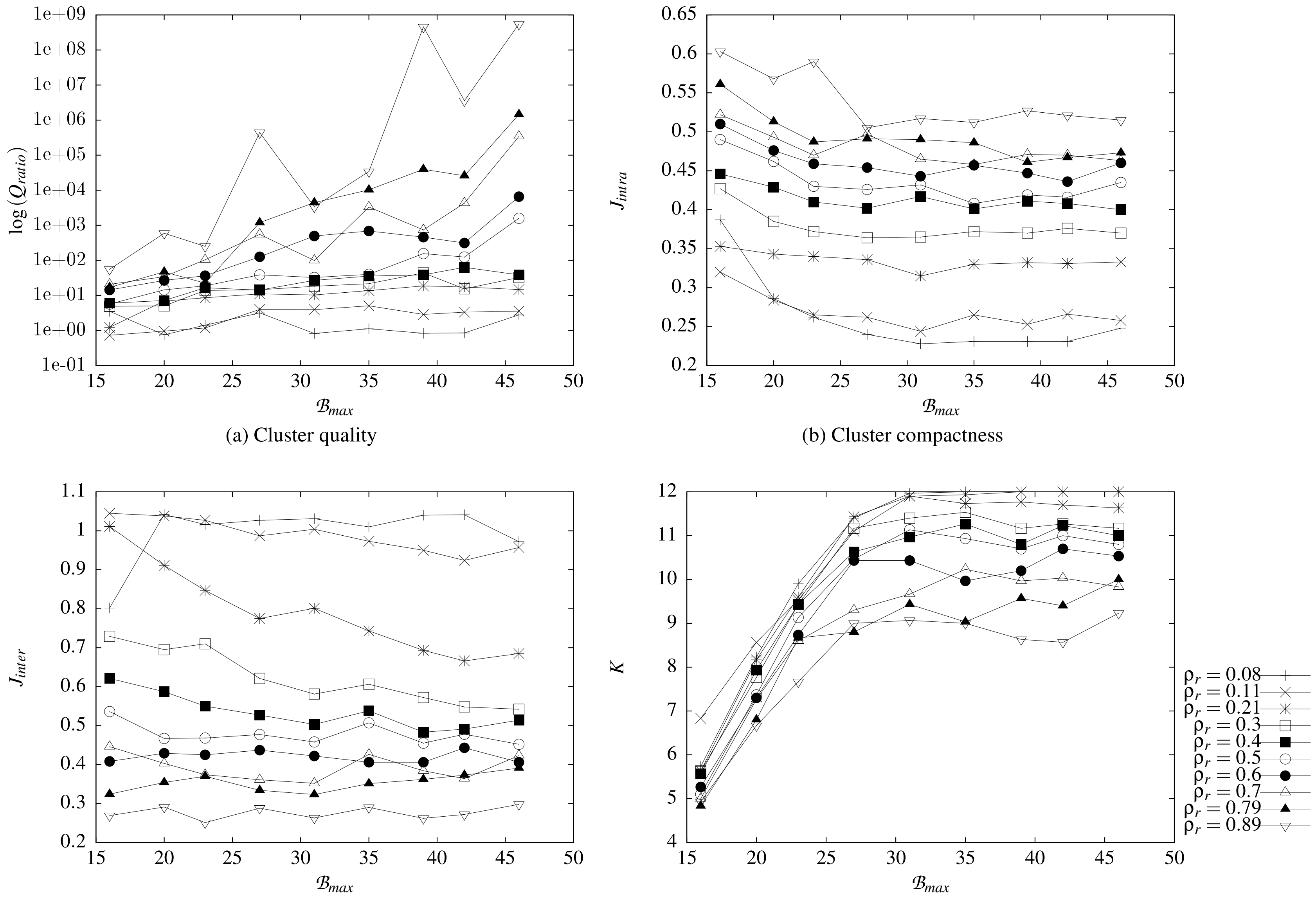
This section investigates the ALC population sizes between SMAIN, DWB and LNNAIS to indicate potential overfit of the data. Overfitting of the data could result in superior clustering quality of a specific model when compared to other models which utilise a smaller ALC population size. Figure 5.4 illustrates a histogram of the ALC population size of SMAIN, DWB and LNNAIS to cluster the data sets. The size of the ALC population is expressed as a ratio of the applicable data set size. Therefore, an ALC population size ratio closer to 1.0 indicates a higher level of overfit of the applicable data set. The figure illustrates that LNNAIS has a population size ratio of less than 0.2 for all of the data sets. On the contrary, SMAIN has a population size ratio of more than 0.4 for six of the data sets (two-spiral, hepta, chainlink, target, ionosphere and glass). For data sets glass and ionosphere, the ALC population size of SMAIN is almost equal to the size of the data sets (ratio close to 1.0). In general, SMAIN utilises a larger ALC population to cluster the data than DWB and LNNAIS. This not only explains the superior clustering quality of SMAIN in the previous section but also a drawback of SMAIN that tends to overfit the data. Compared to SMAIN in view of these findings, LNNAIS delivers clusters of high quality without overfitting the data.

5.7.8 Influence of LNNAIS parameters

This section investigates the influence of the LNNAIS parameters on the clustering quality of the model with reference to Q_{ratio}, J_{intra}, J_{inter} and the number of obtained clusters K. These parameters are the maximum population size, $\mathcal{B}_{max}$, the neighbourhood size, ρ, and the clonal level threshold, ε_{clone}. Compared to the network based AIS models which are used in this chapter, LNNAIS has significantly less parameters. The clustering results of a representative data set were selected from each of the defined data groups for the discussion. All of the other clustering results of the remaining data sets within the same data group, followed similar trends unless stated otherwise. The identified data sets include two-spiral from group 1, chainlink from group 2, glass from group 3 and image segmentation from group 4. The LNNAIS model has been executed with population sizes of 10 to 50 ALCs, clonal level threshold values of 6 to 27 and neighbourhood sizes which are calculated as a ratio of the population size. Neighbourhood size ratios from 0.05 to 0.9 were used to calculate the neighbourhood size ρ using $\rho = \rho_r \times \mathcal{B}_{max}$ (ρ_r is the neighbourhood size ratio). In cases where a parameter was kept constant, the parameter was set to the value as listed in table 5.6 for each of the applicable data sets.

Population Size: Figures 5.5 to 5.8 show the effect of different ALC population sizes, $\mathcal{B}_{max}$, at different neighbourhood size ratios, ρ_r, and a constant clonal level threshold, ε_{clone}. These figures show that for small neighbourhood sizes an increase in the ALC population size has a less significant influence on the clustering quality, Q_{ratio}, when compared to larger neighbourhood sizes. The cluster compactness and separation do however tend to decrease at low neighbourhood sizes with an increase in the ALC population size (increasing J_{intra} and decreasing J_{inter}). Furthermore, figures 5.5 to 5.8 also show that no significant improvement is achieved for all the different neighbourhood sizes in the number of obtained clusters for ALC population sizes larger than a specific optimal value (which is problem dependant). This can also be observed in figures 5.13 to 5.16. Figures 5.13 to 5.16 show that an increase in the ALC population size increases the cluster compactness and separation (decreasing J_{intra} and increasing J_{inter}) for different clonal level threshold values with a low constant neighbourhood size. Therefore, an increase in the ALC population size increases diversity which obtains the required number of clusters and improves the clustering quality.

Neighbourhood Size: Figures 5.9 to 5.12 show the effect of different neighbourhood size ratios, ρ_r, at different clonal level threshold values, ε_{clone}, and a constant ALC population size, $\mathcal{B}_{max}$. An increase in the neighbourhood size decreases the cluster compactness and separation



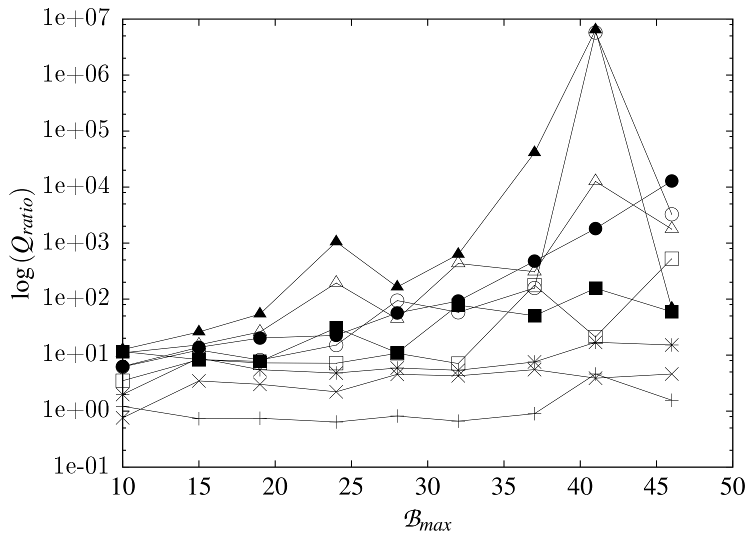
(a) Cluster quality

(b) Cluster compactness

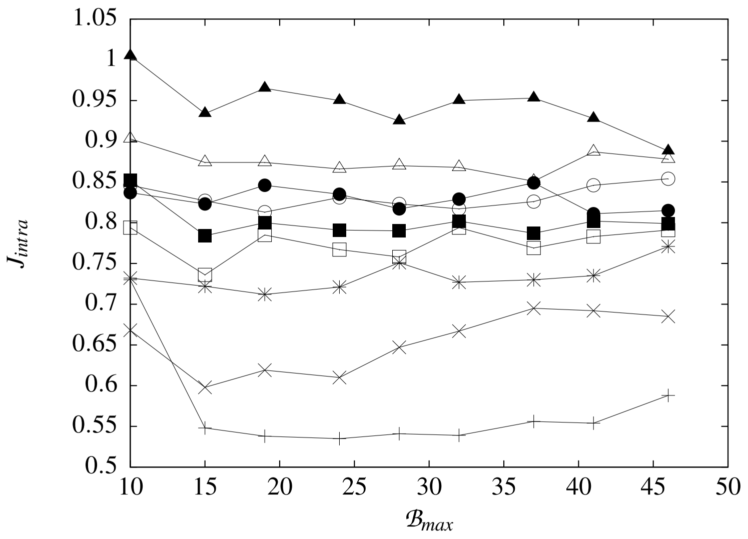
(c) Cluster separation

(d) Number of obtained clusters

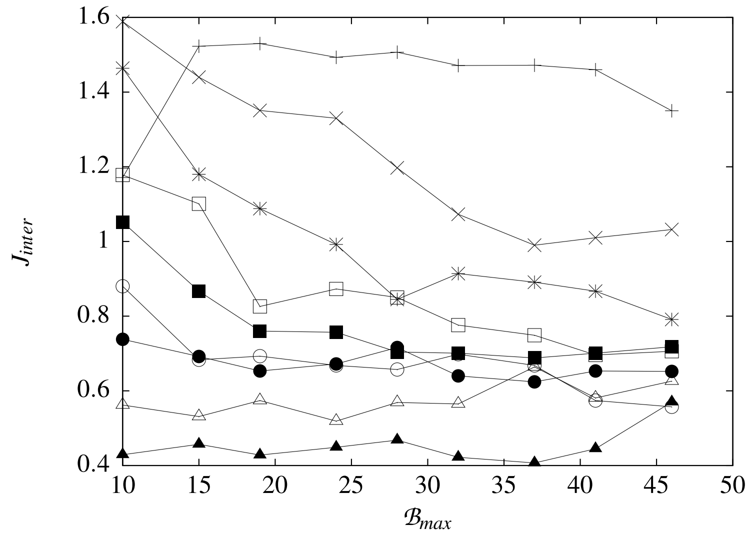
Figure 5.5 Two-spiral data set ($\varepsilon_{clone} = 6$): Effect of the ALC population size with a constant clonal level threshold



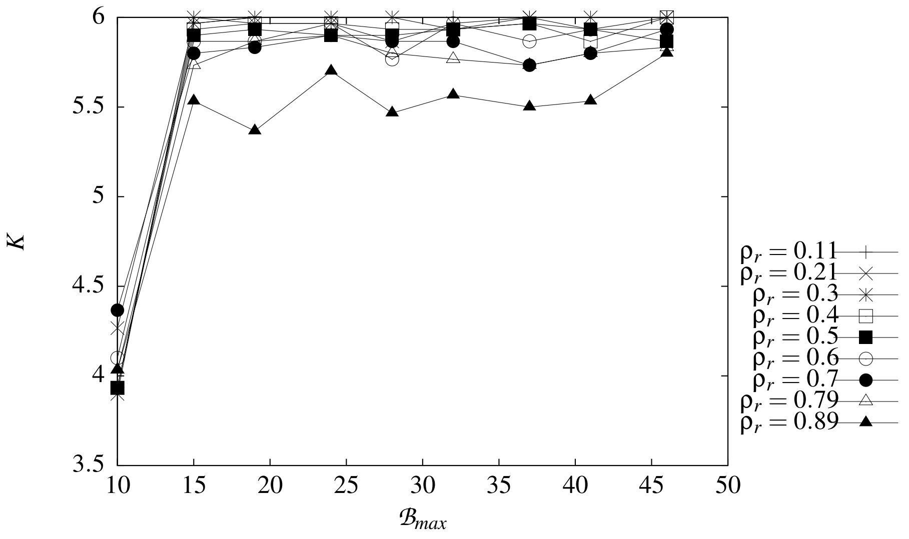
(a) Cluster quality



(b) Cluster compactness

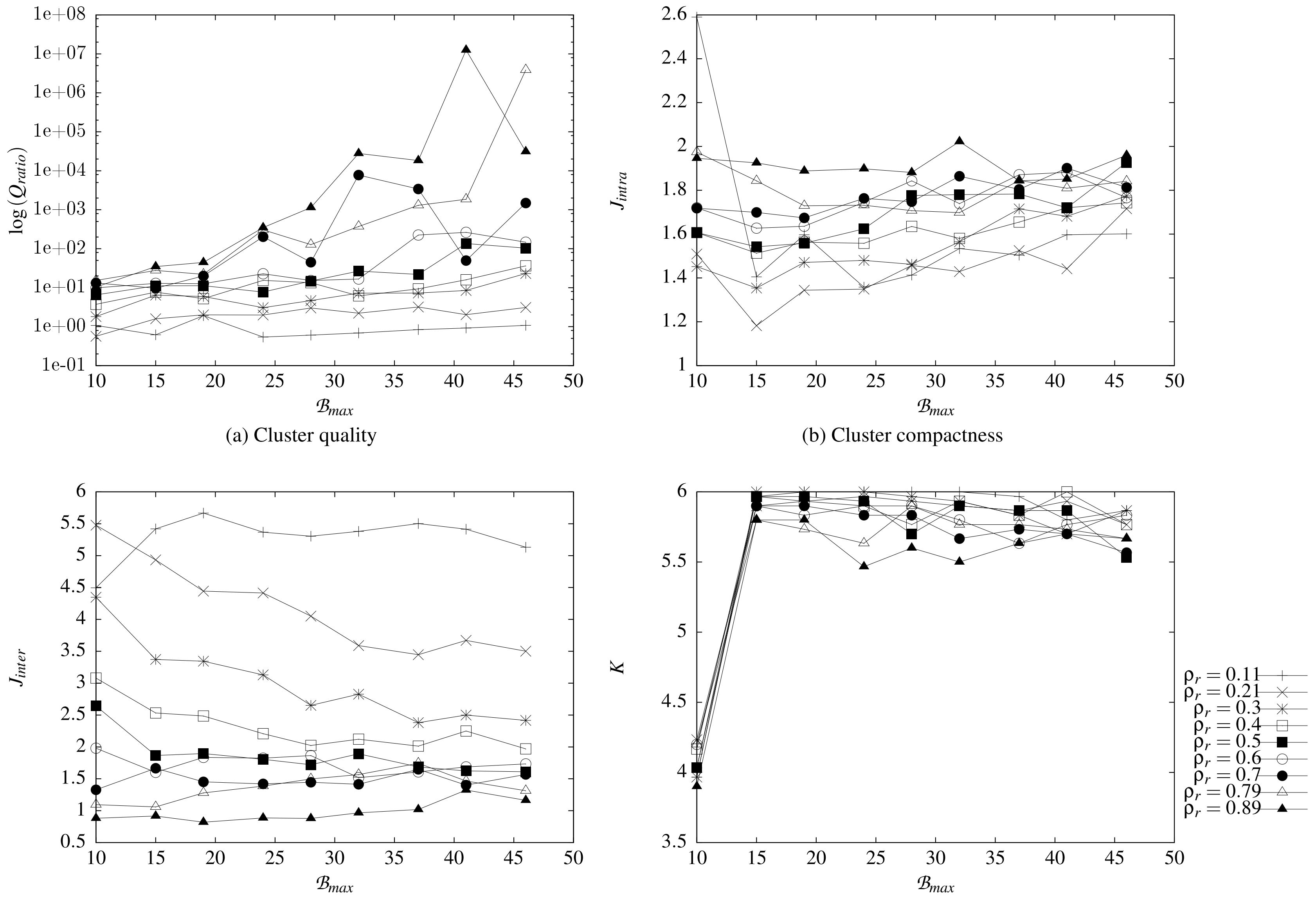


(c) Cluster separation



(d) Number of obtained clusters

Figure 5.6 Chainlink data set ($\varepsilon_{clone} = 8$): Effect of the ALC population size with a constant clonal level threshold



(a) Cluster quality

(b) Cluster compactness

(c) Cluster separation

(d) Number of obtained clusters

Figure 5.7 Glass data set ($\varepsilon_{clone} = 8$): Effect of the ALC population size with a constant clonal level threshold

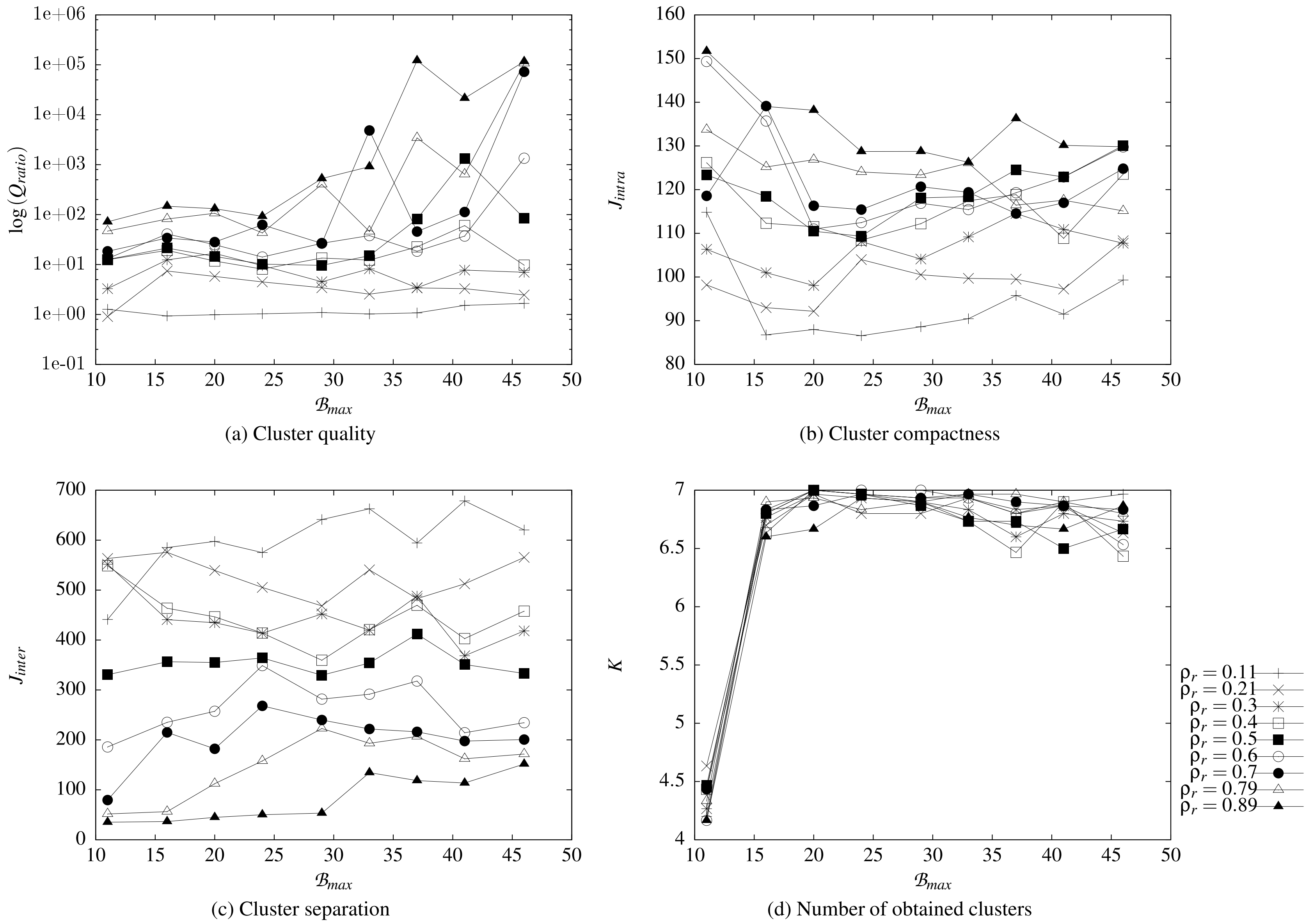


Figure 5.8 Image Segmentation data set ($\varepsilon_{clone} = 27$): Effect of the ALC population size with a constant clonal level threshold

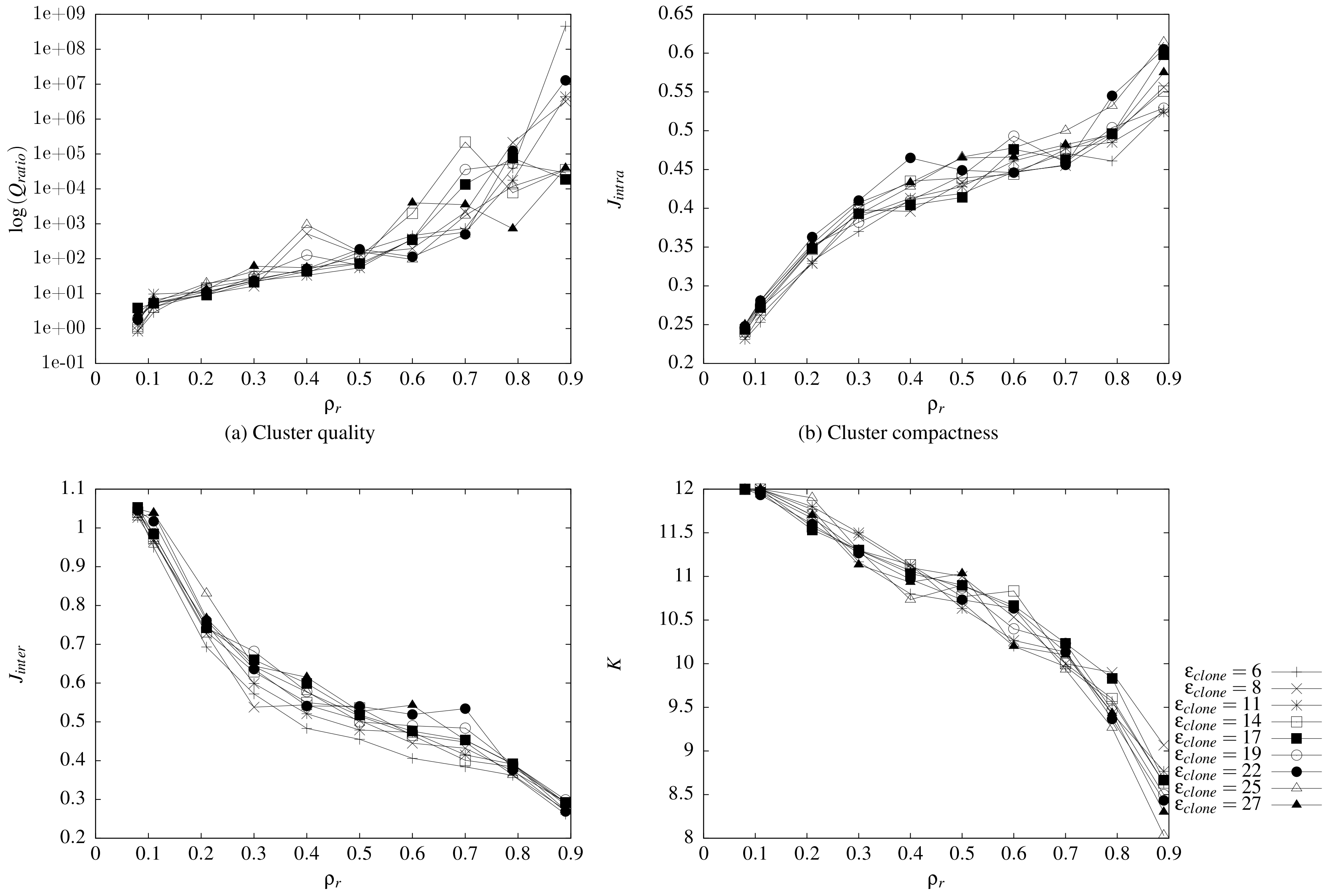
for all of the different clonal level threshold values, resulting in clusters of a lower quality (increasing Q_{ratio} and J_{intra} with a decreasing J_{inter}). This effect is also shown in figures 5.5 to 5.8 where an increase in the neighbourhood size ratio decreases the cluster compactness (increases J_{intra}) and decreases the cluster separation (decreases J_{inter}) for all values of $\mathcal{B}_{max}$. From these observations it can be concluded that small values of ρ_r deliver more compact and more separated clusters (lower J_{intra}, higher J_{inter}) and therefore clusters of higher quality (lower Q_{ratio}) when compared to higher values of ρ_r. From the above mentioned figures, lower neighbourhood sizes also tend to obtain the required number of clusters.

Clonal Level Threshold: Figures 5.13 to 5.16 show the effect of different clonal level threshold values, ε_{clone}, at different ALC population sizes, $\mathcal{B}_{max}$, and a constant neighbourhood size, ρ. An increase in the clonal level threshold has no significant improvement in the number of obtained clusters at different ALC population sizes (as illustrated in figures 5.13 to 5.16) and also not at different neighbourhood sizes (as illustrated in figures 5.9 to 5.12). Furthermore, the different clonal level threshold values follow similar trends with reference to the quality, compactness and separation of the clusters when the neighbourhood size increases (as illustrated in figures 5.9 to 5.12 and explained in the previous paragraph). In the case of the chainlink and image segmentation data sets, increasing the clonal level threshold also results in less compact clusters at different ALC population sizes (as illustrated in figures 5.14 and 5.16), whereas there is no significant change in the compactness of the clusters for the two-spiral and glass data sets (as illustrated in figures 5.13 and 5.15). Therefore, the clonal level threshold influences the cluster compactness and is problem specific.

In summary, the clustering performance of LNNAIS is sensitive to the values of the ALC population size and neighbourhood size. The ALC population size is problem specific and in general low neighbourhood size values deliver clusters of higher quality. The clustering performance of LNNAIS is generally insensitive to the value of the clonal level threshold.

5.8 Conclusion

A new network based AIS model (LNNAIS) was proposed for data clustering. LNNAIS utilises a different network topology, which is an index based ALC neighbourhood topology to determine the network connectivity between ALCs. The clustering performance of the LNNAIS model was compared against classical clustering algorithms (K-means clustering and CPSO) and existing



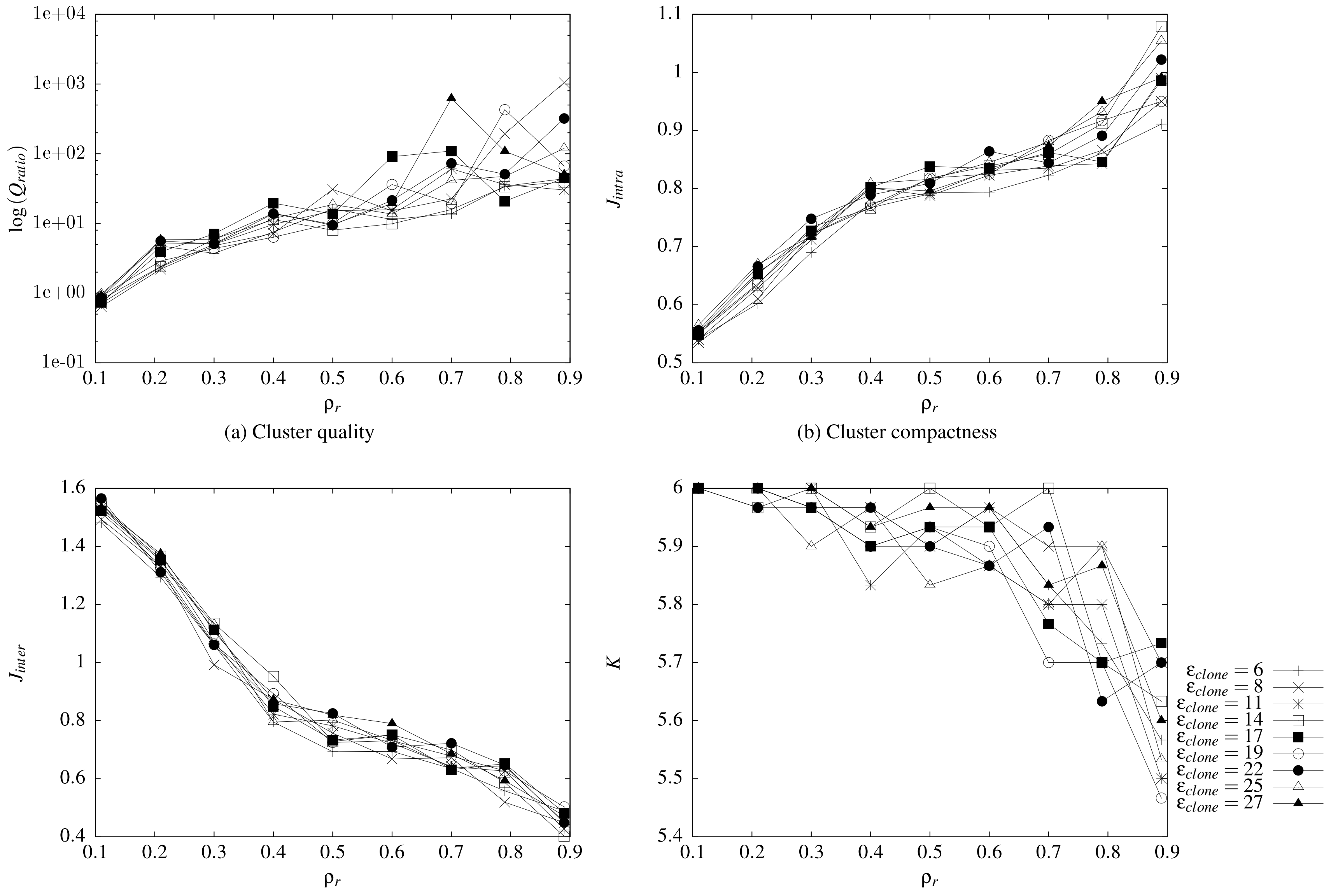
(a) Cluster quality

(b) Cluster compactness

(c) Cluster separation

(d) Number of obtained clusters

Figure 5.9 Two-spiral data set ($\mathcal{B}_{max} = 39$): Effect of the neighbourhood size with a constant ALC population size



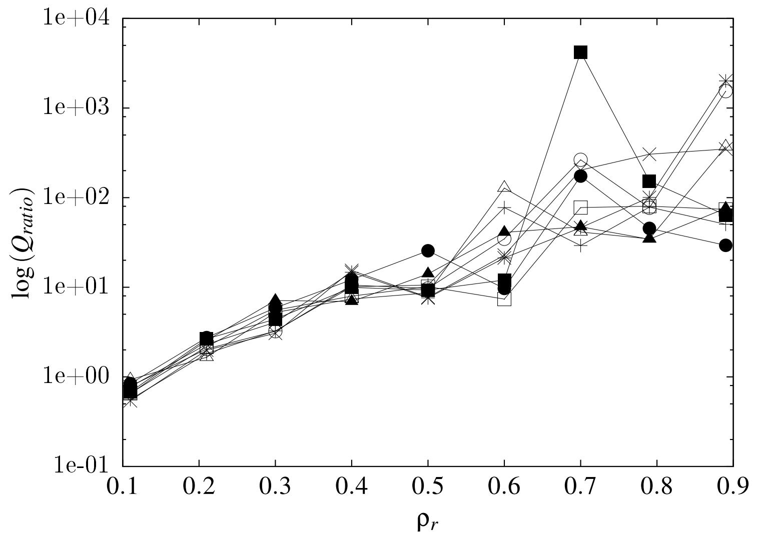
(a) Cluster quality

(b) Cluster compactness

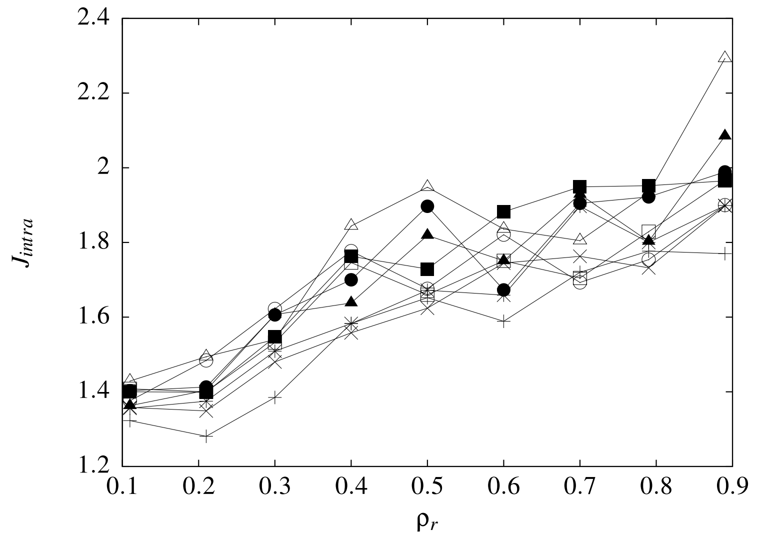
(c) Cluster separation

(d) Number of obtained clusters

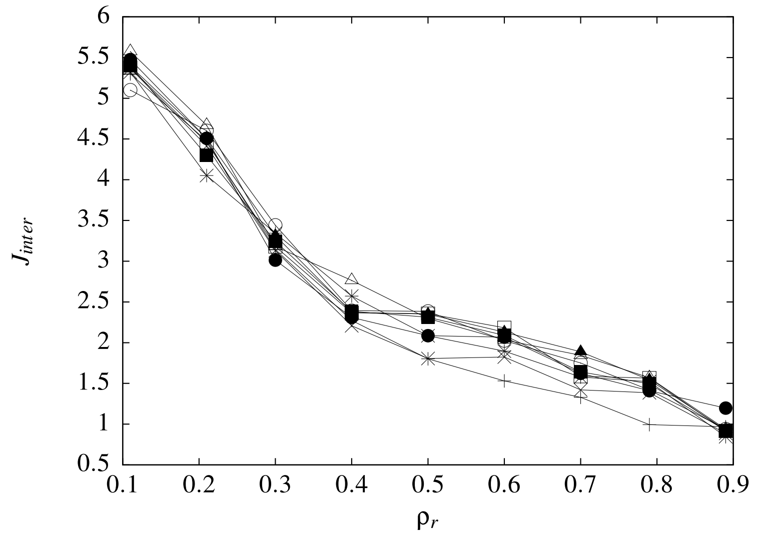
Figure 5.10 Chainlink data set ($\mathcal{B}_{max} = 24$): Effect of the neighbourhood size with a constant ALC population size



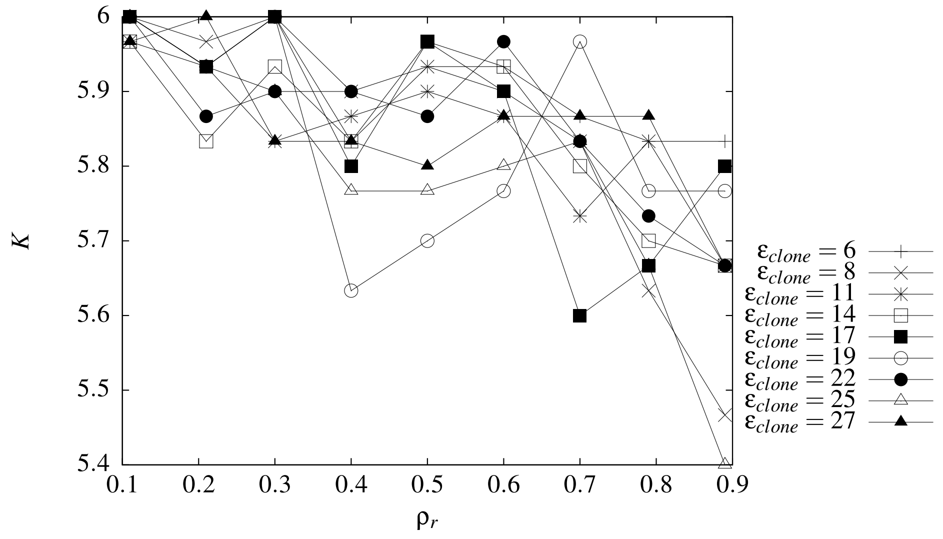
(a) Cluster quality



(b) Cluster compactness



(c) Cluster separation



(d) Number of obtained clusters

Figure 5.11 Glass data set ($\mathcal{B}_{max} = 24$): Effect of the neighbourhood size with a constant ALC population size

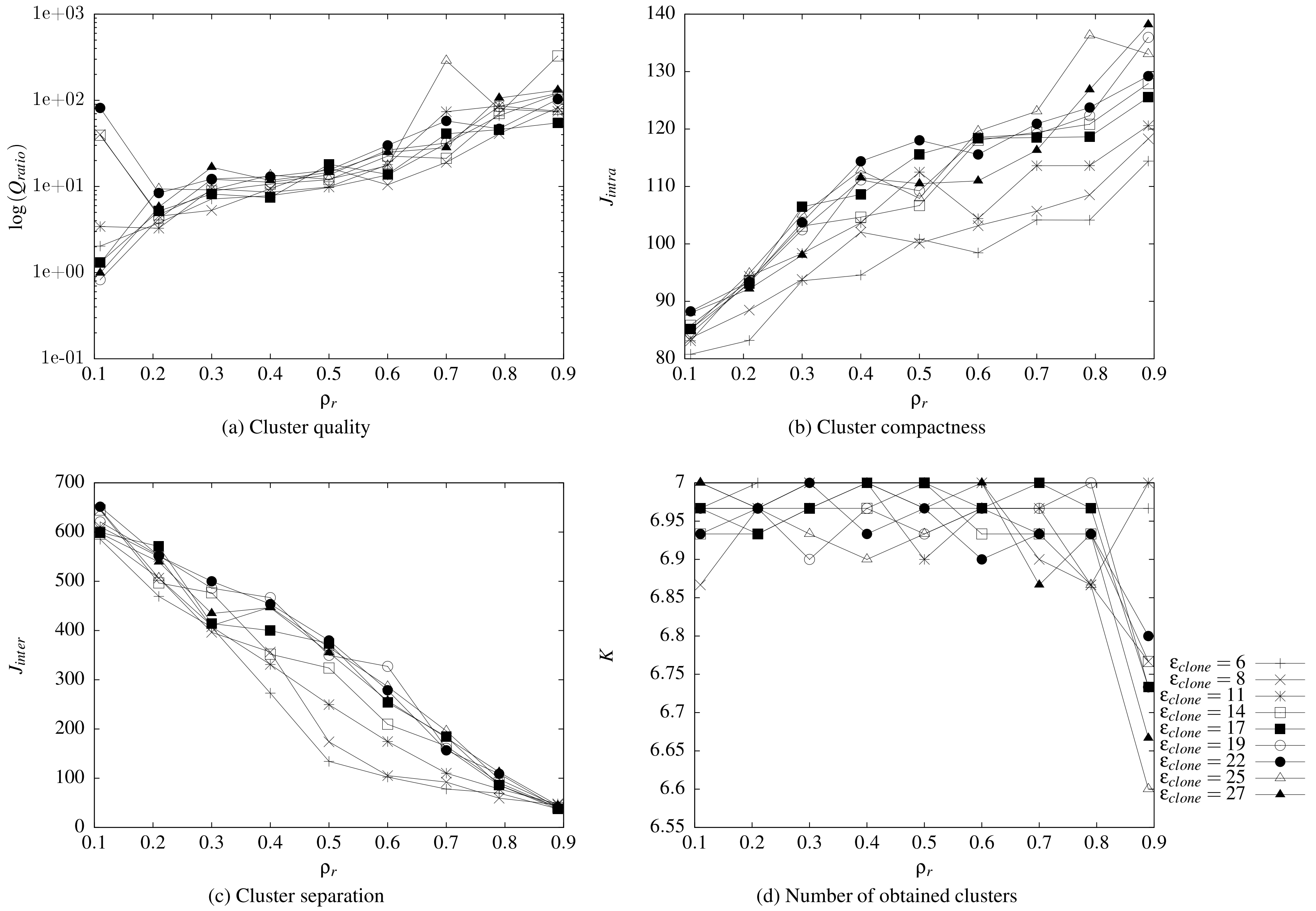
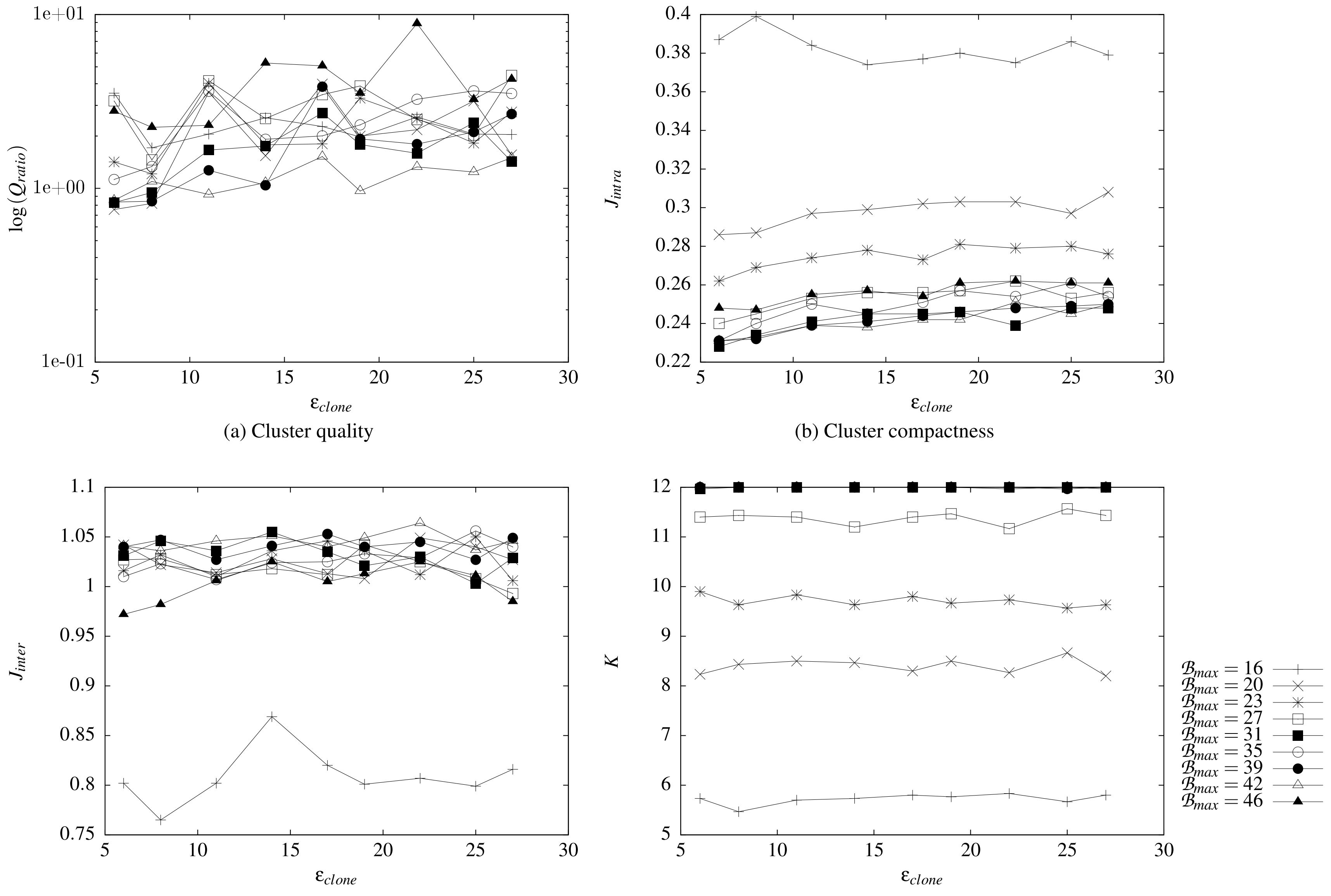


Figure 5.12 Image Segmentation data set ($\mathcal{B}_{max} = 20$): Effect of the neighbourhood size with a constant ALC population size



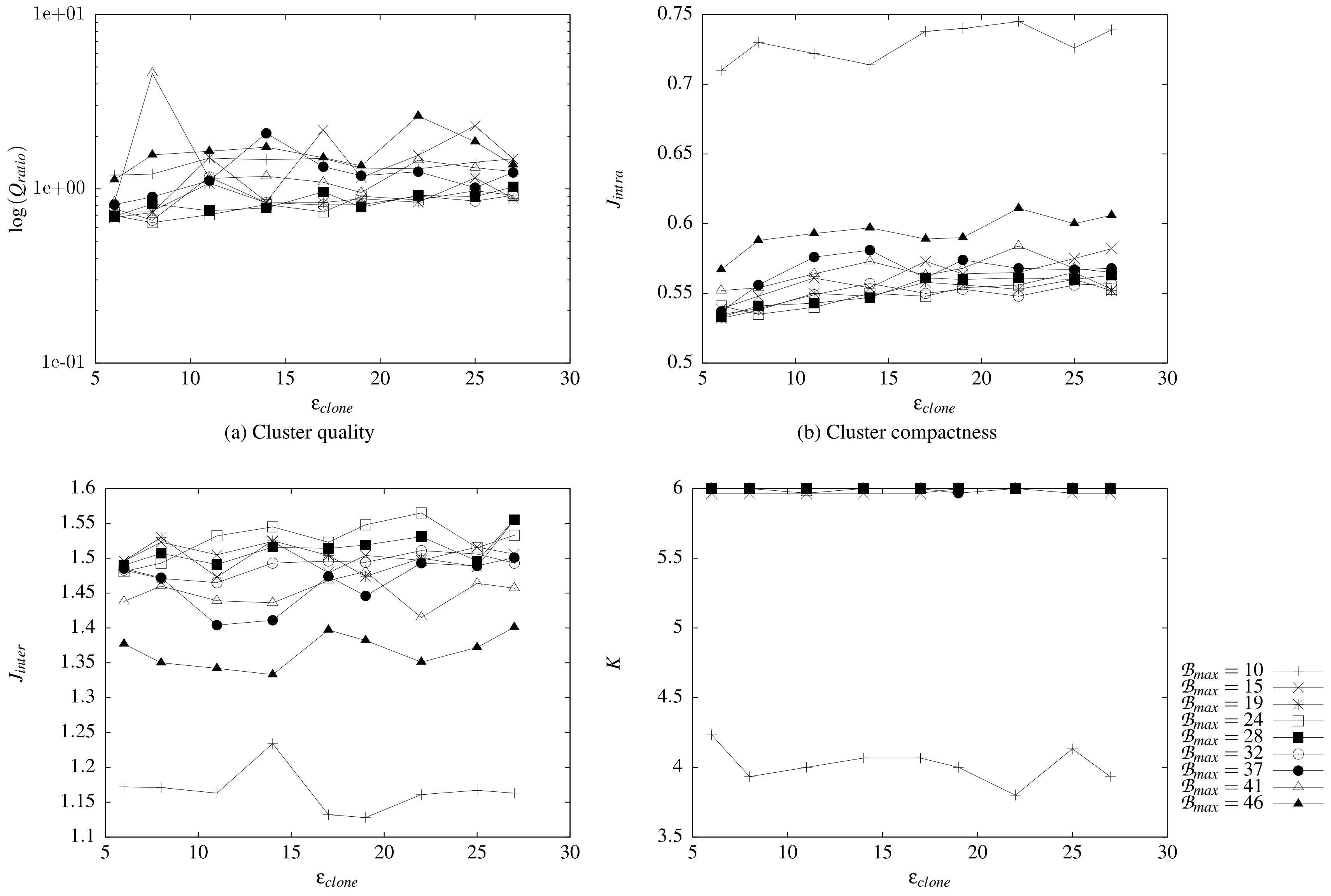
(a) Cluster quality

(b) Cluster compactness

(c) Cluster separation

(d) Number of obtained clusters

Figure 5.13 Two-spiral data set ($\rho = 3$): Effect of the clonal level threshold with a constant neighbourhood size



(a) Cluster quality

(b) Cluster compactness

(c) Cluster separation

(d) Number of obtained clusters

Figure 5.14 Chainlink data set ($\rho = 3$): Effect of the clonal level threshold with a constant neighbourhood size

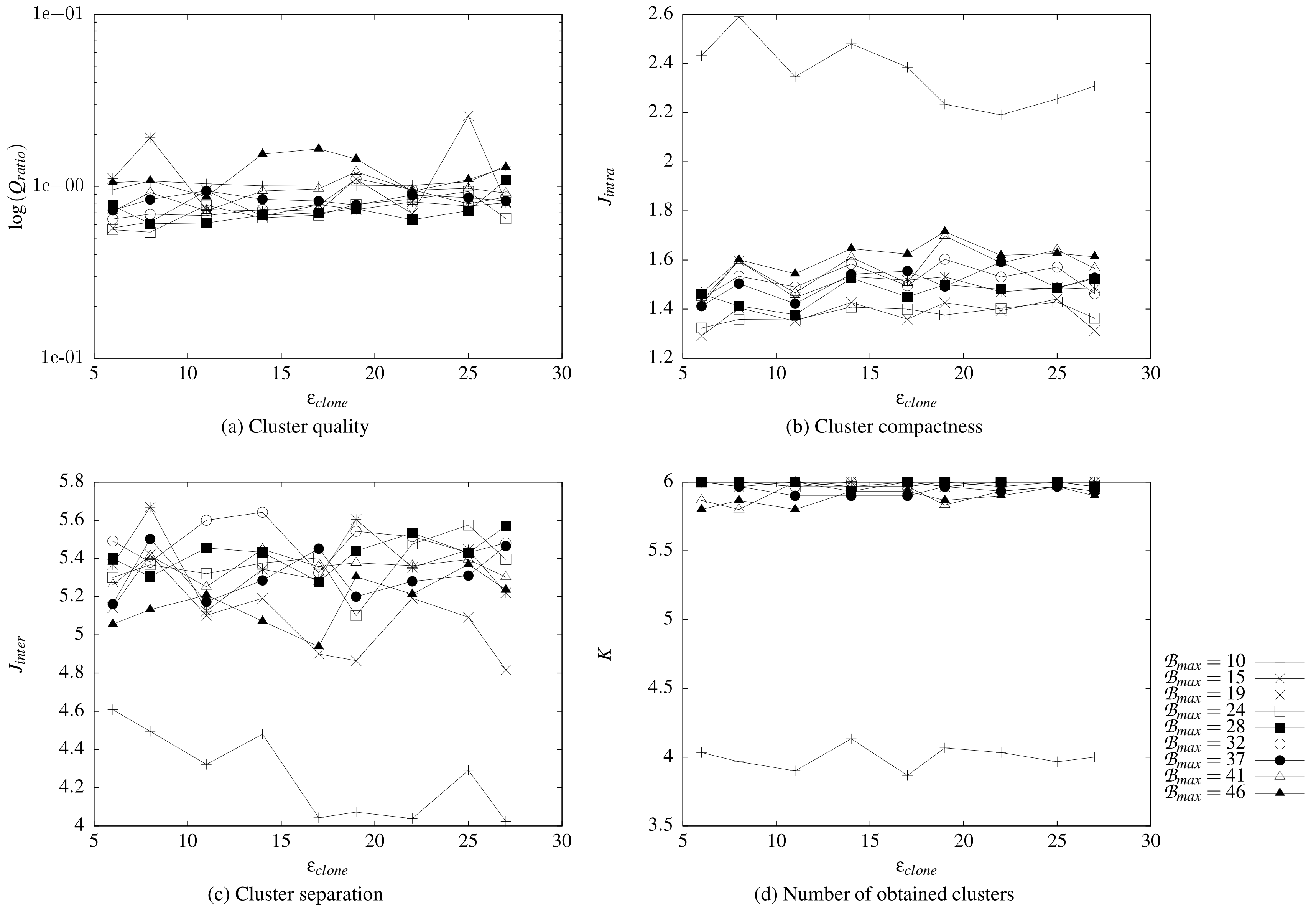
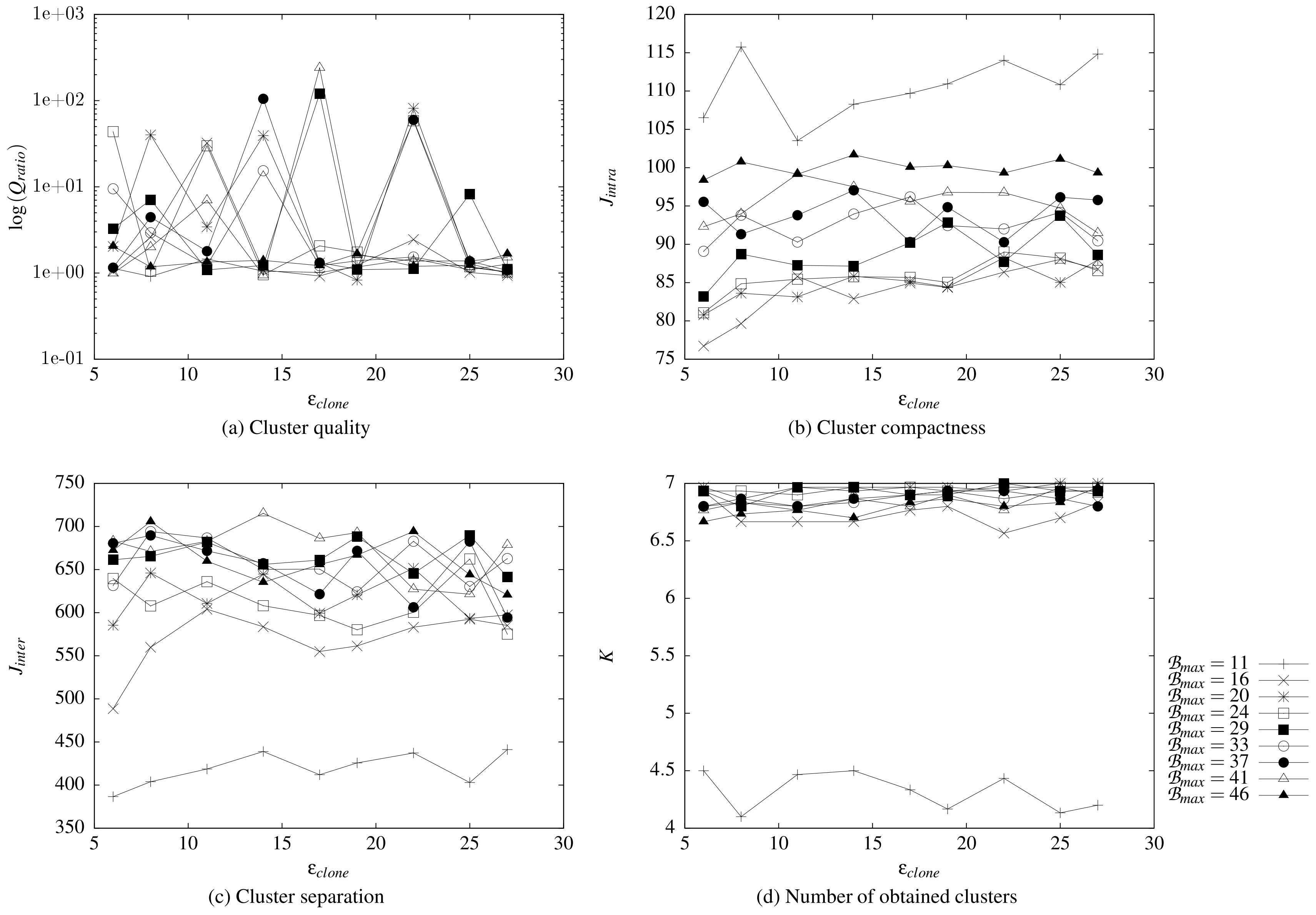


Figure 5.15 Glass data set ($\rho = 3$): Effect of the clonal level threshold with a constant neighbourhood size



(a) Cluster quality (b) Cluster compactness

(c) Cluster separation (d) Number of obtained clusters

Figure 5.16 Image Segmentation data set ($\rho = 2$): Effect of the clonal level threshold with a constant neighbourhood size

network based AIS models (SMAIN, DWB and Opt-aiNet). In most cases, LNNAIS produced better or similar results with reference to the quality, compactness and separation of the clusters. Although SMAIN tends to deliver clusters of a higher quality than LNNAIS, further investigation showed that SMAIN tend to utilise a larger ALC population than LNNAIS.

A sensitivity analysis was done on the LNNAIS parameters to investigate the effect of the parameters on the clustering quality. An increase in the ALC population size increases diversity which obtains the required number of clusters and improves the clustering quality. Smaller neighbourhood sizes deliver more compact and more separated clusters when compared to larger neighbourhood sizes, and tend to obtain the required number of clusters. Therefore small neighbourhood sizes deliver clusters of a higher quality. The clonal level threshold influences the compactness of the clusters and is problem specific.

Although existing network based AIS models and LNNAIS do not require any user specified parameter of the number of required clusters to cluster the data, the techniques used by these models to determine the number of ALC networks do, however. Therefore, the following chapter investigates and proposes two alternative techniques that can be used with LNNAIS to dynamically determine the number of clusters in a data set.

Chapter 6

Dynamically Determining the Number of Clusters Found by a Local Network Neighbourhood Artificial Immune System

A challenge in data clustering is to determine the optimal number of clusters in a data set. Section 2.4 discussed a number of approaches to validate and determine the number of clusters in a data set. These approaches include validation of the formed clusters by visual inspection and/or multiple execution of the clustering algorithm, each time with a different number of clusters and validating the clustered data set with a cluster validity index. The former visual approach becomes infeasible for multidimensional problems where the number of dimensions is greater than three and even though the latter multiple execution approach is familiar in the field, it is computationally expensive and time consuming. Therefore a clustering technique or model which can dynamically determine the number of clusters in a data set and which is computationally inexpensive will have an added advantage.

Although most of the existing network based artificial immune models do not require any user specified parameter of the number of required clusters to cluster the data, these models do have a drawback in the techniques used to determine the number of clusters. These techniques and their drawbacks were discussed in section 5.5.6. All of the techniques share a mutual drawback which is the user specified parameter of the number of required clusters.

This chapter discusses some of the existing data clustering methods to dynamically determine the number of clusters in a data set. Two techniques are then proposed which can be used with the local network neighbourhood artificial immune model to dynamically determine the number of clusters in a data set. The first technique utilises cluster validity indices and is similar to the multiple execution approach, though computationally less expensive. The second technique is based on sequential deviation outlier detection, which was discussed in section 2.6. The end result of both techniques is an enhanced LNNAIS model that can dynamically determine the number of clusters in a data set.

Experimental results of K-means clustering using the multiple execution technique are compared with the results of the proposed LNNAIS techniques.

6.1 Dynamic Data Clustering Methods

Dynamically determining the optimal number of clusters in a data set is a challenging task, since *a priori* knowledge of the data is required and not always available. As discussed in section 2.4, cluster validity indices can be used with a multiple execution of the clustering algorithm to dynamically determine the number of clusters. A disadvantage of the multiple execution approach is that the technique is computationally expensive and time consuming. Other techniques and clustering models have also been proposed in the literature and are discussed next.

Ball and Hall [10] proposed the Iterative Self-Organising Data Analysis Technique (ISODATA) to dynamically determine the number of clusters in a data set. As with K-means clustering, ISODATA iteratively assigns patterns to the closest centroids. Different to K-means clustering, ISODATA utilises two user-specified thresholds to respectively merge two clusters (if the distance between their centroids is below the first threshold) and also split a cluster into two clusters (based on the second threshold). Even though ISODATA has an advantage above K-means clustering to dynamically determine the number of clusters in the data set, ISODATA has two additional user parameters (merging and splitting thresholds) which have an effect on the number of clusters determined. A similar model to ISODATA is the Dynamic Optimal Cluster-seek (DYNOC) which was proposed by Tou [172]. DYNOC also follows an iterative approach with splitting and merging of clusters but at the same time maximises the ratio of the minimum inter-clustering to the maximum intra-clustering distance. DYNOC also requires a user specified parameter which determines the splitting of a cluster. SYNERACT was proposed by Huang [87] as an alternative to ISODATA. SYNERACT uses a hyperplane to split a cluster into smaller clusters for which the centroids need to be calculated. Similar to ISODATA and DYNOC, an iterative approach is followed to assign patterns to available clusters. Even though SYNERACT is faster than ISODATA and does not require the initial location of centroids or the number of clusters to be specified, SYNERACT does require values for two parameters which have an effect on the splitting of a cluster.

Veenman proposed a partitional clustering model which minimises a cluster validity index in order to dynamically determine the number of clusters in a data set [175]. The initial number of clusters is equal to the number of patterns in the data set. An iterative approach is followed to determine the splitting and merging of clusters. In each iteration, tests which are based on the minimisation of the cluster validity index determine the splitting or merging of clusters. The proposed algorithm has similar drawbacks as the multiple execution approaches, namely that the model is computationally expensive and has user parameters for the cluster validity index which influences the clustering results.

Another K-means based model was proposed by Pelleg and Moore [128] and uses model selection. The model is called X-means and initially start with a single cluster, $K = 1$ (which is the minimum number of clusters in any data set). The first step is then to apply K-means clustering on the K clusters which are then split in a second step according to a Bayesian Information Criterion (BIC) [106]. If the BIC is improved with the splitting of the clusters, the newly formed clusters are accepted, otherwise it is rejected. These steps are repeated until a user specified upper bound on K is reached. X-means clustering dynamically determines the number of clusters in the data set as the value of K which has the best BIC value. X-means also has a drawback of a user specified parameter for the upper bound on K. Hamerly and Elkan proposed a similar model as X-means clustering, called G-means clustering [72]. G-means also starts with a small value of K but only splits clusters which data do not have a Gaussian distribution. This is also a drawback of G-means clustering, since it is assumed that the data has spherical and/or elliptical clusters [72].

There are also other models proposed in the literature which is either based on K-means clustering or utilises K-means with similar approaches of splitting and merging clusters. These models are *Snob* [176] and Modified Linde-Buzo-Gray (MLBG) [154]. All of the discussed models suffer from either user parameters which influence the clustering results or can only cluster data sets with specific characteristics.

The following section proposes two techniques which can be used with the local network neighbourhood artificial immune model to dynamically determine the number of clusters in a data set.

6.2 Dynamic Clustering Techniques for LNNAIS

This section proposes two alternative techniques which can be used by LNNAIS to dynamically determine the number of clusters in a data set. Both of these techniques have advantages and drawbacks which are also discussed. This section first recapitulates the technique used by LNNAIS to determine a user specified number of clusters (as discussed in section 5.5.6).

Different to other network based AIS models, LNNAIS need not to follow a hybrid approach nor a proximity matrix of network affinities in order to determine the formed ALC networks in the ALC population. This is due to the index based neighbourhood topology utilised by LNNAIS. An index based neighbourhood results in the formation of a ring-like network topology as illustrated in figure 5.3. The required number of ALC networks (or rather clusters), K, can be determined by sorting the network affinities in descending order and selecting the first K network affinities in the sorted set. The K selected network affinities determine the boundaries of the ALC networks.

Figure 5.3 illustrates this technique where $K = 3$. Separate ALC networks are formed by pruning the edges of the K selected boundaries (illustrated as dotted lines in figure 5.3). The centroid of each of the formed ALC networks (illustrated as clouds in figure 5.3) is calculated using equation (2.18). An alternative approach to sorting the network affinities is to plot the network affinities against the numbered edges (as illustrated in figure 6.1). The K edges in the graph with the lowest plotted network affinity (highest Euclidean distance) are then selected as the boundaries of the ALC networks.

Iterative Pruning Technique (IPT): Instead of specifying K, the above pruning technique is done with an iterative value of K. First K is set to 2 where only the top two boundaries are selected for pruning (top two network affinities in the sorted set of network affinities). The quality of the clusters is then measured with a cluster validity index of choice. The same procedure is followed for $K = \{3,4,5,\ldots,\mathcal{B}_{max}\}$, measuring the quality with a cluster validity index for each value of K. The value of K with the highest (or lowest, depending on the validity index used)

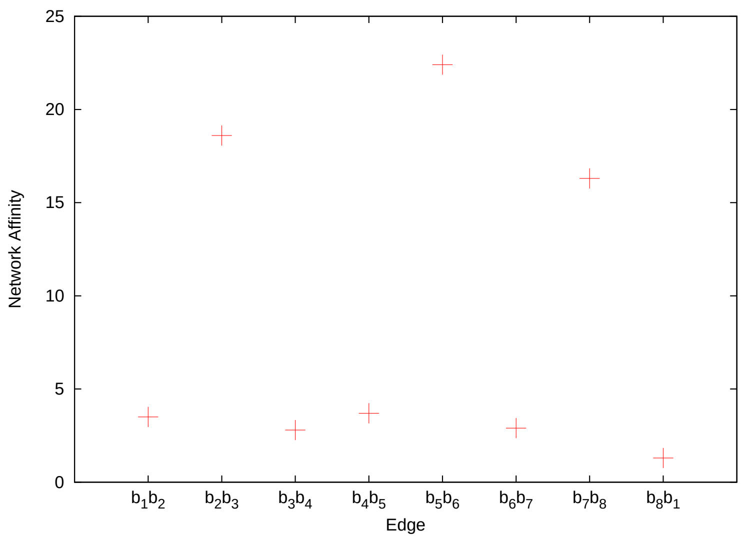


Figure 6.1 Network Affinity Plot

cluster validity index is then selected as the optimal number of clusters. It is also possible to set a minimum and maximum for K, but this can also be seen as a drawback since two parameters need to be specified. If no minimum/maximum is specified it could also be a time consuming task (to a lesser extent when compared to the multiple execution technique) to iterate through all values of K, especially with large values of $\mathcal{B}_{max}$. Whether K is bounded by a minimum/maximum or not, an advantage of the Iterative Pruning Technique to dynamically determine the number of clusters is that the LNNAIS model needs not to be executed for each value of K as in the case of the multiple execution technique. Therefore the Iterative Pruning Technique is computationally less expensive.

Sequential Deviation Outlier Technique (SDOT): Section 2.6 defined outliers and explained three different approaches for outlier detection. One of these approaches is the sequential exception technique which forms part of the deviation based techniques for outlier detection. The reader is referred to section 2.6 for a refresher on the sequential exception technique. As illustrated in figure 6.1, the network affinities which form clear boundaries between the ALC networks tend to be outliers to the remainder of the network affinities.

In the context of dynamically determining the boundaries between the ALCs in LNNAIS, the sequential exception technique can be applied to a sorted set (descending) of network affinities between the ALCs in LNNAIS. The set of network affinities is sorted to guarantee that the lowest network affinities (potential outliers with the highest Euclidean distance) forms part of the first sequential subsets. The first subset, S_1, will then contain the lowest network affinity, followed by S_2 which consists of S_1 and the second lowest network affinity and so forth. The function of dissimilarity $D(S_o)$ in equation (2.67) is calculated as the variance between the network affinities in subset S_o. Therefore the exception set S_e contains the lowest network affinities between the ALCs in LNNAIS and eventually determines the boundaries between the ALCs.

An added advantage of the Sequential Deviation Outlier Technique (SDOT) is that not only is the technique computationally less expensive, but it also has no need for any boundary constraints on K. K is solely determined by the size of S_e. Furthermore, SDOT is a non-parametric technique. The following section discusses the time complexity of SDOT and IPT.

6.3 Time Complexity of SDOT and IPT

The time complexity of both SDOT and IPT are based on the complexity of sorting the network affinities between the ALCs in the ALC population and determining the number of boundaries between the ALCs in the ALC population of size $\mathcal{B}_{max}$. The maximum number of boundaries in an ALC population of size $\mathcal{B}_{max}$ is $\mathcal{B}_{max}$. The time complexity of sorting the $\mathcal{B}_{max}$ network affinities depends on the sorting algorithm used. Assume the time complexity of the sorting algorithm is some constant, χ_1, and that the time complexity of the selected validity index is χ_2. The worst case of time complexity for IPT is when the clustering quality of all possible boundaries needs to be calculated, giving a time complexity of $O(\chi_2 \mathcal{B}_{max} |\mathcal{A}| N)$ where $|\mathcal{A}|$ is the size of the data set that needs to be partitioned and N is the number of dimensions of data set $\mathcal{A}$. The $\mathcal{B}_{max}$ and χ_2 parameters are fixed in advance and usually $\mathcal{B}_{max} << |\mathcal{A}|$. If $\mathcal{B}_{max} << |\mathcal{A}|$ then the time complexity of IPT is $O(|\mathcal{A}|)$ and if $\mathcal{B}_{max} \approx |\mathcal{A}|$ then the time complexity of IPT is $O\left(|\mathcal{A}|^2\right)$. Focusing on SDOT, the maximum number of smoothing factor function evaluations is equal to the size of the ALC population, which is $\mathcal{B}_{max}$. Assume the time complexity of the smoothing function is χ_3. The worst case of time complexity for SDOT is when the smoothing factor of $\mathcal{B}_{max}$ subsets need to be calculated to determine the exception set S_e (as discussed in section 2.6). This gives a time complexity of $O(\chi_3 \mathcal{B}_{max})$ for SDOT. Compared to the time complexity of IPT, the time complexity of SDOT is not influenced by the size of data set $\mathcal{A}$ and also not by the number of dimensions, N.

The following section discusses and compares the results obtained from K-means clustering using the multiple execution technique to determine the number of clusters in a data set and the

Table 6.1 LNNAIS Parameter Values

Data set	$\mathcal{B}_{max}$	ρ	ε_{clone}
iris	25	3	5
two-spiral	20	3	5
hepta	40	3	5
engytime	20	3	10
chainlink	40	3	5
target	30	3	5
ionosphere	20	3	20
glass	20	3	5
image segmentation	30	3	20
spambase	10	5	20

results obtained from LNNAIS using SDOT and IPT to determine the number of clusters in a data set.

6.4 Experimental Results

This section compares and discusses the clustering results obtained by K-means clustering, LNNAIS using IPT, and LNNAIS using SDOT to dynamically determine the number of clusters in a data set. K-means utilises the multiple execution technique with the Q_{DB} (as defined in equation (2.41)) and Q_{RT} (as defined in equation (2.51)) validity indices, referred to as KM$_{DB}$ and KM$_{RT}$, respectively. Two of the LNNAIS models utilises the iterative pruning technique with the same Q_{DB} and Q_{RT} validity indices as K-means, referred to as LNN$_{DB}$ and LNN$_{RT}$, respectively. For the Q_{RT} validity index, parameter c was set to 10 in all the experiments. The value of c was found empirically and values of $c > 10$ have no effect on Q_{RT} for all the data sets. LNN$_{SDOT}$ utilises the sequential deviation outlier technique and thus need no validity index.

All experimental results reported in this section are averages taken over 50 runs, where each run consisted of 1000 iterations of a data set. The parameter values for each data set were empirically found to deliver the best performance for each of the algorithms. The value of K was iterated from $K = 2$ to $K = 12$ for all data sets. Table 6.1 summarises the parameter values used by the respective algorithms for each data set. The clustering quality of the algorithms (based on the number of clusters determined by each of the algorithms) is determined by the Q_{ratio} index, J_{intra} and J_{inter} performance measures (as defined in equations (2.49),(2.17) and (2.16), respec-

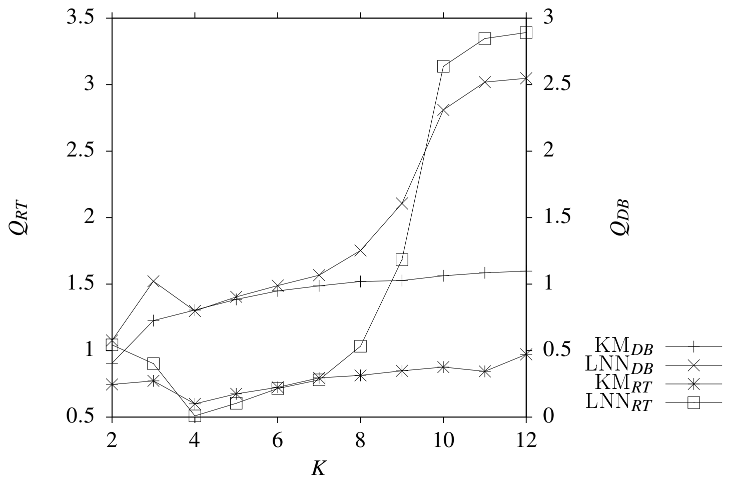


Figure 6.2 Optimal number of clusters obtained by K-means and LNNAIS for the iris data set

tively). The following hypothesis is defined to determine whether there is a difference between the clustering quality of two algorithms for a specific data set or not:

- *Null* hypothesis, H_0: There is no difference in the clustering quality, Q_{ratio}.
- *Alternative* hypothesis, H_1: There is a difference in the clustering quality, Q_{ratio}.

A non-parametric Mann-Whitney U test with a 0.95 confidence interval ($\alpha = 0.05$) was used to test the above hypothesis. The result is statistically significant if the calculated probability (p-value is the probability of H_0 being true) is less than α. In cases where there is a statistical significant difference between the clustering quality of two algorithms, the algorithm with the lowest critical value, z, tends to find clusters in the data set with a higher quality. The results for each of the data sets used are discussed next.

6.4.1 Iris data set

Figure 6.2 illustrates the Q_{RT} values where $c = 10$ for KM$_{RT}$ and LNN$_{RT}$ on the $y1$-axis at different values of K. The Q_{DB} values for KM$_{DB}$ and LNN$_{DB}$ is illustrated on the $y2$-axis of figure 6.2. Figure 6.2 highlights that the optimal number of clusters in the iris data set is obtained by KM$_{RT}$ and LNN$_{RT}$ at $K = 4$ and by KM$_{DB}$ and LNN$_{DB}$ at $K = 2$. Therefore, the optimal range of K is $K = 2$ to $K = 4$ for the iris data set. The average number of clusters determined by LNN$_{SDOT}$ is $K = 2.64$ which falls within the optimal range of K as determined above. Figure 6.3 illustrates for the iris data set the number of clusters respectively determined by the SDOT and IPT

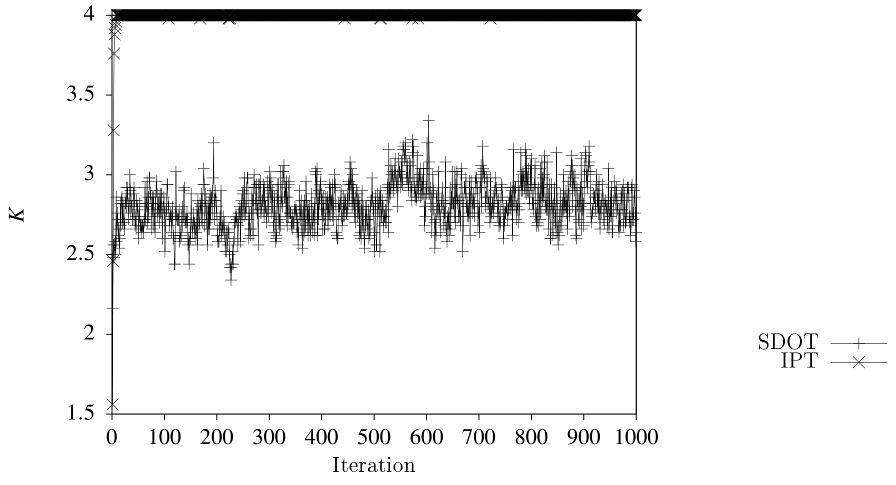


Figure 6.3 Convergence of LNNAIS using SDOT and IPT to optimal K for iris data set

techniques over time. The value of K for IPT rapidly increases to 4 in the first few iterations and remains at 4 for the most of the remaining iterations. The value of K for SDOT increases to 2.7 and oscillates between 2.4 and 3.3 around an average K of 2.64 for the remaining iterations. Since LNNAIS is a stochastic algorithm which utilises a dynamic population of ALCs, the affinities between neighbouring ALCs change over time. Thus, it is expected that the network boundaries detected by SDOT to determine the value of K will also differ over time and oscillate around an average K. Figure 6.4 illustrates a histogram of the frequency distribution of the number of clusters determined by LNN$_{SDOT}$ for the iris data set. The figure illustrates that LNN$_{SDOT}$ has high frequencies at $K = 2$ and $K = 3$. The figure also illustrates that LNN$_{SDOT}$ obtained $K = 4$ for some of the runs, still being within the optimal range of K for the iris data set.

Table 6.2 shows the results obtained by the different models to determine the optimal number of clusters in the iris data set. Referring to table 6.12, the Mann-Whitney U statistical hypothesis test rejects H_0 that the Q_{ratio} means are the same at a 0.05 level of significance between KM$_{RT}$ and LNN$_{SDOT}$ ($z = 7.58$, $p < 0.001$) and between LNN$_{RT}$ and LNN$_{SDOT}$ ($z = 6.69$, $p < 0.001$). Thus, there is a statistical significant difference in the clustering quality, Q_{ratio}, of the iris data set between KM$_{RT}$ and LNN$_{SDOT}$ and between LNN$_{RT}$ and LNN$_{SDOT}$. LNN$_{SDOT}$ tends to find clusters in the iris data set with a higher quality.

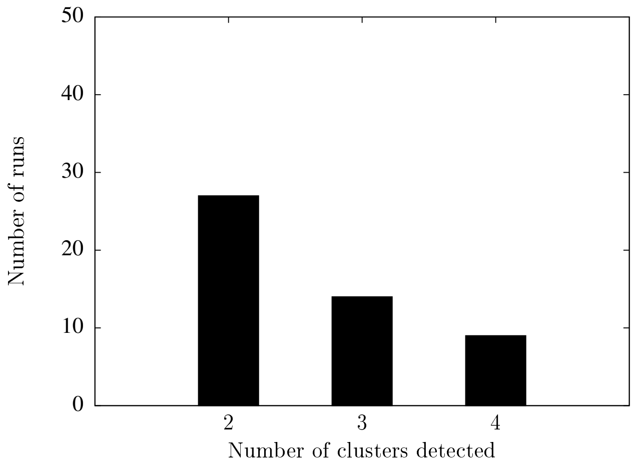


Figure 6.4 Histogram of the number of clusters detected in the iris data set by LNN$_{SDOT}$

Table 6.2 Descriptive Statistics: Iris

Algorithm	K	J_{intra}	J_{inter}	Q_{ratio}	Q_{DB}
KM$_{DB}$	2.00 (± 0.00)	0.856 (± 0.000)	3.927 (± 0.000)	0.218 (± 0.000)	0.405 (± 0.000)
KM$_{RT}$	4.00 (± 0.00)	0.581 (± 0.021)	3.048 (± 0.153)	0.575 (± 0.165)	0.805 (± 0.045)
LNN$_{DB}$	2.00 (± 0.00)	0.923 (± 0.097)	3.994 (± 0.352)	0.233 (± 0.035)	0.432 (± 0.072)
LNN$_{RT}$	4.00 (± 0.00)	0.618 (± 0.036)	3.126 (± 0.221)	0.488 (± 0.154)	0.798 (± 0.154)
LNN$_{SDOT}$	2.64 (± 0.77)	0.788 (± 0.109)	3.738 (± 0.466)	0.364 (± 0.552)	0.643 (± 0.858)

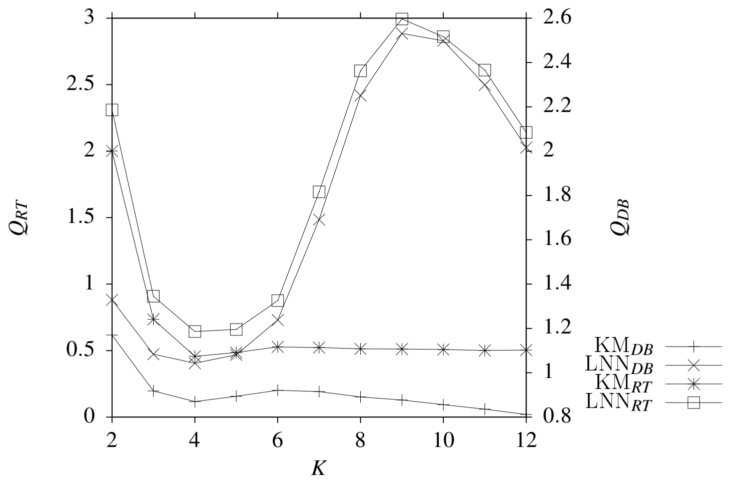


Figure 6.5 Optimal number of clusters obtained by K-means and LNNAIS for the two-spiral data set

6.4.2 Two-spiral data set

The optimal range of K as determined by the different models for the two-spiral data set is $[3,12]$ (as illustrated in figure 6.5). Furthermore, figure 6.5 shows that although the optimal number of clusters in the two-spiral data set is obtained by KM$_{DB}$ at $K = 12$, the majority of the models obtain the optimal number of clusters in the two-spiral data set at $K = 4$. The average number of clusters determined by LNN$_{SDOT}$ is $K = 4.06$ which is similar to the optimal number of clusters obtained by the majority of the models. Figure 6.6 illustrates a histogram of the frequency distribution of the number of clusters determined by LNN$_{SDOT}$ for the two-spiral data set. The figure illustrates that LNN$_{SDOT}$ has high frequencies for $2 \leq K \leq 5$. Figure 6.7 illustrates that for the two-spiral data set the IPT technique converges to $K = 4$ and SDOT oscillates between $K = 3.5$ and $K = 5$ around an average $K = 4.2$ which is near the value of K as determined by IPT. The statistical hypothesis test rejects H_0 that the Q_{ratio} means are the same between KM$_{RT}$ and LNN$_{SDOT}$ ($z = 8.328$, $p < 0.001$). There is thus a statistical significant difference between the clustering quality of KM$_{RT}$ and LNN$_{SDOT}$. KM$_{RT}$ tends to find clusters in the two-spiral data set with a higher quality than LNN$_{SDOT}$. There is however no statistical significant difference between the Q_{ratio} means of LNN$_{RT}$ and LNN$_{SDOT}$ (statistical hypothesis test accepts H_0, refer to table 6.12). Table 6.3 shows the results obtained by the different models to determine the optimal number of clusters in the two-spiral data set.

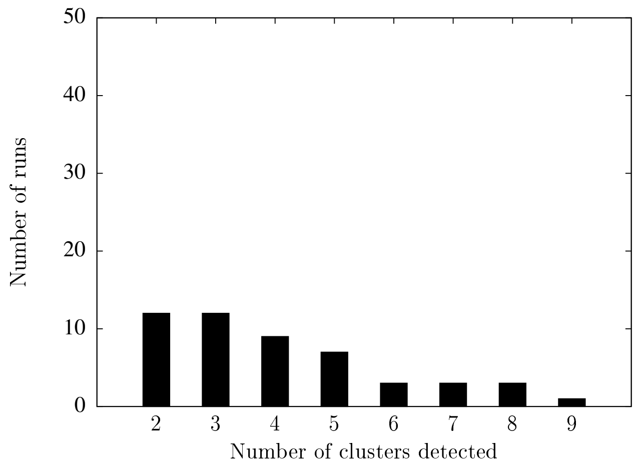


Figure 6.6 Histogram of the number of clusters detected in the two-spiral data set by LNN$_{SDOT}$

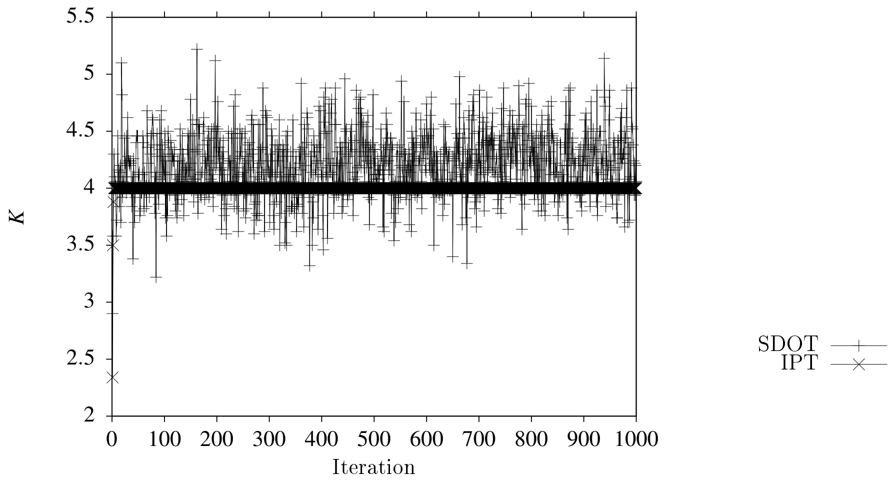


Figure 6.7 Convergence of LNNAIS using SDOT and IPT to optimal K for two-spiral data set

Table 6.3 Descriptive Statistics: Two-spiral

Algorithm	K	J_{intra}	J_{inter}	Q_{ratio}	Q_{DB}
KM$_{DB}$	12.00 (± 0.00)	0.212 (± 0.004)	1.018 (± 0.024)	0.504 (± 0.084)	0.812 (± 0.034)
KM$_{RT}$	4.00 (± 0.00)	0.369 (± 0.003)	0.993 (± 0.011)	0.437 (± 0.016)	0.870 (± 0.031)
LNN$_{DB}$	3.00 (± 0.00)	0.477 (± 0.023)	1.115 (± 0.146)	0.544 (± 0.122)	0.992 (± 0.191)
LNN$_{RT}$	4.00 (± 0.00)	0.405 (± 0.019)	1.021 (± 0.099)	0.616 (± 0.149)	1.043 (± 0.168)
LNN$_{SDOT}$	4.06 (± 1.89)	0.427 (± 0.087)	1.021 (± 0.088)	0.699 (± 0.736)	1.116 (± 0.537)

6.4.3 Hepta data set

The average number of clusters determined by LNN$_{SDOT}$ for the hepta data set is $K = 6.64$ which is close to the true number of clusters in the hepta data set (hepta consists of seven clusters) and falls within the optimal range of K which is $[4,7]$ (as illustrated in figure 6.8). Figure 6.9 illustrates a histogram of the frequency distribution of the number of clusters determined by LNN$_{SDOT}$ for the hepta data set. Figure 6.9 highlights that LNN$_{SDOT}$ has the highest frequency at seven clusters, which is the number of clusters in the hepta data set. Figure 6.10 illustrates for the hepta data set the number of clusters respectively determined by the SDOT and IPT techniques over time. The value of K for IPT converges to 6. The value of K for SDOT oscillates between $K = 6$ and $K = 7$ around an average K of 6.7 for the remaining iterations. Referring to table 6.12, there is a statistical significant difference between the clustering quality of KM$_{RT}$ and LNN$_{SDOT}$ and between LNN$_{RT}$ and LNN$_{SDOT}$. Although KM$_{RT}$ and LNN$_{RT}$ tend to find clusters in the hepta data set with a higher quality than LNN$_{SDOT}$ (refer to table 6.4), LNN$_{SDOT}$ was able to determine the number of clusters in the hepta data set more accurately.

6.4.4 Engytime data set

Table 6.5 shows the results obtained by the different models to determine the optimal number of clusters in the engytime data set. Figure 6.11 illustrates that the optimal range of K for the engytime data set is $2 \leq K \leq 7$ (also shown in table 6.5). LNN$_{SDOT}$ determined the number of clusters in the engytime data set as $K = 3.86$. The histogram of the frequency distribution of the number of clusters determined by LNN$_{SDOT}$ for the engytime data set illustrates that LNN$_{SDOT}$

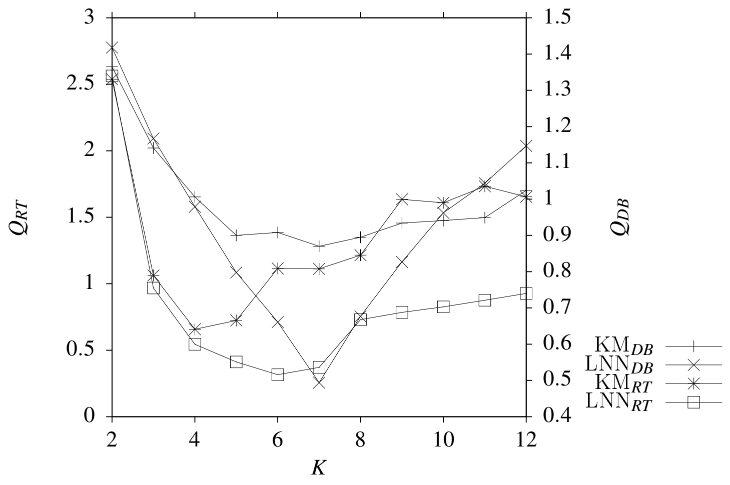


Figure 6.8 Optimal number of clusters obtained by K-means and LNNAIS for the hepta data set

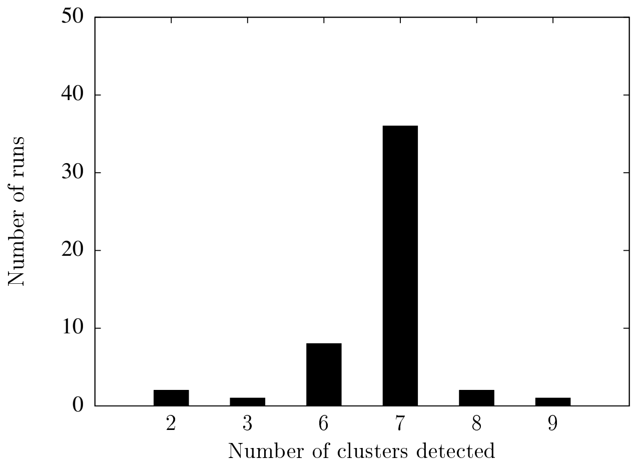


Figure 6.9 Histogram of the number of clusters detected in the hepta data set by LNN_{SDOT}

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

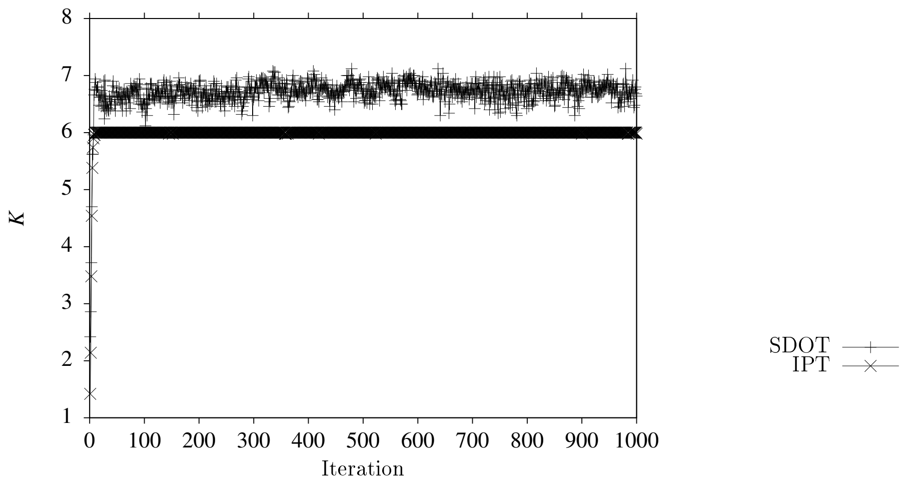


Figure 6.10 Convergence of LNNAIS using SDOT and IPT to optimal K for hepta data set

Table 6.4 Descriptive Statistics: Hepta

Algorithm	K	J_{intra}	J_{inter}	Q_{ratio}	Q_{DB}
KM$_{DB}$	7.00 ($\pm$ 0.00)	0.993 ($\pm$ 0.199)	4.041 ($\pm$ 0.148)	1.112 ($\pm$ 0.459)	0.870 ($\pm$ 0.247)
KM$_{RT}$	4.00 ($\pm$ 0.00)	1.680 ($\pm$ 0.083)	3.902 ($\pm$ 0.184)	0.630 ($\pm$ 0.419)	1.006 ($\pm$ 0.153)
LNN$_{DB}$	6.98 ($\pm$ 0.14)	0.740 ($\pm$ 0.122)	4.161 ($\pm$ 0.097)	0.371 ($\pm$ 0.259)	0.494 ($\pm$ 0.219)
LNN$_{RT}$	5.98 ($\pm$ 0.14)	1.019 ($\pm$ 0.052)	4.307 ($\pm$ 0.146)	0.316 ($\pm$ 0.059)	0.661 ($\pm$ 0.049)
LNN$_{SDOT}$	6.64 ($\pm$ 1.21)	0.830 ($\pm$ 0.397)	4.120 ($\pm$ 0.231)	1.015 ($\pm$ 4.978)	0.541 ($\pm$ 0.365)

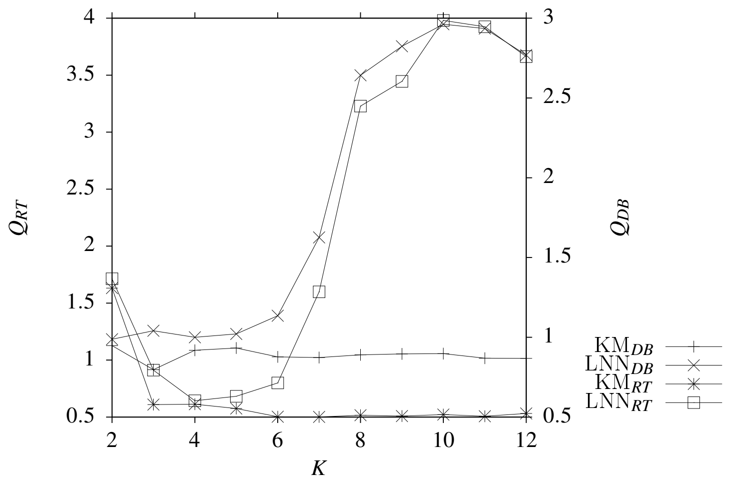


Figure 6.11 Optimal number of clusters obtained by K-means and LNNAIS for the engytime data set

has high frequencies for $2 \leq K \leq 4$ which is within the optimal range of K (refer to figure 6.12 for frequency distribution). Figure 6.13 illustrates that IPT obtains $K = 4$ for all iterations and SDOT oscillates around an average K of 4.4 over time for the engytime data set. There is no statistically significant difference between the clustering quality of any of the models (refer to table 6.12). Therefore, all models tend to deliver clusters with similar quality. LNN_{SDOT} has the advantage of dynamically determining the number of clusters in the engytime data set with similar clustering quality as the other models.

6.4.5 Chainlink data set

The optimal range of K for the chainlink data set is $[8, 12]$ (as illustrated in figure 6.14). Figure 6.15 illustrates that LNN_{SDOT} has high frequencies for $K = 2$ and $4 \leq K \leq 7$ which are not within the optimal range of K. However, the figure also shows that there are cases where LNN_{SDOT} determined the number of clusters within the optimal range of K at lower frequencies. Note that the similarity between the range of determined clusters in figure 6.15 and the range of K for the iterative and multiple execution approaches in figure 6.14 is a coincidence. Figure 6.16 illustrates that IPT obtains $K = 8$ for all iterations and SDOT oscillates around an average K of 6.5 between $K = 5.5$ and $K = 8$ over time for the chainlink data set. The average number of clusters determined by LNN_{SDOT} for the chainlink data set is $K = 5.76$ (refer to table 6.6). Table 6.6 shows the results obtained by the different models to determine the optimal number of clusters in the chainlink data set.

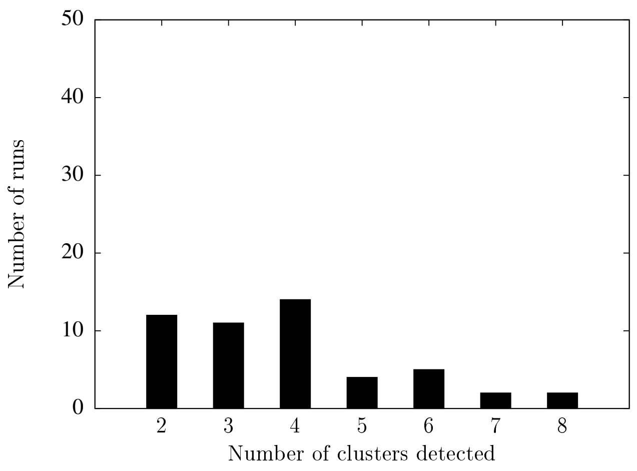


Figure 6.12 Histogram of the number of clusters detected in the engytime data set by LNN$_{SDOT}$

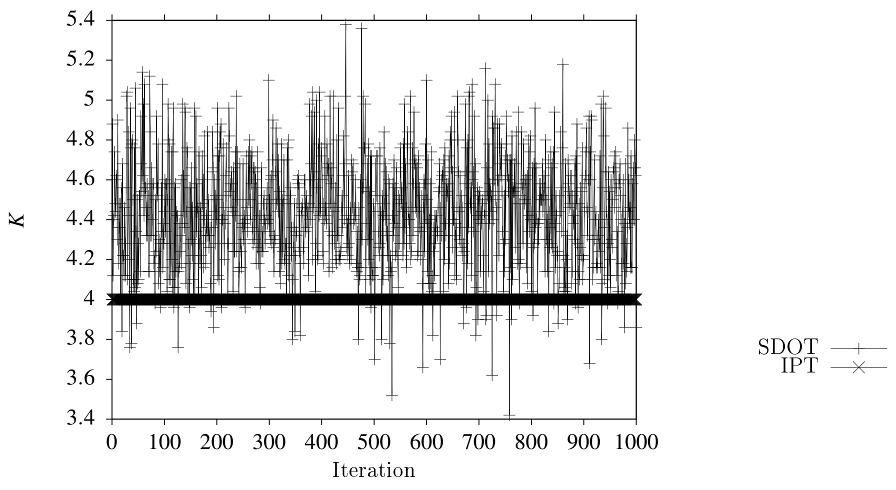


Figure 6.13 Convergence of LNNAIS using SDOT and IPT to optimal K for engytime data set

Table 6.5 Descriptive Statistics: Engytime

Algorithm	K	J_{intra}	J_{inter}	Q_{ratio}	Q_{DB}
KM$_{DB}$	3.00 (± 0.00)	1.165 (± 0.000)	3.184 (± 0.000)	0.396 (± 0.000)	0.797 (± 0.000)
KM$_{RT}$	7.00 (± 0.00)	0.805 (± 0.004)	3.188 (± 0.109)	0.502 (± 0.021)	0.873 (± 0.017)
LNN$_{DB}$	2.00 (± 0.00)	1.833 (± 0.213)	4.133 (± 1.032)	0.465 (± 0.107)	0.910 (± 0.194)
LNN$_{RT}$	4.00 (± 0.00)	1.284 (± 0.113)	4.020 (± 0.712)	0.616 (± 0.226)	1.000 (± 0.258)
LNN$_{SDOT}$	3.86 (± 1.62)	1.381 (± 0.304)	3.978 (± 0.808)	0.582 (± 0.217)	0.992 (± 0.287)

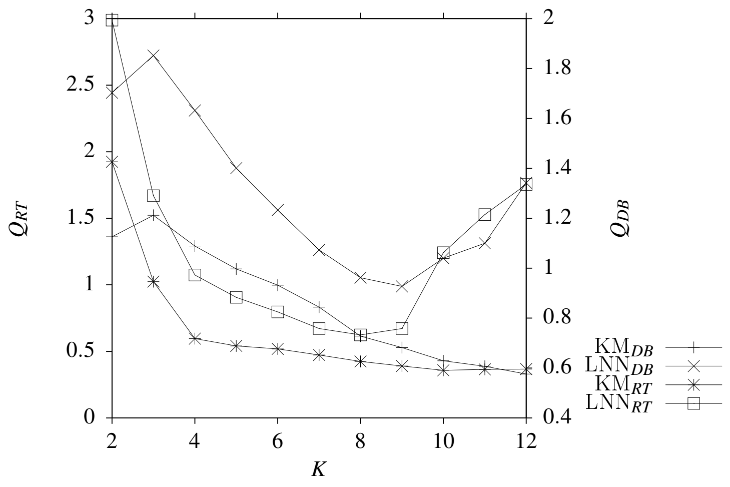


Figure 6.14 Optimal number of clusters obtained by K-means and LNNAIS for the chainlink data set

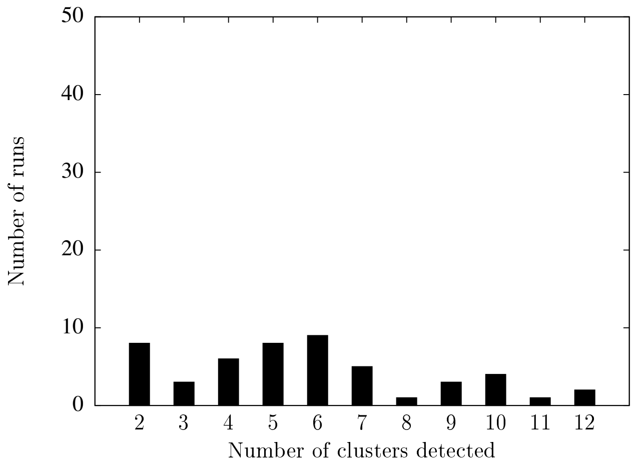


Figure 6.15 Histogram of the number of clusters detected in the chainlink data set by LNN$_{SDOT}$

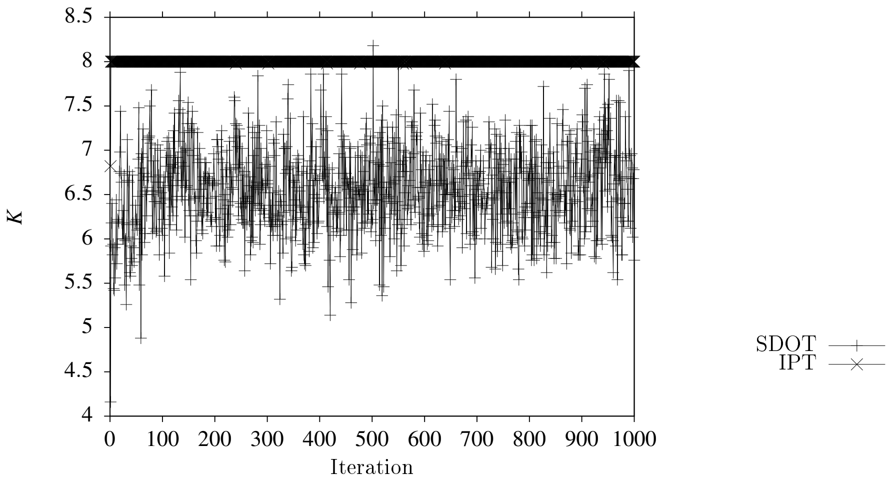


Figure 6.16 Convergence of LNNAIS using SDOT and IPT to optimal K for chainlink data set

Table 6.6 Descriptive Statistics: Chainlink

Algorithm	K	J_{intra}	J_{inter}	Q_{ratio}	Q_{DB}
KM$_{DB}$	12.00 (± 0.00)	0.262 (± 0.009)	1.500 (± 0.025)	0.367 (± 0.063)	0.576 (± 0.017)
KM$_{RT}$	10.00 (± 0.00)	0.308 (± 0.007)	1.509 (± 0.031)	0.358 (± 0.028)	0.629 (± 0.030)
LNN$_{DB}$	9.00 (± 0.00)	0.384 (± 0.018)	1.475 (± 0.068)	0.629 (± 0.210)	0.906 (± 0.144)
LNN$_{RT}$	8.00 (± 0.00)	0.427 (± 0.021)	1.464 (± 0.057)	0.624 (± 0.302)	0.962 (± 0.190)
LNN$_{SDOT}$	5.76 (± 2.76)	0.588 (± 0.184)	1.402 (± 0.235)	0.770 (± 0.400)	1.283 (± 0.666)

Referring to table 6.12, the statistical hypothesis test rejects H_0 that the Q_{ratio} means are the same between KM$_{RT}$ and LNN$_{SDOT}$ ($z = 8.483$, $p < 0.001$). There is thus a statistical significant difference between the clustering quality of KM$_{RT}$ and LNN$_{SDOT}$. KM$_{RT}$ tends to find clusters in the chainlink data set with a higher quality than LNN$_{SDOT}$. There is also a statistical significant difference between the Q_{ratio} means of LNN$_{RT}$ and LNN$_{SDOT}$ ($z = 2.547$, $p = 0.011$). LNN$_{RT}$ tends to find clusters in the chainlink data set with a higher quality than LNN$_{SDOT}$.

6.4.6 Target data set

The average number of clusters determined by LNN$_{SDOT}$ for the target data set is $K = 4.04$ which is close to the optimal range of K (as illustrated in figure 6.17, $5 \leq K \leq 8$). The frequency distribution of the number of clusters determined by LNN$_{SDOT}$ for the target data set is illustrated in figure 6.18. LNN$_{SDOT}$ has high frequencies for $K \leq 5$. Figure 6.19 illustrates for the target data set the number of clusters respectively determined by the SDOT and IPT techniques over time. IPT obtains $K = 6$ for the majority of the iterations. The value of K for SDOT oscillates between $K = 3$ and $K = 5.5$ around an average K of 4.2 for the remaining iterations. Table 6.7 shows the results obtained by the different models to determine the optimal number of clusters in the target data set.

The statistical hypothesis test rejects H_0 that the Q_{ratio} means are the same between KM$_{RT}$ and LNN$_{SDOT}$ ($z = 7.835$, $p < 0.001$). There is thus a statistical significant difference between the clustering quality of KM$_{RT}$ and LNN$_{SDOT}$ and KM$_{RT}$ tends to find clusters in the target data

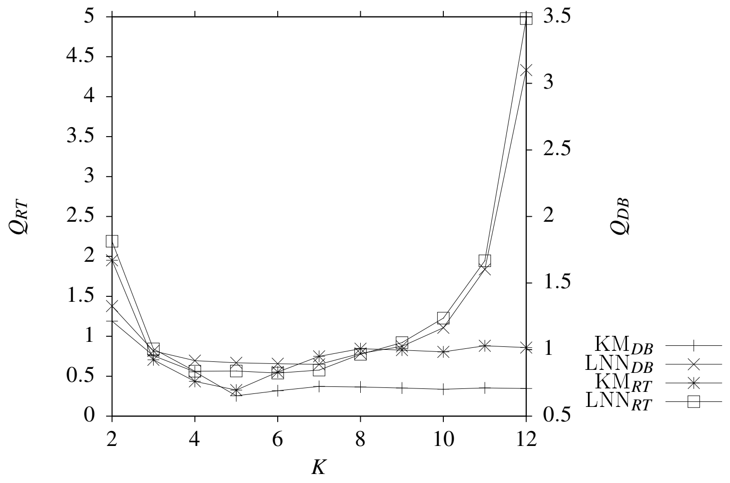


Figure 6.17 Optimal number of clusters obtained by K-means and LNNAIS for the target data set

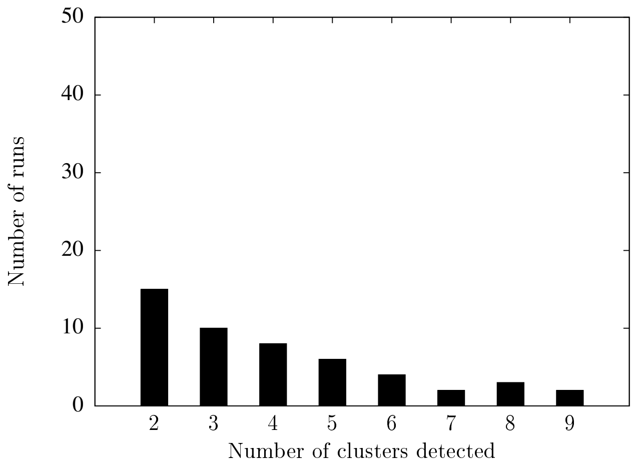


Figure 6.18 Histogram of the number of clusters detected in the target data set by LNN$_{SDOT}$

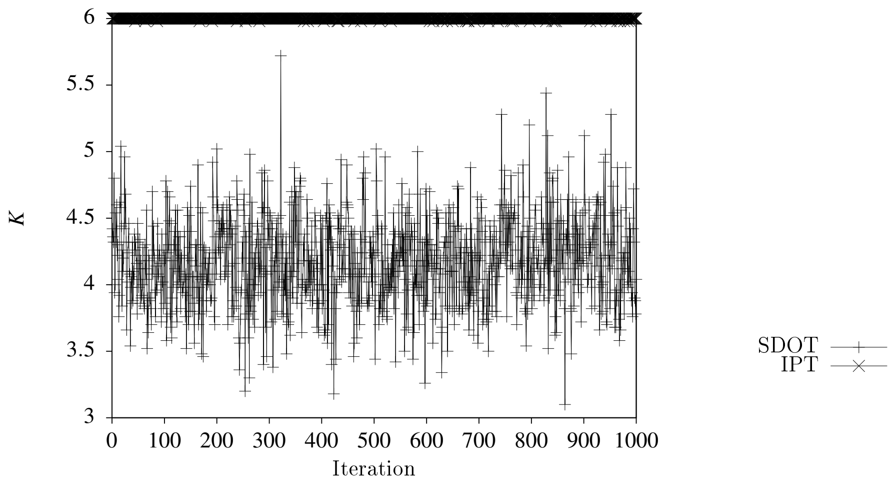


Figure 6.19 Convergence of LNNAIS using SDOT and IPT to optimal K for target data set

Table 6.7 Descriptive Statistics: Target

Algorithm	K	J_{intra}	J_{inter}	Q_{ratio}	Q_{DB}
KM_{DB}	5.00 (± 0.00)	0.533 (± 0.012)	2.313 (± 0.102)	0.326 (± 0.013)	0.653 (± 0.014)
KM_{RT}	5.00 (± 0.00)	0.533 (± 0.012)	2.313 (± 0.102)	0.326 (± 0.013)	0.653 (± 0.014)
LNN_{DB}	7.98 (± 0.14)	0.538 (± 0.075)	3.076 (± 0.343)	0.569 (± 0.477)	0.836 (± 0.284)
LNN_{RT}	6.00 (± 0.00)	0.661 (± 0.117)	2.806 (± 0.417)	0.539 (± 0.178)	0.894 (± 0.225)
LNN_{SDOT}	4.04 (± 2.04)	0.878 (± 0.208)	2.841 (± 0.751)	0.577 (± 0.438)	1.024 (± 0.860)

Table 6.8 Descriptive Statistics: Ionosphere

Algorithm	K	J_{intra}	J_{inter}	Q_{ratio}	Q_{DB}
KM$_{DB}$	2.00 ($\pm$ 0.00)	2.289 ($\pm$ 0.098)	3.156 ($\pm$ 0.413)	0.730 ($\pm$ 0.039)	1.484 ($\pm$ 0.153)
KM$_{RT}$	4.00 ($\pm$ 0.00)	2.085 ($\pm$ 0.065)	3.438 ($\pm$ 0.481)	0.877 ($\pm$ 0.164)	1.776 ($\pm$ 0.283)
LNN$_{DB}$	2.00 ($\pm$ 0.00)	2.888 ($\pm$ 0.278)	4.083 ($\pm$ 0.642)	0.720 ($\pm$ 0.100)	1.437 ($\pm$ 0.257)
LNN$_{RT}$	5.00 ($\pm$ 0.00)	2.473 ($\pm$ 0.272)	4.277 ($\pm$ 0.517)	0.911 ($\pm$ 0.180)	1.755 ($\pm$ 0.258)
LNN$_{SDOT}$	8.28 ($\pm$ 2.12)	2.251 ($\pm$ 0.322)	5.012 ($\pm$ 0.424)	2.791 ($\pm$ 6.519)	1.956 ($\pm$ 1.737)

set with a higher quality than LNN$_{SDOT}$. There is however no statistical significant difference between the Q_{ratio} means of LNN$_{RT}$ and LNN$_{SDOT}$ (statistical hypothesis test accepts H_0, refer to table 6.12).

6.4.7 Ionosphere data set

Table 6.8 shows the results obtained by the different models to determine the optimal number of clusters in the ionosphere data set. Figure 6.20 illustrates that the optimal range of K for the ionosphere data set is $2 \leq K \leq 5$ (also shown in table 6.8). LNN$_{SDOT}$ determined the average number of clusters in the ionosphere data set as $K = 8.28$. The frequency distribution of the number of clusters determined by LNN$_{SDOT}$ for the ionosphere data set illustrates that LNN$_{SDOT}$ has high frequencies for $8 \leq K \leq 11$ which is not within the optimal range of K (refer to figure 6.21 for frequency distribution). Figure 6.22 illustrates for the ionosphere data set the number of clusters respectively determined by the SDOT and IPT techniques over time. The value of K for IPT rapidly increases to 5 in the first few iterations and remains at 5 for the majority of the remaining iterations. The value of K for SDOT rapidly increases to 8 and oscillates between $K = 7$ and $K = 9$ around an average K of 8 for the remaining iterations. Even though there is a difference in the optimal range of K between the models, there is no statistically significant difference between the clustering qualities of any of the models (refer to table 6.12). Therefore, all models tend to deliver clusters with similar quality at different optimal number of clusters. LNN$_{SDOT}$ has the advantage of dynamically determining the number of clusters in the ionosphere data set with similar clustering quality as the other models.

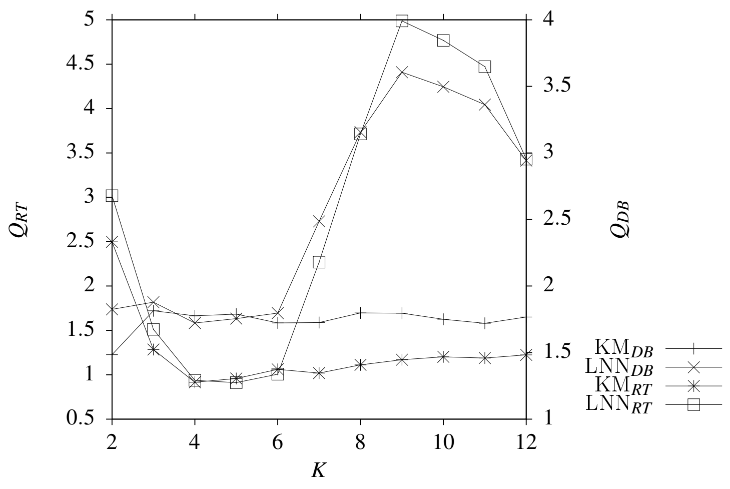


Figure 6.20 Optimal number of clusters obtained by K-means and LNNAIS for the ionosphere data set

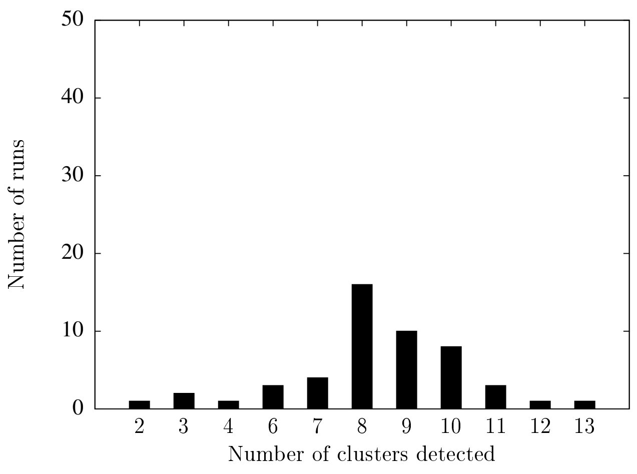


Figure 6.21 Histogram of the number of clusters detected in the ionosphere data set by LNN$_{SDOT}$

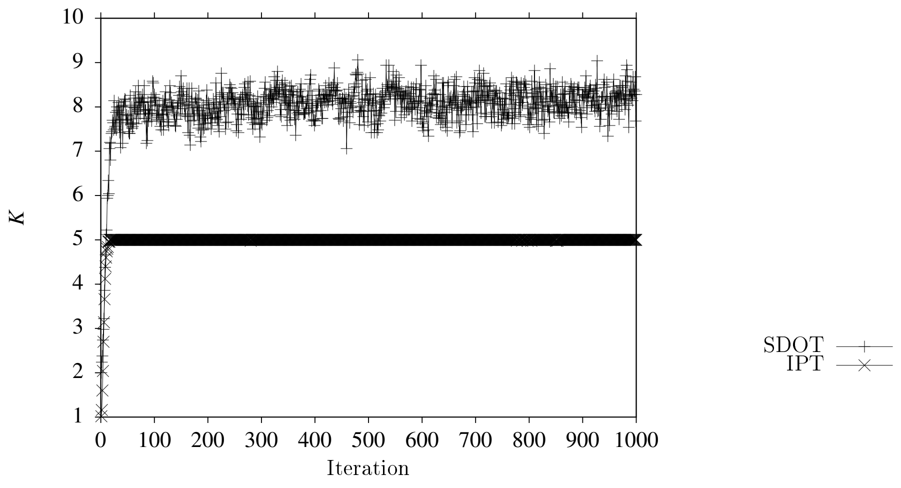


Figure 6.22 Convergence of LNNAIS using SDOT and IPT to optimal K for ionosphere data set

6.4.8 Glass data set

Figure 6.23 shows that the optimal number of clusters in the glass data set is obtained by KM_{DB} and LNN_{DB} at $K = 2$ and by KM_{RT} and LNN_{RT} at $K = 4$. Therefore the optimal range of K as determined by the different models for the glass data set is $[2,4]$. Figure 6.25 illustrates that the value of K for IPT rapidly increases to $K = 4$ and SDOT oscillates around an average K of 3.6 in range $[3,4.5]$ over time for the glass data set. Table 6.9 shows the results obtained by the different models to determine the number of clusters in the glass data set. The average number of clusters determined by LNN_{SDOT} is $K = 3.34$ which falls within the optimal range of K.
A histogram of the frequency distribution of the number of clusters determined by LNN_{SDOT} for the glass data set is illustrated in figure 6.24. LNN_{SDOT} has high frequencies for $K \leq 5$. Referring to table 6.12, the Mann-Whitney U statistical hypothesis test rejects H_0 that the Q_{ratio} means are the same between KM_{RT} and LNN_{SDOT} ($z = 3.364$, $p < 0.001$) and between LNN_{RT} and LNN_{SDOT} ($z = 1.996$, $p = 0.046$). LNN_{SDOT} tends to find clusters in the glass data set with a higher quality than KM_{RT} and LNN_{RT}.

6.4.9 Image Segmentation data set

Table 6.10 shows the results obtained by the different models to determine the optimal number of clusters in the image segmentation data set. Figure 6.26 shows that the optimal number of clusters in the image data set is obtained by KM_{DB} and LNN_{DB} at $K = 2$, by KM_{RT} at $K = 9$ and LNN_{RT} at $K = 3$. The average number of clusters determined by LNN_{SDOT} is $K = 3.28$

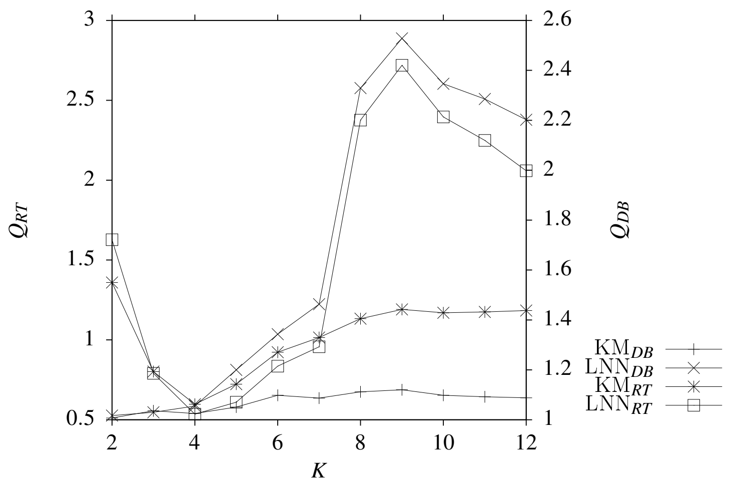


Figure 6.23 Optimal number of clusters obtained by K-means and LNNAIS for the glass data set

Table 6.9 Descriptive Statistics: Glass

Algorithm	K	J_{intra}	J_{inter}	Q_{ratio}	Q_{DB}
KM$_{DB}$	2.00 (± 0.00)	1.531 (± 0.100)	3.879 (± 0.546)	0.397 (± 0.019)	1.007 (± 0.116)
KM$_{RT}$	4.00 (± 0.00)	1.212 (± 0.056)	4.263 (± 0.627)	0.572 (± 0.152)	1.025 (± 0.149)
LNN$_{DB}$	2.00 (± 0.00)	2.354 (± 0.484)	5.792 (± 1.379)	0.427 (± 0.121)	0.892 (± 0.236)
LNN$_{RT}$	4.00 (± 0.00)	1.575 (± 0.208)	5.197 (± 0.769)	0.512 (± 0.161)	1.055 (± 0.266)
LNN$_{SDOT}$	3.34 (± 1.56)	2.003 (± 0.518)	5.998 (± 0.929)	0.493 (± 0.310)	0.875 (± 0.291)

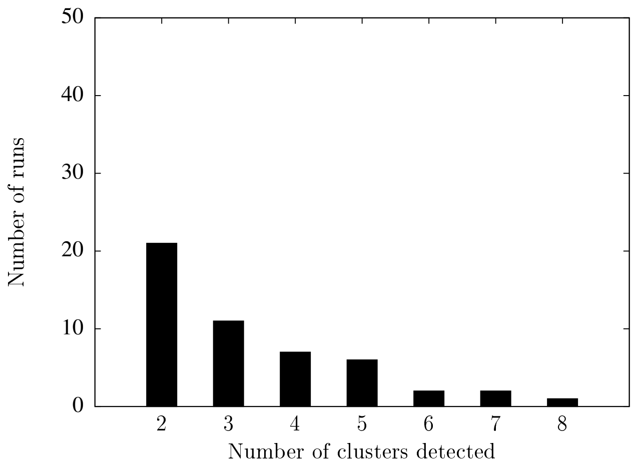


Figure 6.24 Histogram of the number of clusters detected in the glass data set by LNN$_{SDOT}$

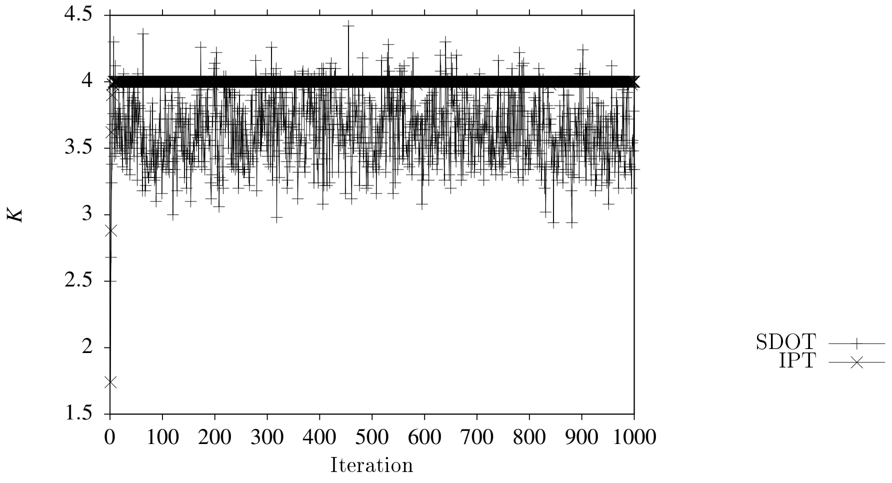


Figure 6.25 Convergence of LNNAIS using SDOT and IPT to optimal K for glass data set

Table 6.10 Descriptive Statistics: Image Segmentation

Algorithm	K	J_{intra}	J_{inter}	Q_{ratio}	Q_{DB}
KM$_{DB}$	2.00 (± 0.00)	101.487 (± 3.313)	238.922 (± 83.995)	0.439 (± 0.041)	0.861 (± 0.024)
KM$_{RT}$	9.00 (± 0.00)	58.442 (± 0.675)	322.656 (± 10.779)	0.688 (± 0.083)	1.021 (± 0.035)
LNN$_{DB}$	2.00 (± 0.00)	168.497 (± 33.481)	1148.026 (± 253.897)	0.155 (± 0.047)	0.551 (± 0.199)
LNN$_{RT}$	3.00 (± 0.00)	137.827 (± 15.718)	881.290 (± 141.104)	0.316 (± 0.253)	1.000 (± 0.602)
LNN$_{SDOT}$	3.28 (± 1.27)	142.847 (± 21.735)	975.017 (± 215.461)	43.919 (± 291.181)	88.577 (± 613.169)

which falls within the optimal range of K. Figure 6.28 illustrates that IPT obtains $K = 3$ for all iterations and SDOT oscillates around an average K of 3.2 in range $[2.6, 3.7]$ over time for the image data set. The frequency distribution of the number of clusters determined by LNN$_{SDOT}$ for the image segmentation data set is illustrated in figure 6.27. LNN$_{SDOT}$ has high frequencies for $K \leq 5$. Referring to table 6.12, the Mann-Whitney U statistical hypothesis test rejects H_0 that the Q_{ratio} means are the same between KM$_{RT}$ and LNN$_{SDOT}$ ($z = 6.89$, $p < 0.001$) and between LNN$_{RT}$ and LNN$_{SDOT}$ ($z = 2.337$, $p = 0.019$). LNN$_{SDOT}$ tends to find clusters in the image segmentation data set with a higher quality than KM$_{RT}$ and LNN$_{RT}$.

6.4.10 Spambase data set

The average number of clusters determined by LNN$_{SDOT}$ for the spambase data set is $K = 2.4$ which is within the optimal range of K (as illustrated in figure 6.29, $2 \leq K \leq 4$). In figure 6.29, note that $Q_{RT} < 0$ for LNN$_{RT}$ where $K \geq 10$. Q_{RT} values less than zero indicates that LNN$_{RT}$ was unable to cluster the data set into the corresponding K clusters. Since $\mathcal{B}_{max} = 10$ for data set spambase (refer to table 6.1), the number of clusters $K \geq 10$ is more than the number of available ALCs in the population. The frequency distribution of the number of clusters determined by LNN$_{SDOT}$ for the spambase data set is illustrated in figure 6.30. LNN$_{SDOT}$ has high frequencies for $K \leq 3$. Figure 6.31 illustrates that IPT obtains $K = 2$ for all iterations and SDOT oscillates around an average K of 2.45 in range $[2.2, 2.7]$ over time for the spambase data set. Table 6.11 shows the results obtained by the different models to determine the optimal number of clusters in the spambase data set. The statistical hypothesis test rejects H_0 that the Q_{ratio} means are the

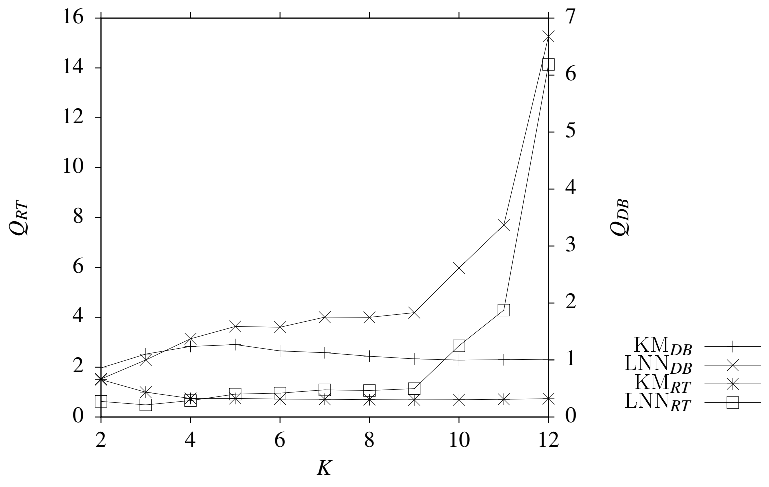


Figure 6.26 Optimal number of clusters obtained by K-means and LNNAIS for the image segmentation data set

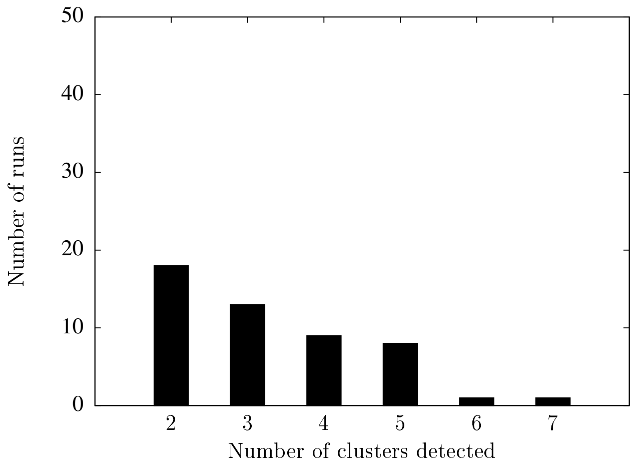


Figure 6.27 Histogram of the number of clusters detected in the image segmentation data set by LNN$_{SDOT}$

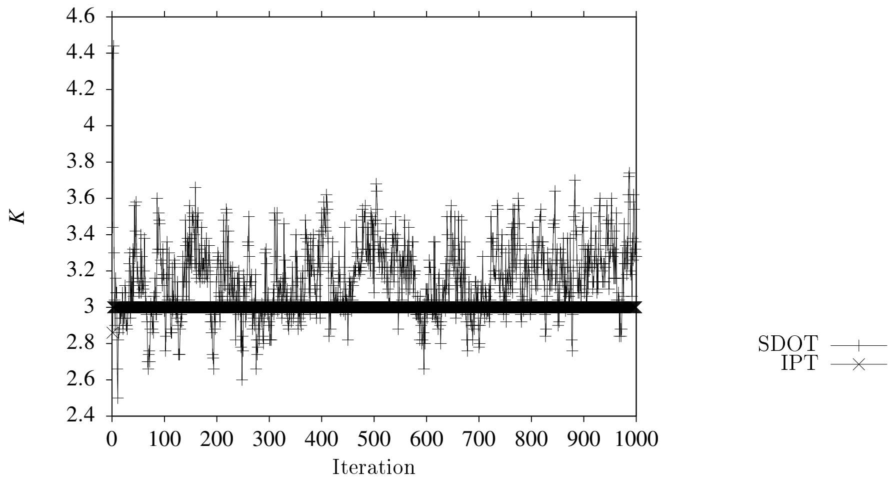


Figure 6.28 Convergence of LNNAIS using SDOT and IPT to optimal K for image data set

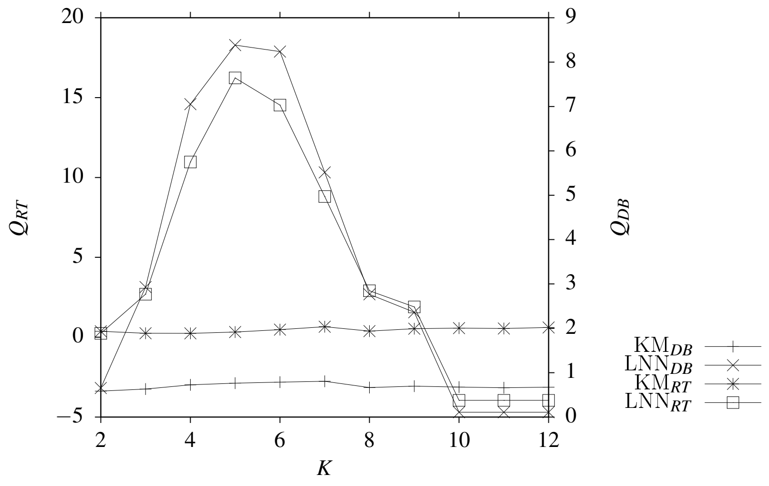


Figure 6.29 Optimal number of clusters obtained by K-means and LNNAIS for the spambase data set

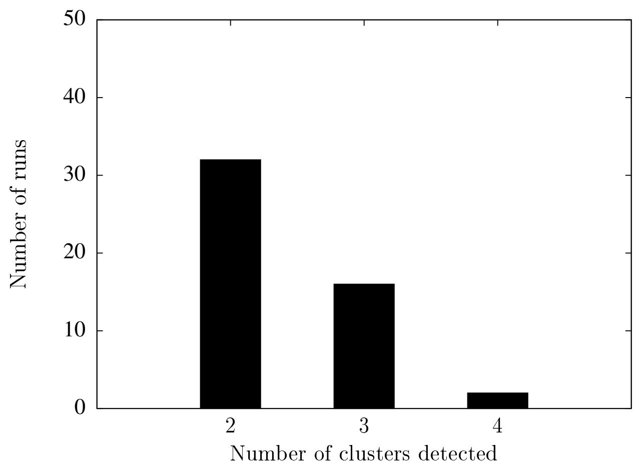


Figure 6.30 Histogram of the number of clusters detected in the spambase data set by LNN_{SDOT}

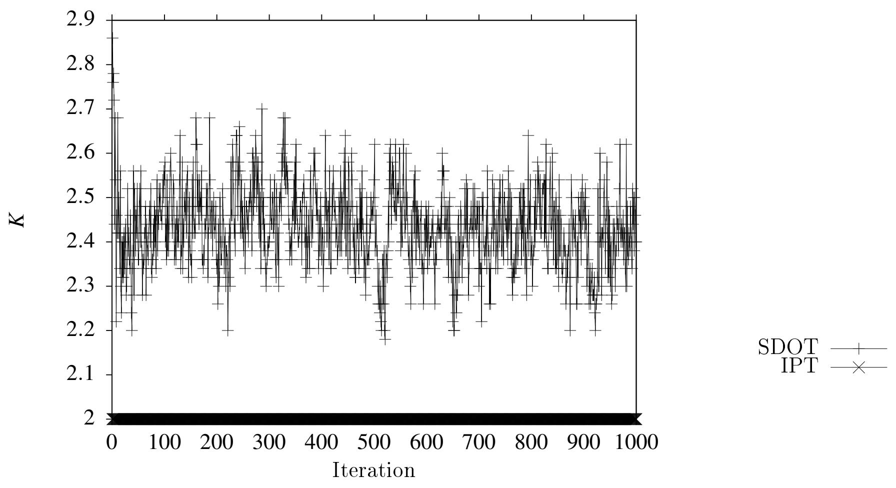


Figure 6.31 Convergence of LNNAIS using SDOT and IPT to optimal K for spambase data set

Table 6.11 Descriptive Statistics: Spambase

Algorithm	K	J_{intra}	J_{inter}	Q_{ratio}	Q_{DB}
KM$_{DB}$	2.00 (± 0.00)	216.058 (± 0.000)	2003.263 (± 0.000)	0.108 (± 0.000)	0.586 (± 0.000)
KM$_{RT}$	4.00 (± 0.00)	129.353 (± 0.000)	2165.832 (± 0.000)	0.229 (± 0.000)	0.727 (± 0.000)
LNN$_{DB}$	2.00 (± 0.00)	771.637 (± 317.716)	8288.589 (± 2462.940)	0.095 (± 0.031)	0.546 (± 0.077)
LNN$_{RT}$	2.00 (± 0.00)	475.834 (± 282.100)	7639.878 (± 2648.505)	0.071 (± 0.053)	0.655 (± 0.171)
LNN$_{SDOT}$	2.40 (± 0.57)	651.896 (± 382.136)	10416.929 (± 2798.913)	0.076 (± 0.042)	0.548 (± 0.222)

same between KM$_{RT}$ and LNN$_{SDOT}$ ($z = 8.269$, $p < 0.001$). There is thus a statistical significant difference between the clustering quality of KM$_{RT}$ and LNN$_{SDOT}$ and LNN$_{SDOT}$ tends to find clusters in the spambase data set with a higher quality than KM$_{RT}$. There is however no statistical significant difference between the Q_{ratio} means of LNN$_{RT}$ and LNN$_{SDOT}$ (statistical hypothesis test accepts H_0, refer to table 6.12).

For completeness, table 6.12 also shows whether there is a statistical significant difference between the clustering quality of KM$_{RT}$ and LNN$_{RT}$ for all the data sets. Referring to table 6.12, LNN$_{SDOT}$ and LNN$_{RT}$ tend to deliver clusters with a similar quality as KM$_{RT}$ for two of the data sets (engytime and ionosphere). Out of the remaining eight data sets, both LNN$_{SDOT}$ and LNN$_{RT}$ deliver clusters of a higher quality than KM$_{RT}$ for five of the data sets. Comparing LNN$_{SDOT}$ with LNN$_{RT}$ for five of the data sets (two-spiral, engytime, target, ionosphere and spambase) LNN$_{SDOT}$ tends to deliver clusters with a similar quality as LNN$_{RT}$. Out of the remaining five data sets, LNN$_{SDOT}$ delivers clusters of a higher quality than LNN$_{RT}$ for four of the data sets. In general, LNN$_{SDOT}$ tends to deliver clusters of similar or higher quality for all data sets, followed by LNN$_{RT}$ and KM$_{RT}$.

Table 6.12 Statistical Hypothesis Testing between All Models for all data sets based on Q_{ratio} as performance criteria ($\alpha = 0.05$; with continuity correction; unpaired; non-directional)

Data set	Model A	Model B	z of A	z of B	p	Outcome	Lowest z-score
iris	LNN$_{SDOT}$	KM$_{RT}$	-7.58	7.58	< 0.001	Reject H_0	LNN$_{SDOT}$
	LNN$_{RT}$	KM$_{RT}$	-3.209	3.209	0.001	Reject H_0	LNN$_{RT}$
	LNN$_{SDOT}$	LNN$_{RT}$	-6.69	6.69	< 0.001	Reject H_0	LNN$_{SDOT}$
two-spiral	LNN$_{SDOT}$	KM$_{RT}$	8.328	-8.328	< 0.001	Reject H_0	KM$_{RT}$
	LNN$_{RT}$	KM$_{RT}$	7.704	-7.704	< 0.001	Reject H_0	KM$_{RT}$
	LNN$_{SDOT}$	LNN$_{RT}$	-0.5	0.5	0.617	Accept H_0	LNN$_{SDOT}$
hepta	LNN$_{SDOT}$	KM$_{RT}$	-6.787	6.787	< 0.001	Reject H_0	LNN$_{SDOT}$
	LNN$_{RT}$	KM$_{RT}$	-8.145	8.145	< 0.001	Reject H_0	LNN$_{RT}$
	LNN$_{SDOT}$	LNN$_{RT}$	-4.391	4.391	< 0.001	Reject H_0	LNN$_{SDOT}$
engytime	LNN$_{SDOT}$	KM$_{RT}$	1.017	-1.017	0.309	Accept H_0	KM$_{RT}$
	LNN$_{RT}$	KM$_{RT}$	1.551	-1.551	0.121	Accept H_0	KM$_{RT}$
	LNN$_{SDOT}$	LNN$_{RT}$	-0.855	0.855	0.393	Accept H_0	LNN$_{SDOT}$
chainlink	LNN$_{SDOT}$	KM$_{RT}$	8.483	-8.483	< 0.001	Reject H_0	KM$_{RT}$
	LNN$_{RT}$	KM$_{RT}$	8.566	-8.566	< 0.001	Reject H_0	KM$_{RT}$
	LNN$_{SDOT}$	LNN$_{RT}$	2.547	-2.547	0.011	Reject H_0	LNN$_{RT}$
target	LNN$_{SDOT}$	KM$_{RT}$	7.835	-7.835	< 0.001	Reject H_0	KM$_{RT}$
	LNN$_{RT}$	KM$_{RT}$	8.145	-8.145	< 0.001	Reject H_0	KM$_{RT}$
	LNN$_{SDOT}$	LNN$_{RT}$	-0.221	0.221	0.825	Accept H_0	LNN$_{SDOT}$
ionosphere	LNN$_{SDOT}$	KM$_{RT}$	0.955	-0.955	0.340	Accept H_0	KM$_{RT}$
	LNN$_{RT}$	KM$_{RT}$	1.169	-1.169	0.243	Accept H_0	KM$_{RT}$
	LNN$_{SDOT}$	LNN$_{RT}$	0.283	-0.283	0.777	Accept H_0	LNN$_{RT}$
glass	LNN$_{SDOT}$	KM$_{RT}$	-3.364	3.364	< 0.001	Reject H_0	LNN$_{SDOT}$
	LNN$_{RT}$	KM$_{RT}$	-1.965	1.965	0.049	Reject H_0	LNN$_{RT}$
	LNN$_{SDOT}$	LNN$_{RT}$	-1.996	1.996	0.046	Reject H_0	LNN$_{SDOT}$
image segmentation	LNN$_{SDOT}$	KM$_{RT}$	-6.89	6.89	< 0.001	Reject H_0	LNN$_{SDOT}$
	LNN$_{RT}$	KM$_{RT}$	-7.18	7.18	< 0.001	Reject H_0	LNN$_{RT}$
	LNN$_{SDOT}$	LNN$_{RT}$	-2.337	2.337	0.019	Reject H_0	LNN$_{SDOT}$
spambase	LNN$_{SDOT}$	KM$_{RT}$	-8.269	8.269	< 0.001	Reject H_0	LNN$_{SDOT}$
	LNN$_{RT}$	KM$_{RT}$	-8.269	8.269	< 0.001	Reject H_0	LNN$_{RT}$
	LNN$_{SDOT}$	LNN$_{RT}$	1.275	-1.275	0.202	Accept H_0	LNN$_{RT}$

6.5 Influence of LNN$_{SDOT}$ Parameters

This section investigates the influence of the LNN$_{SDOT}$ parameters on the number of obtained clusters, K, in a data set. These parameters are the maximum population size, $\mathcal{B}_{max}$, the neighbourhood size, ρ, and the clonal level threshold, ε_{clone}. The influence of each parameter was evaluated for all the data sets listed in table 6.1 with the remaining parameters fixed at the values given in table 6.1.

Table 6.13: Effect of $\mathcal{B}_{max}$ on the number of detected clusters, K, by LNN$_{SDOT}$

Data set	$\mathcal{B}_{max}$	Optimal range	K
iris	10	$[2,4]$	2.48 $\pm$0.608
	15		2.58 $\pm$0.751
	20		2.84 $\pm$0.857
	25		2.64 $\pm$0.768
	30		2.88 $\pm$1.070
	35		2.64 $\pm$0.866
	40		2.88 $\pm$0.952
two-spiral	10	$[3,12]$	3.46 $\pm$1.445
	15		4.16 $\pm$1.804
	20		4.06 $\pm$1.891
	25		4.82 $\pm$2.447
	30		4.56 $\pm$2.410
	35		4.32 $\pm$2.140
	40		5.40 $\pm$3.521
hepta	10	$[4,7]$	4.28 $\pm$1.470
	15		5.84 $\pm$1.332
	20		6.18 $\pm$1.571
	25		6.40 $\pm$1.281
	30		6.60 $\pm$1.149
	35		6.82 $\pm$0.712
	40		6.64 $\pm$1.213
engytime	10	$[2,7]$	3.32 $\pm$1.009
	15		3.98 $\pm$1.543
	20		3.86 $\pm$1.625
	25		5.46 $\pm$2.586
	30		5.20 $\pm$3.013

Continued on next page

Data set	$\mathcal{B}_{max}$	Optimal range	K
	35		5.48 ±3.093
	40		6.34 ±4.043
chainlink	10	[8, 12]	3.48 ±1.330
	15		4.64 ±2.278
	20		4.74 ±2.423
	25		6.54 ±3.145
	30		6.18 ±3.315
	35		5.78 ±3.472
	40		5.76 ±2.761
target	10	[5, 8]	3.22 ±1.316
	15		4.16 ±1.901
	20		4.24 ±1.715
	25		4.08 ±2.505
	30		4.04 ±2.039
	35		3.96 ±2.433
	40		3.50 ±1.792
ionosphere	10	[2, 5]	4.24 ±1.069
	15		6.40 ±1.510
	20		8.28 ±2.117
	25		10.16 ±2.716
	30		13.06 ±1.654
	35		15.72 ±2.764
	40		16.48 ±4.813
glass	10	[2, 4]	3.22 ±1.238
	15		3.72 ±1.698
	20		3.34 ±1.557
	25		3.94 ±2.195
	30		3.68 ±2.083
	35		3.80 ±2.010
	40		3.94 ±2.378
	10		2.46 ±0.727

Continued on next page

Data set	$\mathcal{B}_{max}$	Optimal range	K
image	15	[2,9]	2.78 ±1.045
	20		3.08 ±1.197
	25		3.58 ±1.443
	30		3.28 ±1.266
	35		3.20 ±1.296
	40		3.52 ±2.823
spam	10	[2,4]	2.40 ±0.566
	15		2.82 ±1.260
	20		3.08 ±1.324
	25		2.90 ±0.900
	30		3.22 ±1.301
	35		3.24 ±1.305
	40		3.30 ±1.300

The influence of $\mathcal{B}_{max}$ was evaluated for all the data sets with the remaining parameters set to the values as listed in table 6.1. Table 6.13 summarises the results of the average detected K for each data set at different values of $\mathcal{B}_{max}$. There is a gradual to no increase in the number of obtained clusters, K, with an increase in $\mathcal{B}_{max}$ (as shown in table 6.13 for data sets iris, two-spiral, hepta, engytime, ionosphere, glass, image segmentation and spambase). There are also cases where K increases to a maximum and then starts to decrease with an increase in $\mathcal{B}_{max}$ (data sets chainlink and target). The effect of $\mathcal{B}_{max}$ on the number of obtained clusters for the ionosphere data set shows that $\mathcal{B}_{max} \geq 15$ tends to overfit the data since the number of obtained clusters is outside the optimal range. Therefore, the clustering performance of LNN$_{SDOT}$ with regards to K is sensitive to the value of $\mathcal{B}_{max}$.

Table 6.14: Effect of ε_{clone} on the number of detected clusters, K, by LNN$_{SDOT}$

Data set	ε_{clone}	Optimal range	K
	5		2.64 ±0.768

Continued on next page

Data set	ε_{clone}	Optimal range	K
iris	10	$[2,4]$	3.04 $\pm$1.326
	15		3.08 $\pm$1.383
	20		3.62 $\pm$1.864
two-spiral	5	$[3,12]$	4.06 $\pm$1.891
	10		4.92 $\pm$2.415
	15		4.62 $\pm$2.297
	20		5.14 $\pm$2.136
hepta	5	$[4,7]$	6.64 $\pm$1.213
	10		6.78 $\pm$1.346
	15		6.66 $\pm$1.365
	20		6.94 $\pm$0.968
engytime	5	$[2,7]$	3.98 $\pm$1.923
	10		3.86 $\pm$1.625
	15		4.64 $\pm$2.124
	20		4.40 $\pm$1.811
chainlink	5	$[8,12]$	5.76 $\pm$2.761
	10		6.76 $\pm$3.456
	15		7.98 $\pm$4.236
	20		7.68 $\pm$4.420
target	5	$[5,8]$	4.04 $\pm$2.039
	10		4.30 $\pm$2.385
	15		4.24 $\pm$2.526
	20		4.60 $\pm$2.400
ionosphere	5	$[2,5]$	5.38 $\pm$2.553
	10		7.50 $\pm$1.652
	15		7.50 $\pm$2.238
	20		8.28 $\pm$2.117
glass	5	$[2,4]$	3.34 $\pm$1.557
	10		4.32 $\pm$1.794
	15		4.54 $\pm$2.427
	20		4.56 $\pm$2.080

Continued on next page

Data set	ε_{clone}	Optimal range	K
image	5	$[2,9]$	3.20 $\pm$2.000
	10		3.08 $\pm$1.573
	15		3.38 $\pm$2.481
	20		3.28 $\pm$1.266
spam	5	$[2,4]$	2.24 $\pm$0.550
	10		2.32 $\pm$0.546
	15		2.30 $\pm$0.500
	20		2.40 $\pm$0.566

The influence of ε_{clone} was evaluated for all the data sets with the remaining parameters set to the values as listed in table 6.1. Table 6.14 summarises the results of the average detected K for each data set at different values of ε_{clone}. There is a gradual or no increase in the number of obtained clusters, K, with an increase in ε_{clone} for all of the data sets (as shown in table 6.14). Therefore, the clustering performance of LNN$_{SDOT}$ with regards to K is sensitive to the value of ε_{clone}.

Table 6.15: Effect of ρ on the number of detected clusters, K, by LNN$_{SDOT}$

Data set	ρ	Optimal range	K
iris	3	$[2,4]$	2.64 $\pm$0.768
	4		2.84 $\pm$0.833
	5		2.58 $\pm$0.724
two-spiral	3	$[3,12]$	4.06 $\pm$1.891
	4		3.62 $\pm$1.948
	5		4.46 $\pm$3.517
hepta	3	$[4,7]$	6.64 $\pm$1.213
	4		6.58 $\pm$1.812
	5		5.80 $\pm$2.307
engytime	3	$[2,7]$	3.86 $\pm$1.625
	4		4.14 $\pm$1.980

Continued on next page

Data set	ρ	Optimal range	K
	5		3.86 ±2.530
chainlink	3	[8, 12]	5.76 ±2.761
	4		5.32 ±2.596
	5		5.10 ±3.775
target	3	[5, 8]	4.04 ±2.039
	4		4.18 ±2.733
	5		4.20 ±2.828
ionosphere	3	[2, 5]	8.28 ±2.117
	4		6.16 ±2.230
	5		5.12 ±1.935
glass	3	[2, 4]	3.34 ±1.557
	4		3.76 ±2.006
	5		3.82 ±2.381
image	3	[2, 9]	3.28 ±1.266
	4		3.48 ±1.910
	5		3.00 ±1.149
spam	3	[2, 4]	2.60 ±0.800
	4		2.72 ±0.694
	5		2.40 ±0.566

The influence of ρ was evaluated for all the data sets with the remaining parameters set to the values as listed in table 6.1. Table 6.15 summarises the results of the average detected K for each data set at different values of ρ. There is generally no trend in the number of obtained clusters, K, with an increase in ρ except for the ionosphere data set where an increase in ρ decreases K (as shown in table 6.15). Therefore, the clustering performance of LNN$_{SDOT}$ with regards to K is generally insensitive to the value of ρ.

6.6 Conclusion

This chapter presented two techniques which can be used with LNNAIS to dynamically determine the number of clusters in a data set. These techniques are the iterative pruning technique (IPT) and the sequential deviation outlier technique (SDOT). Although both of these techniques are computationally less expensive than the multiple execution approaches, the IPT technique either needs a specified range for K or needs to iterate through all possible edges (to a maximum of $\mathcal{B}_{max}$) which makes the IPT technique parameter dependant in the former case and computationally slightly more expensive than SDOT in the latter. An advantage of IPT is that the technique can use any cluster validity index to determine the number of clusters. The SDOT technique neither uses a cluster validity index nor does it require any boundary constraints on K. SDOT is a non-parametric technique. This is an advantage, since it is not always feasible to visually inspect formed clusters, and a specified range for K might not contain the optimum number of clusters.

LNN$_{RT}$, LNN$_{DB}$ (both using IPT with Q_{RT} and Q_{DB}, respectively) and LNN$_{SDOT}$ (using SDOT) were applied on different data sets to determine the optimal number of clusters. These results were compared to the results obtained from K-means clustering which used the multiple execution approach to determine the optimal number of clusters in each data set. Based on the Q_{ratio} index, in general, LNN$_{SDOT}$ tends to deliver clusters of similar or higher quality for all data sets, followed by LNN$_{RT}$ and KM$_{RT}$. The influence of the different LNN$_{SDOT}$ parameters was also investigated.

Since the LNN$_{SDOT}$ model is computationally less expensive and is able to dynamically determine the number of clusters in a data set, the model can be seen as an enhancement to the LNNAIS model. Due to the possibility of the LNN$_{SDOT}$ model to dynamically determine the number of clusters, the model might indicate division or merging of clusters in a non-stationary environment. The next chapter defines and discusses different non-stationary environments and applies the proposed LNNAIS and LNN$_{SDOT}$ to the clustering of generated synthetic non-stationary data.

Chapter 7

Data Clustering in Non-stationary Environments using a Local Network Neighbourhood Artificial Immune System

A non-stationary environment can be defined as feature vectors in space which move or adapt to different spatial positions over time [64]. The data is thus dynamic over time. Clustering of non-stationary data results into different partitions of the data at different points in time and depends on the severity and the frequency of change in the data. Therefore, from a clustering perspective in a non-stationary environment, the initial formed clusters of a data set can *adapt* over time. This means that, at each time step, the feature vectors associated with different clusters can follow different *migration* types to and from other clusters. The *migration* of feature vectors from one cluster to another implies that the centroids of the different clusters can also move in space to different positions. Therefore, clusters (centroids) may move, disappear and/or new clusters may appear.

This chapter investigates different data migration types and proposes a technique to generate artificial non-stationary data which follows different migration types. Furthermore, the chapter revises the proposed clustering performance measures in section 2.5 which are more applicable to measure the clustering quality in a non-stationary environment compared to the clustering performance measures for stationary environments. The proposed clustering performance measures are then used to compare the clustering results of LNNAIS and LNN$_{SDOT}$ with two other network based artificial immune models.

Section 7.1 revises the clustering performance measures as discussed in section 2.5 that can be used to evaluate the partitioning quality of clustering algorithms in non-stationary environments. This is followed by a discussion and investigation into different data *migration* types in section 7.2. Section 7.3 proposes a technique to generate artificial non-stationary data with different migration types. Section 7.4 analyses and discusses the sensitivity of the LNNAIS parameters on the different artificial non-stationary data sets for each of the defined data migration types. Section 7.5 discusses and compares the clustering results obtained by LNNAIS, LNN$_{SDOT}$, SMAIN and DWB to cluster generated artificial non-stationary data in different dimensions with different cluster sizes, frequencies of change and severities of change.

7.1 Clustering Performance Measures for Non-stationary Environments

Clustering of a data set at a specific point in time, t, is the partitioning of the data set such that patterns within the same partition are more *similar* when compared to patterns which form part of different partitions. The partitioning of a data set into different clusters may differ among different clustering algorithms at a specific point in time. Also, clusters may differ among different points in time. Therefore the clustering quality needs to be evaluated at each point in time. The quality of the clusters can be validated using a cluster validity index. The cluster validity index used in this chapter was proposed by Ray and Turi [151] (as defined in equation (2.49)).

In the context of clustering of non-stationary environments, J_{intra} (cluster compactness) and J_{inter} (cluster separation) can be used, in addition to the validity index, Q_{ratio}, to quantify the quality of partitioning by clustering algorithms over time. An example of a clustering algorithm's partitioning in a non-stationary environment was given in section 2.5. The average separation (J_{inter}) plotted against time will increase in value if the clusters become more separated in time, i.e. clusters move away from one another. If there is any *migration* of feature vectors between clusters, it is expected that the average *intra-clustering* distance (J_{intra}) plotted against time will fluctuate from the time of *migration* until the feature vectors become stationary again (data *migration* types are discussed in section 7.2). It is expected that a change in the number of clusters will result in a change in J_{intra} and/or J_{inter}.

The measured clustering quality (Q_{ratio}) of different clustering algorithms can be compared by averaging each algorithm's clustering quality measure at each step in time over the total running time T (as proposed in section 2.8). This will give each algorithm a mean value of measured clustering quality, $\bar{Q}_i$, for a specific run, i, calculated as

$$\bar{Q}_i = \frac{\sum_{t=1}^{T} Q_{best}(t)}{T} \tag{7.1}$$

where $Q_{best}(t)$ is the cluster quality at time t, calculated as

$$Q_{best}(t) = \min\left\{Q_{ratio}\left(K^{(t)}\right) : 1 < K^{(t)} \leq |P^{(t)}|\right\} \tag{7.2}$$

and Q_{ratio} is minimised by an optimal partitioning of data set $P^{(t)}$ into $K^{(t)}$ clusters at time t. The trajectory of the clustering quality across the entire dynamic landscape is then calculated by averaging $\bar{Q}_i$ over the number of independent runs E, referred to as the collective mean quality. The collective mean quality, $\hat{Q}$, is calculated as

$$\hat{Q} = \frac{\sum_{i=1}^{E} \bar{Q}_i}{E} \tag{7.3}$$

$\hat{Q}$ is derived from the collective mean fitness which is defined in [131] for function optimisation in non-stationary environments. The collective mean fitness takes into account the fitness trajectory across the entire dynamic landscape [131] by averaging the mean value of measured performance over the number of runs E.

The next section discusses the different data *migration* types that can occur in non-stationary environments, as investigated in this chapter.

7.2 Data Migration Types

Feature vectors in a data set can change at any point in time with different severities. Feature vectors in a non-stationary data set can follow different migration types. Based on the different migration types which were discussed in sections 2.4 and 2.5, three generic data migration types can be identified:

- *Pattern migration*: A feature vector can migrate from one cluster to another in the data set. The severity of change, $\tilde{s}$, can be expressed as a percentage of all the feature vectors among the different clusters in the data set which can each migrate to a randomly selected cluster in the data set. Pattern migration can result in some clusters to decrease in size while other clusters increase in size.

- *Cluster migration*: Similar to pattern migration, but instead of selecting a ratio of feature vectors for migration, a fraction of the number of clusters in the data set is selected for migration. Each pattern of a selected cluster migrates to a randomly selected cluster (which was not selected for migration). Cluster migration will result in the disappearance of clusters. In the case where the patterns of the selected cluster migrate to random spatial positions, new clusters will appear.

- *Centroid migration*: All the clusters in a data set adapt the spatial position of their centroids in such a way that feature vectors associated with each cluster remain part of that cluster after the change. The severity of change for centroid migration is discussed in more detail in section 7.3. Centroid migration can result in merging or division of clusters.

The number of times data changes occur within a fixed period of time is referred to as the *frequency* of change, $\tilde{f}$. A higher number of changes within a fixed period of time results in a higher frequency of change in the data and vice versa.

The next section discusses the procedure followed to generate artificial non-stationary data with different frequencies and severities of change in the data.

7.3 Generating Artificial Non-stationary Data

Referring to section 2.1, a cluster, C_k, is a partition of patterns and is represented by a centroid, $\mathbf{c}_k$. The distances between a centroid and the patterns of the cluster determine the *compactness* of the cluster. Therefore a cluster can be generated using the following multidimensional Gaussian function:

$$g\left(\mathbf{x}_k, \mathbf{c}_k\right) = a \exp^{-\left[\sum_{n=1}^{N} \frac{\left(x_{k,n}-c_{k,n}\right)^2}{2\sigma_{k,n}^2}\right]} \tag{7.4}$$

where a is the amplitude, $\mathbf{x}_{k,n}$ is the offset from the centroid $\mathbf{c}_{k,n}$ in dimension n, and $\sigma_{k,n}$ is the spread (compactness) in dimension n. Different types of clusters can be generated using these Gaussian function parameters differing in centroid position ($\mathbf{c}$), compactness (σ) and patterns ($\mathbf{c} \pm \mathbf{x}$) which is associated with the cluster.

In order to simulate a non-stationary environment, the generated Gaussian clusters need to follow a specific migration type (the different migration types were discussed in section 7.2). Focusing on *centroid migration*, the centroid of a cluster needs to change along a specific path. A hypersphere in N-dimensional space ((N-1)-dimensional sphere) with a radius, φ, and middle point, $\mathbf{m}$, was used as the path. A centroid, $\mathbf{c}_k$, is represented by an angle vector, θ_k. The angle vector is projected onto the surface of the (N-1)-dimensional sphere to determine the centroid $\mathbf{c}_k$. A change in $\theta_{k,n-1}$ will result in a projected change in $\mathbf{c}_{k,n}$ and move the cluster centroid on the surface of the (N-1)-dimensional sphere. The severity of change in an angle is a ratio, $\frac{\tilde{s}}{10}$, of the angle difference between $\theta_{k,n-1}$ and a randomly generated angle sampled from $U[0,\pi]$.

Let $\mathbf{c}^{'}_{k,n}$ be the new centroid position in dimension n after a change in $\theta_{k,n-1}$. The spread of patterns in cluster C_k (compactness) needs to remain the same as before the change. Therefore the offset (vector difference) between $\mathbf{c}^{'}_{k,n}$ and the previous centroid $\mathbf{c}_{k,n}$ needs to be added (vector addition) to each pattern in cluster C_k.

The remaining migration types are simulated by removing patterns from a cluster and generating new random patterns for the remaining clusters such that the initial total number of patterns, $|P|$, remains the same. The data in this chapter was generated in different dimensions ($N \in [3,8,15]$). Each data set initially consists of eight clusters ($K = 8$) which are uniformly distributed on the surface of the (N-1)-dimensional sphere ($\varphi = 15$ and $\mathbf{m}$ is zero in every dimension). Each cluster is generated with the Gaussian function of equation (7.4) where $a = 1$ and $\sigma = 1$ (for each dimension). Clusters initially have the same size. The clusters are also generated in different sizes of $[10,25,50]$ ($|P| = K \times |C| \in [80,200,400]$). Data was generated for each migration type at different frequencies, $\tilde{f} \in [1,2,3,4,5]$, and different severities of change, $\tilde{s} \in [1,2,3,4,5]$ giving 225 different non-stationary data sets for each migration type. The generated artificial non-stationary data sets represent a good distribution of data clustering problems in non-stationary environments with the number of features in the range $[3,8,15]$ and the number of patterns in the range $[80,200,400]$ which change/migrate at different frequencies $\tilde{f} \in [1,2,3,4,5]$ and severities $\tilde{s} \in [1,2,3,4,5]$ of change. The different values of $\tilde{f}$ and $\tilde{s}$ are expressed as a ratio $\frac{\tilde{f}}{10}$ and $\frac{\tilde{s}}{10}$ respectively. The ratio of $\tilde{f}$ is then multiplied with the total number of time steps T to determine the time step size at which a change occurs ($\frac{\tilde{f}}{10} \times T$). Therefore higher values of $\tilde{f}$ imply lower frequencies of change. The ratio of $\tilde{s}$ is used to determine the severity of change for the applicable migration type. Higher values of $\tilde{s}$ imply higher severities of change.

The following sections discuss the sensitivity of the LNNAIS parameters to changes in dimension, cluster size, frequency of change and severity of change in the data for each of the migration types. All experimental results reported in the following sections are collective means taken over 100 time steps ($T = 100$) for 50 runs ($E = 50$) unless stated otherwise. A time step is a single presentation of the data set at that specific point in time. The parameter values for each non-stationary data set were found empirically to deliver the best performance for each of the models.

7.4 Sensitivity of LNNAIS parameters

This section discusses the sensitivity of the LNNAIS parameters for each of the migration types. Parallel coordinates [43, 94] is used to illustrate the effect of changes in the non-stationary environment (e.g. number of dimensions, cluster size, frequency of change and severity of change) on the parameters of LNNAIS.

Parallel coordinates is a visualisation technique to analyse multivariate data. The attribute values of a pattern in an N dimensional data set are each plotted on a parallel line. Each parallel line represents a dimension, therefore an N dimensional data set has N parallel lines which are equally spaced and each data pattern is plotted as a polyline with vertices on the parallel axes. The parallel coordinates visualisation technique was invented by Maurice d'Ocagne [43] and popularised by Alfred Inselberg [94]. Recently, Franken proposed parallel coordinates as an information visualisation technique to show any interdependencies and trends between parameters of a model (if any) [54]. A similar approach is followed in this section.

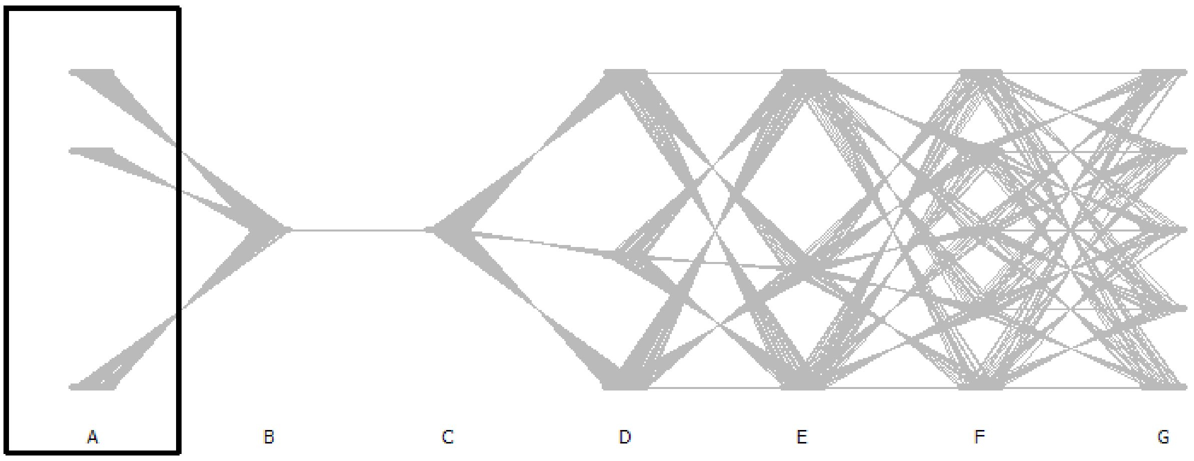
The optimal set of LNNAIS parameter values for each non-stationary data set is plotted with the associated environment parameters which defines the specific non-stationary data set. The LNNAIS parameter values for each data set were found empirically to deliver the best performance. The parallel coordinates plot of the optimal set of LNNAIS parameter values will illustrate the effect of changes in the environment on the optimal set of parameter values of LNNAIS. All parallel coordinates plots in this section consist of axes which are represented by letters A to G. These letters map to the following parameters: $A \mapsto \mathcal{B}_{max}$, $B \mapsto \rho$, $C \mapsto \varepsilon_{clone}$, $D \mapsto N$ (number of dimensions), $E \mapsto |C|$ (cluster size), $F \mapsto \tilde{f}$ (frequency of change) and $G \mapsto \tilde{s}$ (severity of change). The lowest value of each axis in the parallel coordinates plot is at the bottom of the axis. Furthermore, three dimensional plots of the environment parameters versus the clustering quality of LNNAIS illustrate the effect of changes in different environments on the clustering performance of LNNAIS.

7.4.1 Pattern Migration

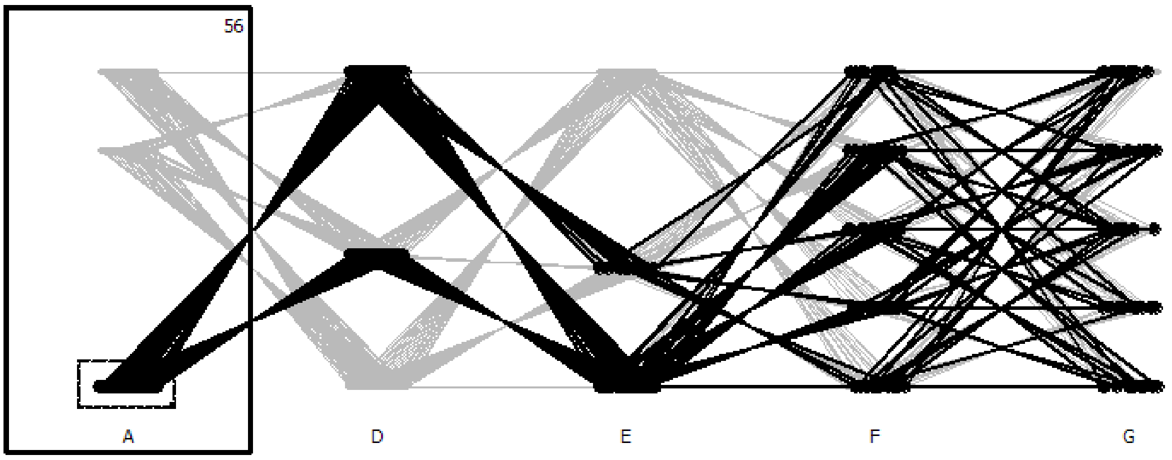
Figure 7.1(a) illustrates the parallel coordinates plot for pattern migration environments. Figure 7.1(a) shows that there is no change in axes B or C for all values of D to G. This means that changes in the pattern migration environment have no effect on the optimal values of ρ and ε_{clone}. These axes are removed in figures 7.1(b) and 7.1(c) to focus more on the effect of changes in the pattern migration environment on parameter $\mathcal{B}_{max}$. The polylines for small values of $\mathcal{B}_{max}$ in figure 7.1(b) and larger values of $\mathcal{B}_{max}$ in figure 7.1(c) are highlighted. The highlighted polylines show that LNNAIS utilises small population sizes for high dimensional environments with small cluster sizes (illustrated by axes D and E in figure 7.1(b)) and larger population sizes at different dimensions for large cluster sizes (illustrated by axes D and E in figure 7.1(c)). Note that there is no effect on $\mathcal{B}_{max}$ with different frequencies or severities of change. Table 7.1 shows that, in general, the clustering quality of LNNAIS is the lowest at high frequencies and high severities of change in pattern migration environments at different dimensions and cluster sizes. The clustering quality of LNNAIS improves with an increase in the cluster size at different dimensions. Increasing the number of dimensions lowers the clustering quality of LNNAIS at different cluster sizes.

7.4.2 Cluster Migration

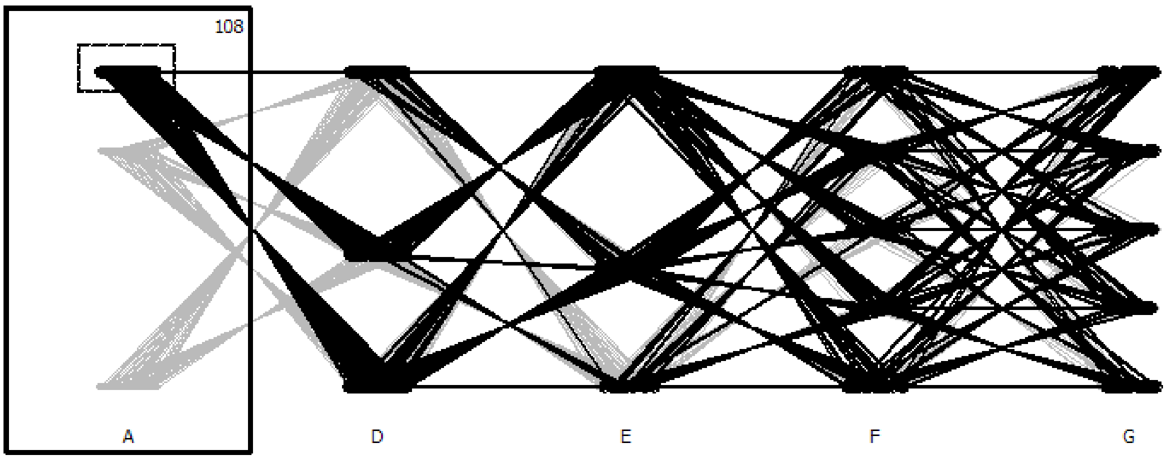
Figure 7.2(a) illustrates the parallel coordinates plot for cluster migration environments. Figures 7.2(b) and 7.2(c) respectively illustrates a similar trend in $\mathcal{B}_{max}$ as for pattern migration environments. The highlighted polylines show that LNNAIS utilises small population sizes for high dimensional cluster migration environments with small cluster sizes (illustrated by axes D and E in figure 7.2(b)) and larger population sizes at different dimensions for large cluster sizes (illustrated by axes D and E in figure 7.2(c)). Note that there is also no effect on $\mathcal{B}_{max}$ with different frequencies or severities of change. Table 7.2 shows similar trends on the clustering performance of LNNAIS for cluster migration environments as for pattern migration environments (as shown in table 7.1). In general, the clustering quality of LNNAIS in cluster migration environments is also the lowest at high frequencies and high severities of change at different dimensions and cluster sizes. The clustering quality of LNNAIS improves with an increase in the cluster size at different dimensions and an increase in the number of dimensions lowers the clustering quality of LNNAIS at different cluster sizes.



(a) All LNNAIS Parameters



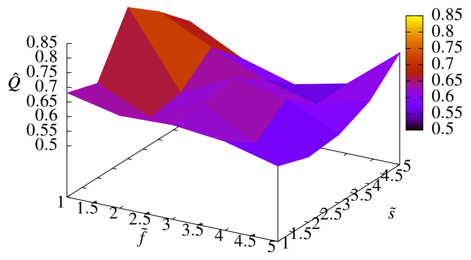
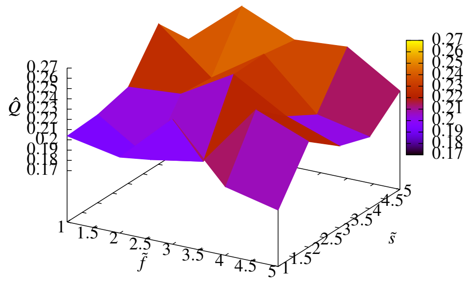
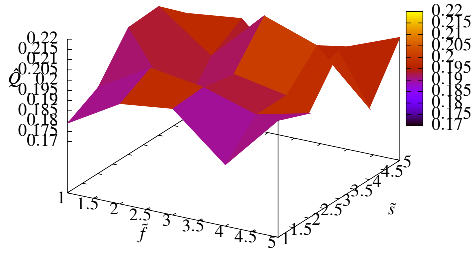
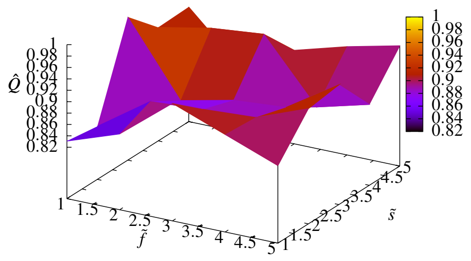
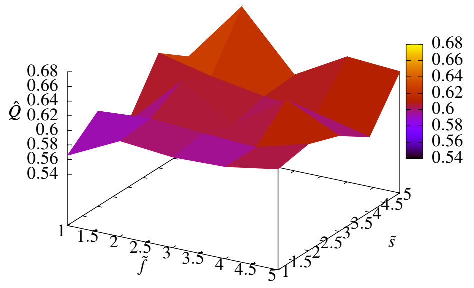
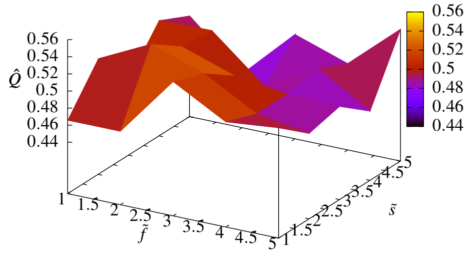
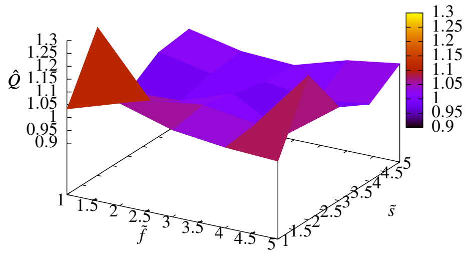
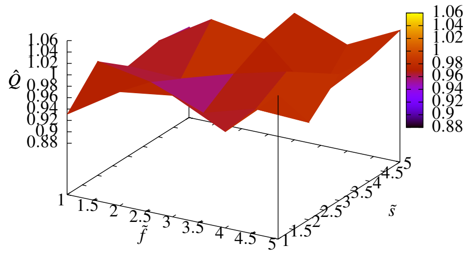
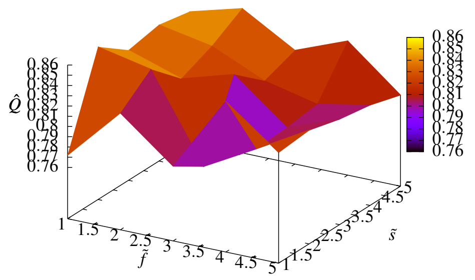
(b) Small $\mathcal{B}_{max}$

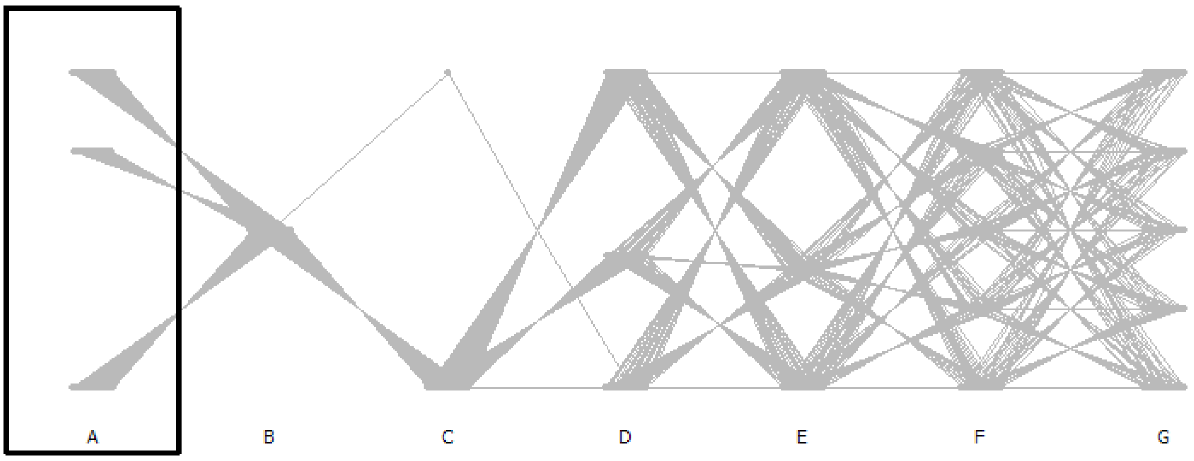


(c) Large $\mathcal{B}_{max}$

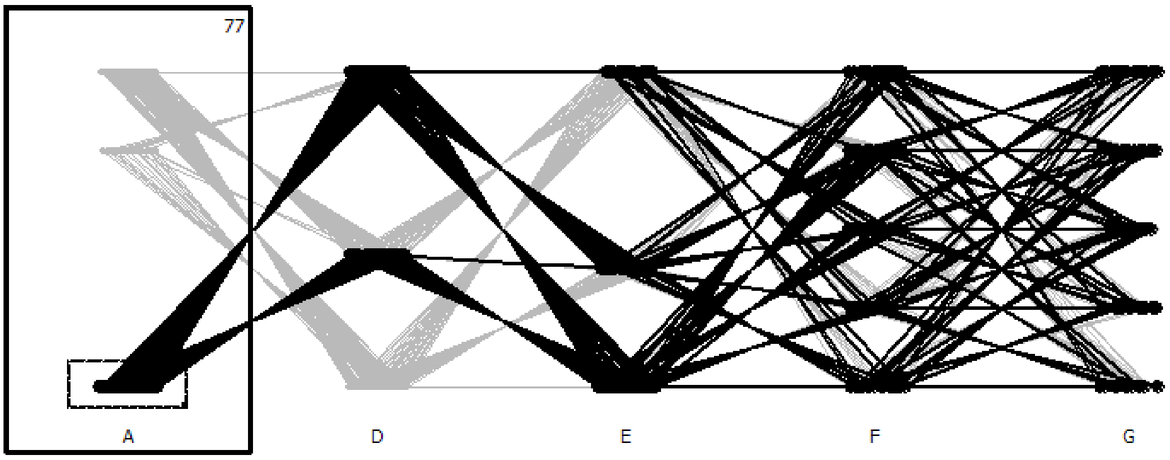
Figure 7.1 Parallel Coordinates of LNNAIS Parameters for Pattern Migration

Table 7.1 Effect of Pattern Migration on Clustering Performance of LNNAIS

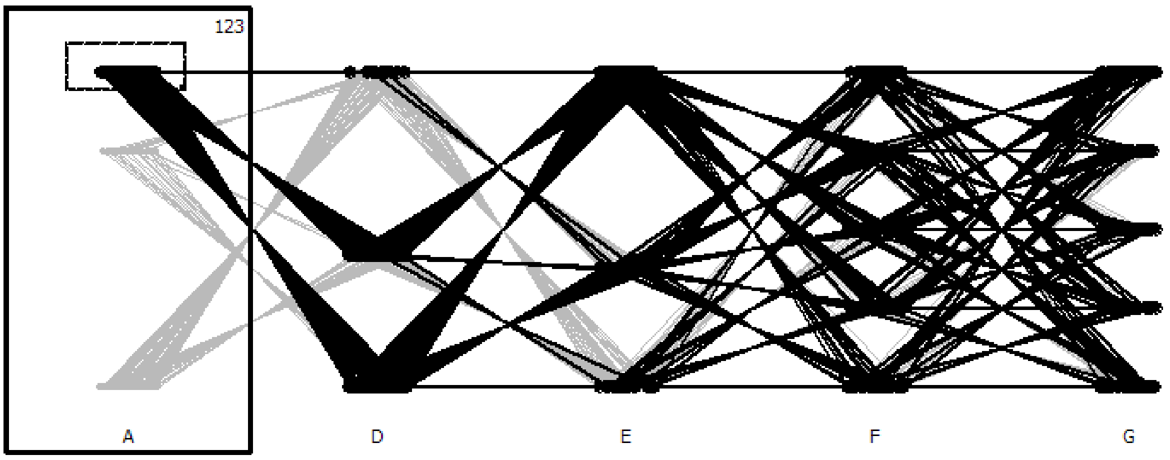
	$\lvert C\rvert = 10$	$\lvert C\rvert = 25$	$\lvert C\rvert = 50$
$N = 3$			
$N = 8$			
$N = 15$			



(a) All LNNAIS Parameters



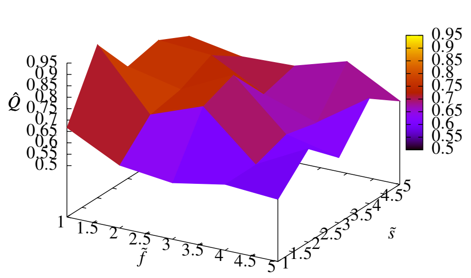
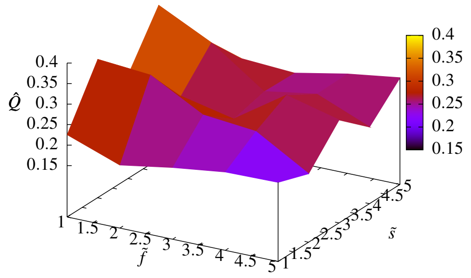
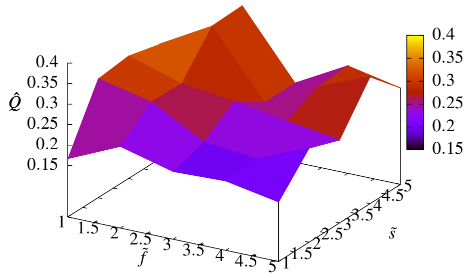
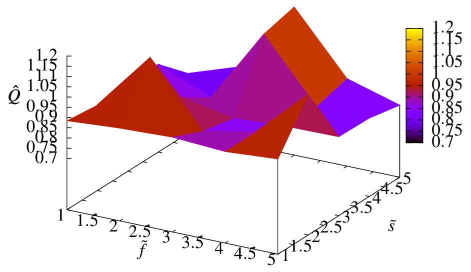
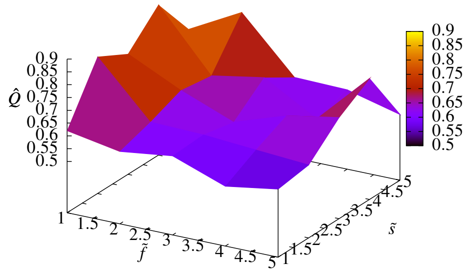
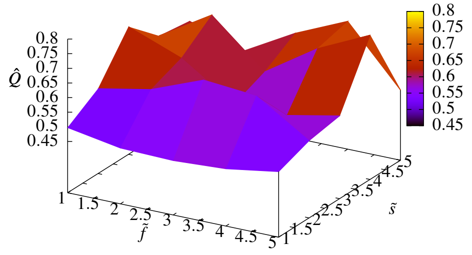
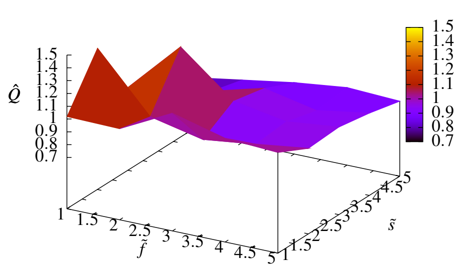
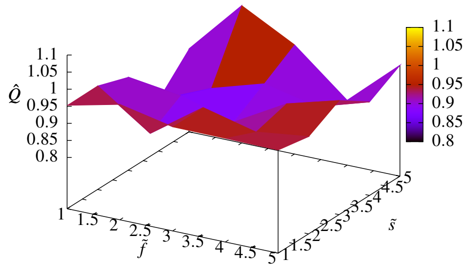
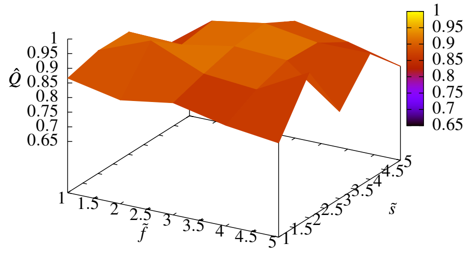
(b) Small $\mathcal{B}_{max}$

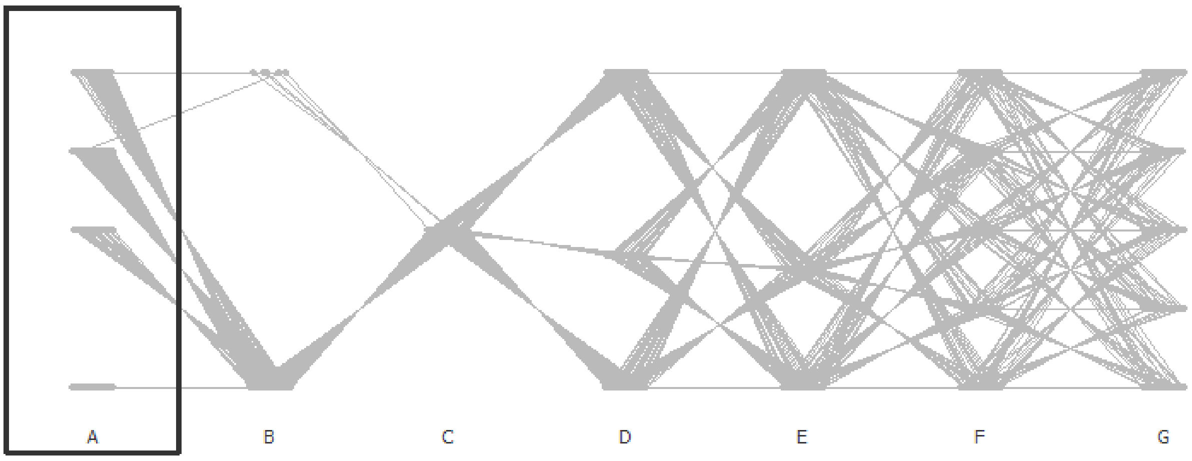


(c) Large $\mathcal{B}_{max}$

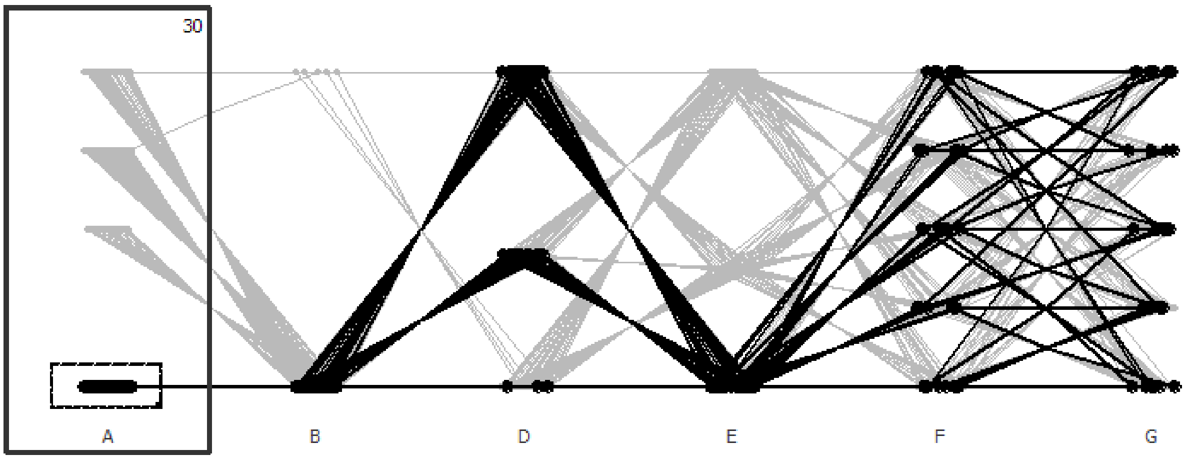
Figure 7.2 Parallel Coordinates of LNNAIS Parameters for Cluster Migration

Table 7.2 Effect of Cluster Migration on Clustering Performance of LNNAIS

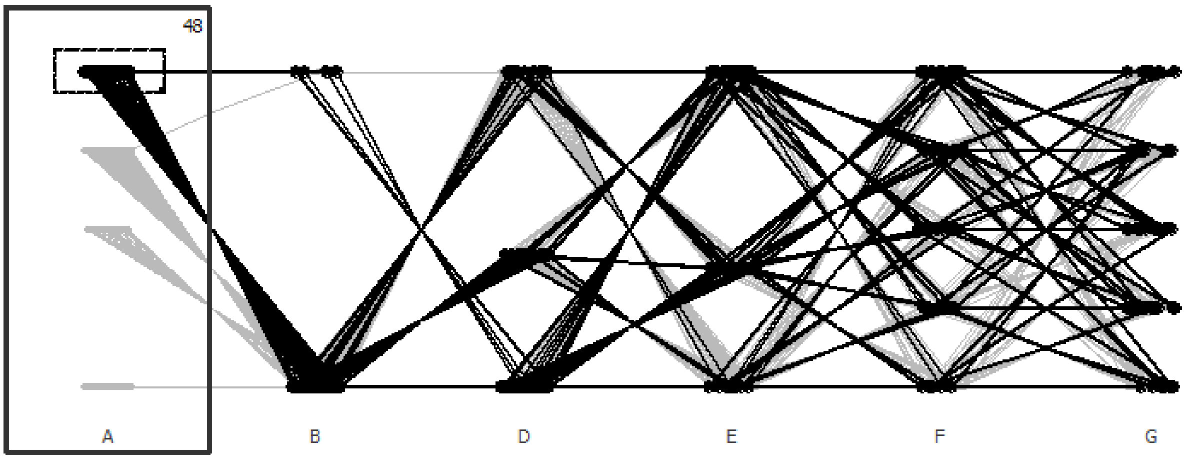
	$\lvert C\rvert = 10$	$\lvert C\rvert = 25$	$\lvert C\rvert = 50$
$N = 3$			
$N = 8$			
$N = 15$			



(a) All LNNAIS Parameters



(b) Small $\mathcal{B}_{max}$



(c) Large $\mathcal{B}_{max}$

Figure 7.3 Parallel Coordinates of LNNAIS Parameters for Centroid Migration

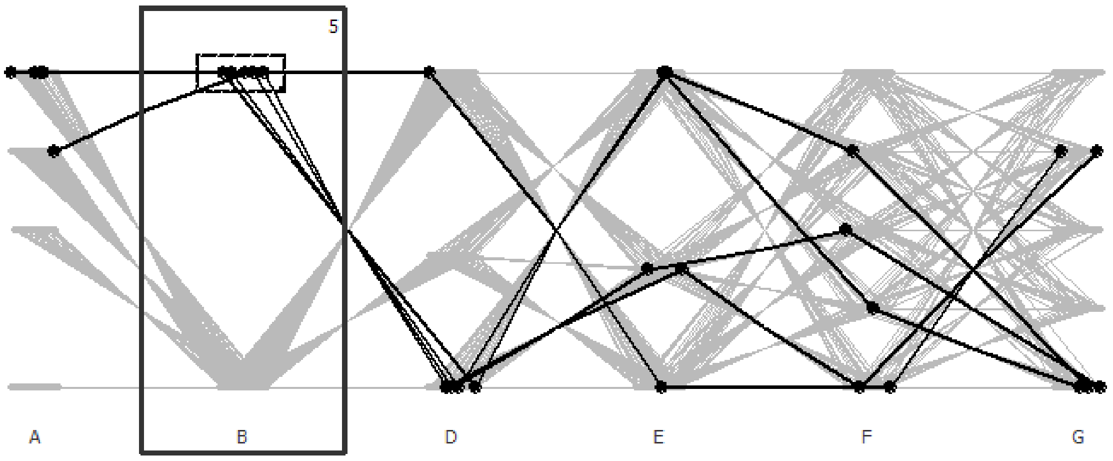
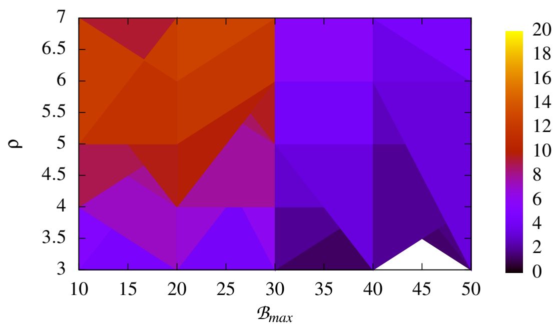


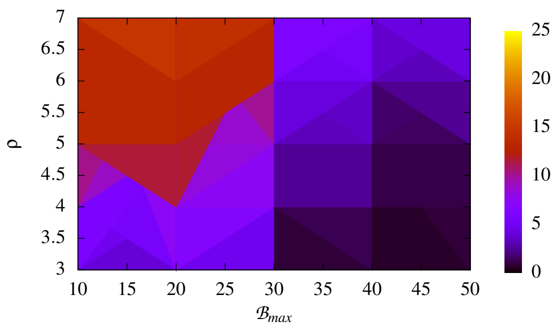
Figure 7.4 Parallel Coordinates of ρ for Centroid Migration

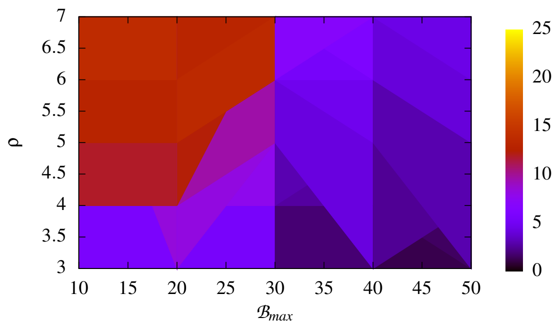
7.4.3 Centroid Migration

Figure 7.3(a) illustrates the parallel coordinates plot for centroid migration environments. Similar to the parallel coordinates plots for pattern and cluster migration environments, there is no effect on the value of ε_{clone} with changes in the centroid migration environment. However, there is an effect on the value of ρ (where the minority of polylines have $\rho = 4$): Figure 7.4 highlights the polylines for $\rho = 4$. The majority of these are for $N = 3$ and are investigated next. Figure 7.5 illustrates the heat maps of the clustering performance of LNNAIS at different values of $\mathcal{B}_{max}$ and ρ for the centroid migration environments which are highlighted in figure 7.4. These heat maps show that there is no distinct difference in the clustering quality of LNNAIS with $\mathcal{B}_{max} = 50$ and $\rho \leq 5$. Therefore these polylines can be seen as outliers to the norm of $\rho = 3$.

Figures 7.3(b) and 7.3(c) respectively illustrate a similar trend in $\mathcal{B}_{max}$ as for pattern and cluster migration environments. The highlighted polylines show that LNNAIS utilises small population sizes with small cluster sizes (illustrated by axis E in figure 7.3(b)) and larger population sizes for large cluster sizes (illustrated by axis E in figure 7.3(c)). Different to the pattern and cluster migration environments, LNNAIS utilises small and large population sizes at different dimensions. Again note that there is also no effect on $\mathcal{B}_{max}$ with different frequencies or severities of change. Similar trends on the clustering performance of LNNAIS for pattern and cluster migration environments are shown in table 7.3 for centroid migration environments. The clustering quality of LNNAIS in centroid migration environments is the lowest at high frequencies and high severities of change at different dimensions and cluster sizes, the clustering quality of LNNAIS improves with an increase in the cluster size at different dimensions, and an increase in the number of di-


(a) $|C| = 25$, $\tilde{f} = 1$, $\tilde{s} = 4$


(b) $|C| = 50$, $\tilde{f} = 2$, $\tilde{s} = 1$


(c) $|C| = 25$, $\tilde{f} = 3$, $\tilde{s} = 1$

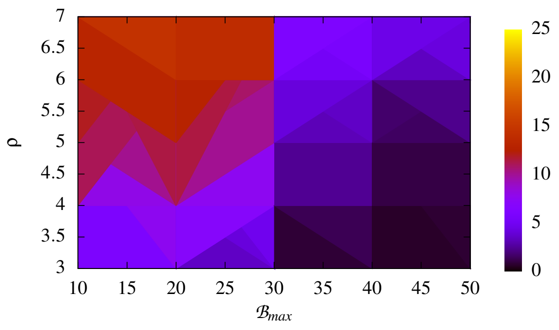
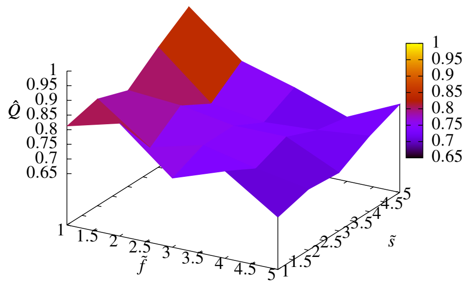
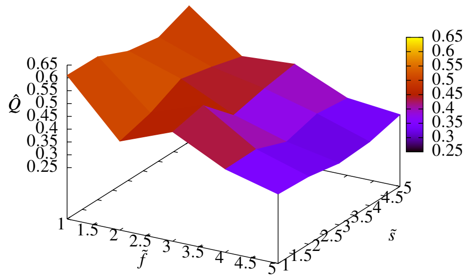
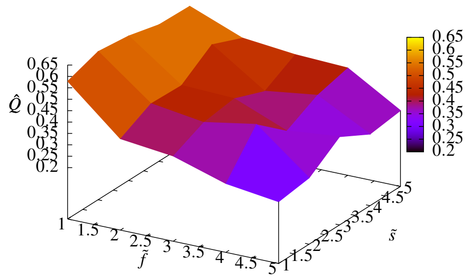
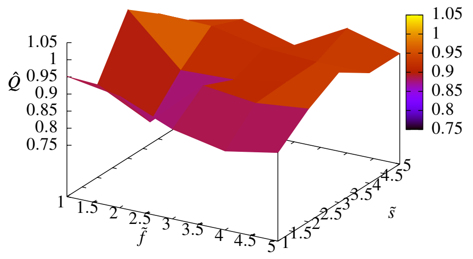
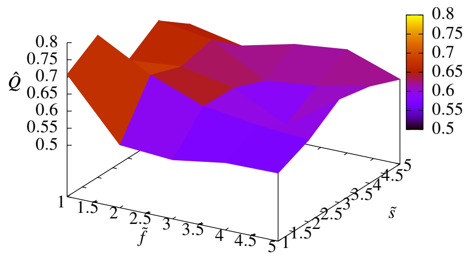
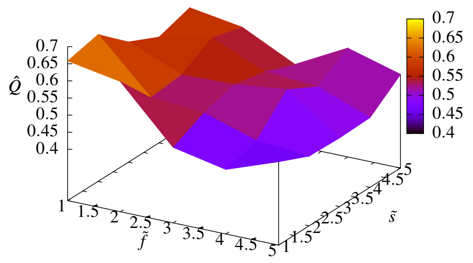
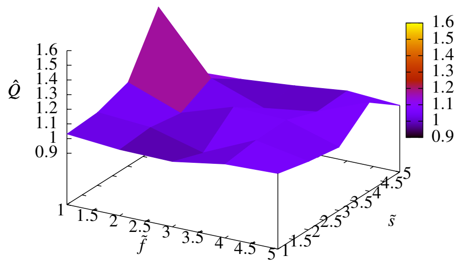
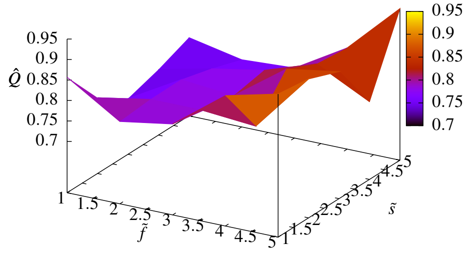
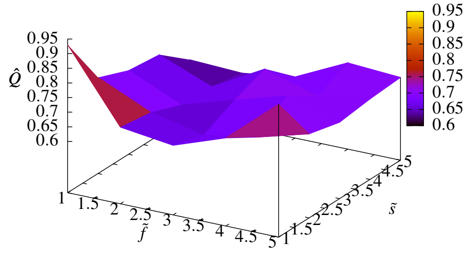

(d) $|C| = 50$, $\tilde{f} = 4$, $\tilde{s} = 1$

Figure 7.5 Heat Maps of LNNAIS Parameters for Centroid Migration ($N = 3$)

Table 7.3 Effect of Centroid Migration on Clustering Performance of LNNAIS

	$\lvert C\rvert = 10$	$\lvert C\rvert = 25$	$\lvert C\rvert = 50$
$N = 3$			
$N = 8$			
$N = 15$			

mensions lowers the clustering quality of LNNAIS at different cluster sizes. Note that different to the pattern and cluster migration environments, at high dimensions in the centroid migration environments, the function of frequency versus severity versus clustering quality tends to flatten at different cluster sizes. This shows that the frequency and severity of change in high dimensional centroid migration environments have a smaller effect on the clustering performance of LNNAIS.

7.5 Experimental Results

This section compares the clustering performance of LNNAIS with the clustering performance of LNN$_{SDOT}$, SMAIN and DWB for each migration type.

The clustering quality of a model for a specific run can be measured using $\bar{Q}_i$ if the data is non-stationary (as defined in equation (7.1)). This section investigates whether there is a difference between the mean clustering quality, $\bar{Q}$, of two models for a specific non-stationary data set or not. The hypothesis can therefore be defined as

- *Null* hypothesis, H_0: There is no difference in $\bar{Q}$.
- *Alternative* hypothesis, H_1: There is a difference in $\bar{Q}$.

A non-parametric Mann-Whitney U test with a 0.95 confidence interval ($\alpha = 0.05$) was used to test the above hypothesis. Furthermore, the clustering quality (Q_{ratio}, J_{intra}, J_{inter} and K) is plotted against time to quantify the quality of partitioning of the non-stationary environment by each clustering algorithm over time.

Due to the prohibitively large number of generated non-stationary data sets, a representative configuration of environment parameter values was selected to compare the clustering performance of the different models. The environment parameters were set to $N = 8$, $|C| = 25$, $\tilde{f} = 3$ and $\tilde{s} = 3$. The value of each parameter in the selected configuration is then changed while the remaining parameter values are kept constant. Therefore each migration type has five data sets representing different severities of change (with the remaining environment parameters kept constant), five data sets representing different frequencies of change, three data sets representing different dimensions and three data sets representing different cluster sizes. This gives a total of sixteen different non-stationary data sets for each migration type. The results for each of these

Table 7.4 LNNAIS Parameter Values - Pattern Migration

Environment parameters	$\mathcal{B}_{max}$	ρ	ε_{clone}
$N=3$; $\|C\|=25$; $\tilde{f}=3$; $\tilde{s}=3$	50	3	5
$N=8$; $\|C\|=25$; $\tilde{f}=3$; $\tilde{s}=3$	50	3	5
$N=15$; $\|C\|=25$; $\tilde{f}=3$; $\tilde{s}=3$	50	3	5
$N=8$; $\|C\|=10$; $\tilde{f}=3$; $\tilde{s}=3$	10	3	5
$N=8$; $\|C\|=50$; $\tilde{f}=3$; $\tilde{s}=3$	40	3	5
$N=8$; $\|C\|=25$; $\tilde{f}=1$; $\tilde{s}=3$	50	3	5
$N=8$; $\|C\|=25$; $\tilde{f}=2$; $\tilde{s}=3$	40	3	5
$N=8$; $\|C\|=25$; $\tilde{f}=4$; $\tilde{s}=3$	50	3	5
$N=8$; $\|C\|=25$; $\tilde{f}=5$; $\tilde{s}=3$	40	3	5
$N=8$; $\|C\|=25$; $\tilde{f}=3$; $\tilde{s}=1$	50	3	5
$N=8$; $\|C\|=25$; $\tilde{f}=3$; $\tilde{s}=2$	50	3	5
$N=8$; $\|C\|=25$; $\tilde{f}=3$; $\tilde{s}=4$	50	3	5
$N=8$; $\|C\|=25$; $\tilde{f}=3$; $\tilde{s}=5$	50	3	5

migration types are discussed next. The parameter values used by each model for each of the migration types are summarised in tables 7.4 - 7.12 for LNNAIS, DWB and SMAIN, respectively.

7.5.1 Pattern Migration

Figure 7.6 illustrates the quality of partitioning by the different models over time for pattern migration. Note the increase in the ALC population size for the SMAIN model with every change in the data (figure 7.6(d) at $t = 30$, $t = 60$ and $t = 90$). Figure 7.6(a) shows that LNNAIS initially finds clusters with a lower quality (high Q_{ratio} value) when compared to the other models. As time progresses, the quality of the clusters found by LNNAIS improves (as illustrated in figures 7.6(b), 7.6(c) and 7.6(e) the number of clusters found increases, becomes more compact and separated). LNNAIS delivers clusters of a higher quality than DWB and LNN$_{SDOT}$ for all pattern migration environments. Figure 7.6(f) illustrates the average number of clusters and the number of clusters with the highest frequency which was dynamically determined by LNN$_{SDOT}$. The number of clusters detected with the highest frequency was $K = 3$ at every point in time with an average number of clusters between $K = 3$ and $K = 3.6$. Although LNN$_{SDOT}$ was unable to detect the correct number of clusters ($K = 8$), figure 7.6(f) shows that there was no change in the number of clusters over time, which is expected for pattern migration where data patterns randomly migrate between a static number of clusters. Section 7.5.3 discusses the argument for

Table 7.5 LNNAIS Parameter Values - Cluster Migration

Environment parameters	$\mathcal{B}_{max}$	ρ	ε_{clone}
$N = 3$; $\|C\| = 25$; $\tilde{f} = 3$; $\tilde{s} = 3$	50	3	5
$N = 8$; $\|C\| = 25$; $\tilde{f} = 3$; $\tilde{s} = 3$	50	3	5
$N = 15$; $\|C\| = 25$; $\tilde{f} = 3$; $\tilde{s} = 3$	10	3	5
$N = 8$; $\|C\| = 10$; $\tilde{f} = 3$; $\tilde{s} = 3$	10	3	5
$N = 8$; $\|C\| = 50$; $\tilde{f} = 3$; $\tilde{s} = 3$	50	3	5
$N = 8$; $\|C\| = 25$; $\tilde{f} = 1$; $\tilde{s} = 3$	40	3	5
$N = 8$; $\|C\| = 25$; $\tilde{f} = 2$; $\tilde{s} = 3$	50	3	5
$N = 8$; $\|C\| = 25$; $\tilde{f} = 4$; $\tilde{s} = 3$	50	3	5
$N = 8$; $\|C\| = 25$; $\tilde{f} = 5$; $\tilde{s} = 3$	50	3	5
$N = 8$; $\|C\| = 25$; $\tilde{f} = 3$; $\tilde{s} = 1$	50	3	5
$N = 8$; $\|C\| = 25$; $\tilde{f} = 3$; $\tilde{s} = 2$	50	3	5
$N = 8$; $\|C\| = 25$; $\tilde{f} = 3$; $\tilde{s} = 4$	50	3	5
$N = 8$; $\|C\| = 25$; $\tilde{f} = 3$; $\tilde{s} = 5$	50	3	5

Table 7.6 LNNAIS Parameter Values - Centroid Migration

Environment parameters	$\mathcal{B}_{max}$	ρ	ε_{clone}
$N = 3$; $\|C\| = 25$; $\tilde{f} = 3$; $\tilde{s} = 3$	50	3	5
$N = 8$; $\|C\| = 25$; $\tilde{f} = 3$; $\tilde{s} = 3$	50	3	5
$N = 15$; $\|C\| = 25$; $\tilde{f} = 3$; $\tilde{s} = 3$	30	3	5
$N = 8$; $\|C\| = 10$; $\tilde{f} = 3$; $\tilde{s} = 3$	10	3	5
$N = 8$; $\|C\| = 50$; $\tilde{f} = 3$; $\tilde{s} = 3$	40	3	5
$N = 8$; $\|C\| = 25$; $\tilde{f} = 1$; $\tilde{s} = 3$	30	3	5
$N = 8$; $\|C\| = 25$; $\tilde{f} = 2$; $\tilde{s} = 3$	30	3	5
$N = 8$; $\|C\| = 25$; $\tilde{f} = 4$; $\tilde{s} = 3$	40	3	5
$N = 8$; $\|C\| = 25$; $\tilde{f} = 5$; $\tilde{s} = 3$	40	3	5
$N = 8$; $\|C\| = 25$; $\tilde{f} = 3$; $\tilde{s} = 1$	40	3	5
$N = 8$; $\|C\| = 25$; $\tilde{f} = 3$; $\tilde{s} = 2$	40	3	5
$N = 8$; $\|C\| = 25$; $\tilde{f} = 3$; $\tilde{s} = 4$	40	3	5
$N = 8$; $\|C\| = 25$; $\tilde{f} = 3$; $\tilde{s} = 5$	40	3	5

Table 7.7 DWB Parameter Values - Pattern Migration

Environment parameters	$\mathcal{B}_{max}$	ϕ_{init}	m_{min}	A	a_{min}	a_{max}	k_{clone}	ς	τ	τ_α	τ_β	$k_{compress}$
$N = 3;\ \lvert C\rvert = 25;\ \tilde{f} = 3;\ \tilde{s} = 3$	84	0.705	0.114	55	6	6	6	0.93	8	8	8	6
$N = 8;\ \lvert C\rvert = 25;\ \tilde{f} = 3;\ \tilde{s} = 3$	34	0.184	0.634	91	23	41	12	0.297	9	3	6	6
$N = 15;\ \lvert C\rvert = 25;\ \tilde{f} = 3;\ \tilde{s} = 3$	29	0.262	0.43	73	74	74	9	0.571	10	4	6	5
$N = 8;\ \lvert C\rvert = 10;\ \tilde{f} = 3;\ \tilde{s} = 3$	43	0.213	0.213	38	63	63	10	0.888	7	2	7	4
$N = 8;\ \lvert C\rvert = 50;\ \tilde{f} = 3;\ \tilde{s} = 3$	44	0.648	0.395	12	24	24	7	0.536	7	1	5	3
$N = 8;\ \lvert C\rvert = 25;\ \tilde{f} = 1;\ \tilde{s} = 3$	32	0.986	0.733	74	37	37	2	0.423	8	9	6	1
$N = 8;\ \lvert C\rvert = 25;\ \tilde{f} = 2;\ \tilde{s} = 3$	40	0.149	0.543	85	13	13	4	0.459	4	3	3	6
$N = 8;\ \lvert C\rvert = 25;\ \tilde{f} = 4;\ \tilde{s} = 3$	44	0.648	0.395	12	24	24	7	0.536	7	1	5	3
$N = 8;\ \lvert C\rvert = 25;\ \tilde{f} = 5;\ \tilde{s} = 3$	37	0.402	0.234	95	17	71	4	0.655	5	8	9	1
$N = 8;\ \lvert C\rvert = 25;\ \tilde{f} = 3;\ \tilde{s} = 1$	34	0.184	0.634	91	23	41	12	0.297	9	3	6	6
$N = 8;\ \lvert C\rvert = 25;\ \tilde{f} = 3;\ \tilde{s} = 2$	37	0.402	0.234	95	17	71	4	0.655	5	8	9	1
$N = 8;\ \lvert C\rvert = 25;\ \tilde{f} = 3;\ \tilde{s} = 4$	41	0.508	0.873	15	15	15	13	0.62	5	7	6	6
$N = 8;\ \lvert C\rvert = 25;\ \tilde{f} = 3;\ \tilde{s} = 5$	37	0.606	0.944	44	7	7	2	0.944	4	5	6	4

Table 7.8 DWB Parameter Values - Cluster Migration

Environment parameters	$\mathcal{B}_{max}$	ϕ_{init}	m_{min}	A	a_{min}	a_{max}	k_{clone}	ς	τ	τ_α	τ_β	$k_{compress}$
$N=3$; $\|C\|=25$; $\tilde{f}=3$; $\tilde{s}=3$	95	0.74	0.346	82	30	34	13	0.318	6	9	6	7
$N=8$; $\|C\|=25$; $\tilde{f}=3$; $\tilde{s}=3$	27	0.81	0.501	93	47	54	5	0.895	8	8	6	2
$N=15$; $\|C\|=25$; $\tilde{f}=3$; $\tilde{s}=3$	31	0.325	0.775	75	26	75	12	0.775	8	8	3	3
$N=8$; $\|C\|=10$; $\tilde{f}=3$; $\tilde{s}=3$	35	0.677	0.93	58	77	95	1	0.677	10	2	4	6
$N=8$; $\|C\|=50$; $\tilde{f}=3$; $\tilde{s}=3$	31	0.325	0.775	75	26	75	12	0.775	8	8	3	3
$N=8$; $\|C\|=25$; $\tilde{f}=1$; $\tilde{s}=3$	29	0.262	0.43	73	74	74	9	0.571	10	4	6	5
$N=8$; $\|C\|=25$; $\tilde{f}=2$; $\tilde{s}=3$	31	0.325	0.775	75	26	75	12	0.775	8	8	3	3
$N=8$; $\|C\|=25$; $\tilde{f}=4$; $\tilde{s}=3$	32	0.986	0.733	74	37	37	2	0.423	8	9	6	1
$N=8$; $\|C\|=25$; $\tilde{f}=5$; $\tilde{s}=3$	27	0.81	0.501	93	47	54	5	0.895	8	8	6	2
$N=8$; $\|C\|=25$; $\tilde{f}=3$; $\tilde{s}=1$	44	0.648	0.395	12	24	24	7	0.536	7	1	5	3
$N=8$; $\|C\|=25$; $\tilde{f}=3$; $\tilde{s}=2$	36	0.445	0.304	10	33	33	3	0.81	9	9	2	6
$N=8$; $\|C\|=25$; $\tilde{f}=3$; $\tilde{s}=4$	36	0.445	0.304	10	33	33	3	0.81	9	9	2	6
$N=8$; $\|C\|=25$; $\tilde{f}=3$; $\tilde{s}=5$	39	0.923	0.388	56	17	17	14	0.107	5	7	2	4

Table 7.9 DWB Parameter Values - Centroid Migration

Environment parameters	$\mathcal{B}_{max}$	ϕ_{init}	m_{min}	A	a_{min}	a_{max}	k_{clone}	ς	τ	τ_α	τ_β	$k_{compress}$
$N = 3$; $\lvert C\rvert = 25$; $\tilde{f} = 3$; $\tilde{s} = 3$	95	0.74	0.346	82	30	34	13	0.318	6	9	6	7
$N = 8$; $\lvert C\rvert = 25$; $\tilde{f} = 3$; $\tilde{s} = 3$	53	0.838	0.923	3	31	88	9	0.866	3	6	2	7
$N = 15$; $\lvert C\rvert = 25$; $\tilde{f} = 3$; $\tilde{s} = 3$	68	0.747	0.297	4	10	29	14	0.409	8	6	7	5
$N = 8$; $\lvert C\rvert = 10$; $\tilde{f} = 3$; $\tilde{s} = 3$	53	0.17	0.311	52	27	27	11	0.733	1	4	5	7
$N = 8$; $\lvert C\rvert = 50$; $\tilde{f} = 3$; $\tilde{s} = 3$	53	0.838	0.923	3	31	88	9	0.866	3	6	2	7
$N = 8$; $\lvert C\rvert = 25$; $\tilde{f} = 1$; $\tilde{s} = 3$	95	0.74	0.346	82	30	34	13	0.318	6	9	6	7
$N = 8$; $\lvert C\rvert = 25$; $\tilde{f} = 2$; $\tilde{s} = 3$	53	0.838	0.923	3	31	88	9	0.866	3	6	2	7
$N = 8$; $\lvert C\rvert = 25$; $\tilde{f} = 4$; $\tilde{s} = 3$	40	0.149	0.543	85	13	13	4	0.459	4	3	3	6
$N = 8$; $\lvert C\rvert = 25$; $\tilde{f} = 5$; $\tilde{s} = 3$	68	0.747	0.297	4	10	29	14	0.409	8	6	7	5
$N = 8$; $\lvert C\rvert = 25$; $\tilde{f} = 3$; $\tilde{s} = 1$	46	0.768	0.712	79	68	68	8	0.515	7	2	1	7
$N = 8$; $\lvert C\rvert = 25$; $\tilde{f} = 3$; $\tilde{s} = 2$	69	0.543	0.937	54	14	93	12	0.29	9	4	8	3
$N = 8$; $\lvert C\rvert = 25$; $\tilde{f} = 3$; $\tilde{s} = 4$	69	0.543	0.937	54	14	93	12	0.29	9	4	8	3
$N = 8$; $\lvert C\rvert = 25$; $\tilde{f} = 3$; $\tilde{s} = 5$	55	0.311	0.958	98	12	12	5	0.198	6	7	3	7

Table 7.10 SMAIN Parameter Values - Pattern Migration

Environment parameters	$\mathcal{B}_{init}$	R_{γ}	R_{Λ}	NAT	R_k	R_{max}	R_{init}
$N = 3$; $\lvert C\rvert = 25$; $\tilde{f} = 3$; $\tilde{s} = 3$	0.042	0.824	22	1.445	0.852	819	31
$N = 8$; $\lvert C\rvert = 25$; $\tilde{f} = 3$; $\tilde{s} = 3$	0.057	0.73	3	3.238	0.539	269	62
$N = 15$; $\lvert C\rvert = 25$; $\tilde{f} = 3$; $\tilde{s} = 3$	0.122	0.828	29	4.872	0.531	875	47
$N = 8$; $\lvert C\rvert = 10$; $\tilde{f} = 3$; $\tilde{s} = 3$	0.198	0.797	10	2.566	0.719	925	29
$N = 8$; $\lvert C\rvert = 50$; $\tilde{f} = 3$; $\tilde{s} = 3$	0.147	0.793	40	3.751	0.914	969	50
$N = 8$; $\lvert C\rvert = 25$; $\tilde{f} = 1$; $\tilde{s} = 3$	0.074	0.992	71	3.911	0.734	488	72
$N = 8$; $\lvert C\rvert = 25$; $\tilde{f} = 2$; $\tilde{s} = 3$	0.098	0.848	5	3.559	0.68	606	98
$N = 8$; $\lvert C\rvert = 25$; $\tilde{f} = 4$; $\tilde{s} = 3$	0.115	0.969	7	4.231	0.938	550	44
$N = 8$; $\lvert C\rvert = 25$; $\tilde{f} = 5$; $\tilde{s} = 3$	0.241	0.52	20	3.687	0.711	831	21
$N = 8$; $\lvert C\rvert = 25$; $\tilde{f} = 3$; $\tilde{s} = 1$	0.216	0.977	37	3.527	0.078	713	38
$N = 8$; $\lvert C\rvert = 25$; $\tilde{f} = 3$; $\tilde{s} = 2$	0.203	0.941	73	3.302	0.492	856	74
$N = 8$; $\lvert C\rvert = 25$; $\tilde{f} = 3$; $\tilde{s} = 4$	0.154	0.902	25	3.366	0.57	494	47
$N = 8$; $\lvert C\rvert = 25$; $\tilde{f} = 3$; $\tilde{s} = 5$	0.154	0.902	25	3.366	0.57	494	47

Table 7.11 SMAIN Parameter Values - Cluster Migration

Environment parameters	$\mathcal{B}_{init}$	R_{γ}	R_{Λ}	NAT	R_k	R_{max}	R_{init}
$N = 3$; $\lvert C\rvert = 25$; $\tilde{f} = 3$; $\tilde{s} = 3$	0.226	0.926	2	1.38	0.898	281	76
$N = 8$; $\lvert C\rvert = 25$; $\tilde{f} = 3$; $\tilde{s} = 3$	0.057	0.73	3	3.238	0.539	269	62
$N = 15$; $\lvert C\rvert = 25$; $\tilde{f} = 3$; $\tilde{s} = 3$	0.244	0.965	13	4.904	0.195	994	84
$N = 8$; $\lvert C\rvert = 10$; $\tilde{f} = 3$; $\tilde{s} = 3$	0.053	0.629	11	2.79	0.367	956	54
$N = 8$; $\lvert C\rvert = 50$; $\tilde{f} = 3$; $\tilde{s} = 3$	0.182	0.703	4	3.847	0.281	675	23
$N = 8$; $\lvert C\rvert = 25$; $\tilde{f} = 1$; $\tilde{s} = 3$	0.147	0.793	40	3.751	0.914	969	50
$N = 8$; $\lvert C\rvert = 25$; $\tilde{f} = 2$; $\tilde{s} = 3$	0.216	0.977	37	3.527	0.078	713	38
$N = 8$; $\lvert C\rvert = 25$; $\tilde{f} = 4$; $\tilde{s} = 3$	0.091	0.957	33	3.174	0.836	331	95
$N = 8$; $\lvert C\rvert = 25$; $\tilde{f} = 5$; $\tilde{s} = 3$	0.203	0.941	73	3.302	0.492	856	74
$N = 8$; $\lvert C\rvert = 25$; $\tilde{f} = 3$; $\tilde{s} = 1$	0.203	0.941	73	3.302	0.492	856	74
$N = 8$; $\lvert C\rvert = 25$; $\tilde{f} = 3$; $\tilde{s} = 2$	0.057	0.73	3	3.238	0.539	269	62
$N = 8$; $\lvert C\rvert = 25$; $\tilde{f} = 3$; $\tilde{s} = 4$	0.098	0.848	5	3.559	0.68	606	98
$N = 8$; $\lvert C\rvert = 25$; $\tilde{f} = 3$; $\tilde{s} = 5$	0.147	0.793	40	3.751	0.914	969	50

Table 7.12 SMAIN Parameter Values - Centroid Migration

Environment parameters	$\mathcal{B}_{init}$	R_γ	R_Λ	NAT	R_k	R_{max}	R_{init}
$N = 3$; $\|C\| = 25$; $\tilde{f} = 3$; $\tilde{s} = 3$	0.021	0.945	18	2.245	0.516	363	77
$N = 8$; $\|C\| = 25$; $\tilde{f} = 3$; $\tilde{s} = 3$	0.036	0.602	12	4.552	0.828	913	13
$N = 15$; $\|C\| = 25$; $\tilde{f} = 3$; $\tilde{s} = 3$	0.22	0.812	63	4.488	0.125	300	64
$N = 8$; $\|C\| = 10$; $\tilde{f} = 3$; $\tilde{s} = 3$	0.115	0.969	7	4.231	0.938	550	44
$N = 8$; $\|C\| = 50$; $\tilde{f} = 3$; $\tilde{s} = 3$	0.036	0.602	12	4.552	0.828	913	13
$N = 8$; $\|C\| = 25$; $\tilde{f} = 1$; $\tilde{s} = 3$	0.151	0.582	8	4.199	0.086	531	70
$N = 8$; $\|C\| = 25$; $\tilde{f} = 2$; $\tilde{s} = 3$	0.158	0.723	30	4.584	0.43	806	73
$N = 8$; $\|C\| = 25$; $\tilde{f} = 4$; $\tilde{s} = 3$	0.158	0.723	30	4.584	0.43	806	73
$N = 8$; $\|C\| = 25$; $\tilde{f} = 5$; $\tilde{s} = 3$	0.115	0.969	7	4.231	0.938	550	44
$N = 8$; $\|C\| = 25$; $\tilde{f} = 3$; $\tilde{s} = 1$	0.036	0.602	12	4.552	0.828	913	13
$N = 8$; $\|C\| = 25$; $\tilde{f} = 3$; $\tilde{s} = 2$	0.158	0.723	30	4.584	0.43	806	73
$N = 8$; $\|C\| = 25$; $\tilde{f} = 3$; $\tilde{s} = 4$	0.244	0.965	13	4.904	0.195	994	84
$N = 8$; $\|C\| = 25$; $\tilde{f} = 3$; $\tilde{s} = 5$	0.212	0.641	16	4.359	0.906	375	60

LNN$_{SDOT}$ not being able to detect the correct number of clusters for pattern and cluster migration data sets. Overall, there is no significant change in Q_{ratio}, J_{intra}, J_{inter} and K for all the models ($t > 40$) in pattern migration environments.

The Mann-Whitney U statistical hypothesis test rejects H_0 that the mean clustering quality, $\bar{Q}$, are the same at a 0.05 level of significance between LNNAIS and LNN$_{SDOT}$, SMAIN and DWB for different dimensions (as summarised in table 7.13), different clusters sizes (as summarised in table 7.14), for all frequencies of change (as summarised in table 7.15) and all severities of change (as summarised in table 7.16). There is thus a statistical significant difference in the clustering quality of all the pattern migration data sets between LNNAIS and all the other models.

Table 7.13: Statistical Hypothesis Testing between LNNAIS and Other Models ($\alpha = 0.05$; with continuity correction; unpaired; non-directional) for Pattern migration at different dimensions ($|C|$=25; $\tilde{f}$=3; $\tilde{s}$=3)

N	LNN$_{SDOT}$	DWB	SMAIN
3	$z = 3.999$	$z = 6.646$	$z = 6.646$
3	$p < 0.001$	$p < 0.001$	$p < 0.001$
3	Reject H_0	Reject H_0	Reject H_0
8	$z = 6.646$	$z = 6.446$	$z = 6.402$

Continued on next page

N	LNN$_{SDOT}$	DWB	SMAIN
8	$p < 0.001$	$p < 0.001$	$p < 0.001$
8	Reject H_0	Reject H_0	Reject H_0
15	$z = 6.646$	$z = 5.988$	$z = 6.646$
15	$p < 0.001$	$p < 0.001$	$p < 0.001$
15	Reject H_0	Reject H_0	Reject H_0

Table 7.14: Statistical Hypothesis Testing between LNNAIS and Other Models ($\alpha = 0.05$; with continuity correction; unpaired; non-directional) for Pattern migration at different cluster sizes (N=8; $\tilde{s}$=3; $\tilde{f}$=3)

$\|C\|$	LNN$_{SDOT}$	DWB	SMAIN
	$z = 6.084$	$z = 6.646$	$z = 6.646$
10	$p < 0.001$	$p < 0.001$	$p < 0.001$
	Reject H_0	Reject H_0	Reject H_0
	$z = 6.646$	$z = 6.446$	$z = 6.402$
25	$p < 0.001$	$p < 0.001$	$p < 0.001$
	Reject H_0	Reject H_0	Reject H_0
	$z = 6.542$	$z = 6.646$	$z = 6.224$
50	$p < 0.001$	$p < 0.001$	$p < 0.001$
	Reject H_0	Reject H_0	Reject H_0

Table 7.15: Statistical Hypothesis Testing between LNNAIS and Other Models ($\alpha = 0.05$; with continuity correction; unpaired; non-directional) for Pattern migration at different frequencies of change (N=8; $|C|$=25; $\tilde{s}$=3)

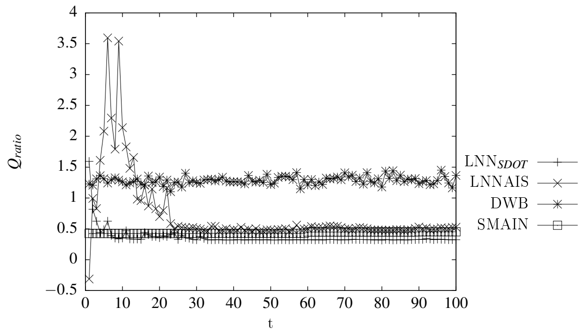
$\tilde{f}$	LNN$_{SDOT}$	DWB	SMAIN
	$z = 6.557$	$z = 6.646$	$z = 6.572$
1	$p < 0.001$	$p < 0.001$	$p < 0.001$
	Reject H_0	Reject H_0	Reject H_0
	$z = 6.246$	$z = 6.646$	$z = 6.646$

Continued on next page

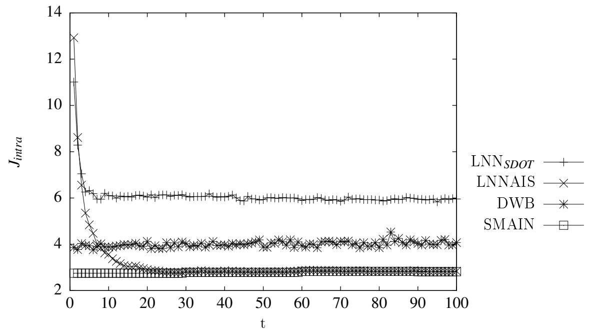
$\tilde{f}$	LNN$_{SDOT}$	DWB	SMAIN
2	$p < 0.001$ Reject H_0	$p < 0.001$ Reject H_0	$p < 0.001$ Reject H_0
3	$z = 6.646$ $p < 0.001$ Reject H_0	$z = 6.446$ $p < 0.001$ Reject H_0	$z = 6.402$ $p < 0.001$ Reject H_0
4	$z = 6.646$ $p < 0.001$ Reject H_0	$z = 6.439$ $p < 0.001$ Reject H_0	$z = 6.594$ $p < 0.001$ Reject H_0
5	$z = 6.646$ $p < 0.001$ Reject H_0	$z = 6.646$ $p < 0.001$ Reject H_0	$z = 6.557$ $p < 0.001$ Reject H_0

Table 7.16: Statistical Hypothesis Testing between LNNAIS and Other Models ($\alpha = 0.05$; with continuity correction; unpaired; non-directional) for Pattern migration at different severities of change (N=8; $|C|$=25; $\tilde{f}$=3)

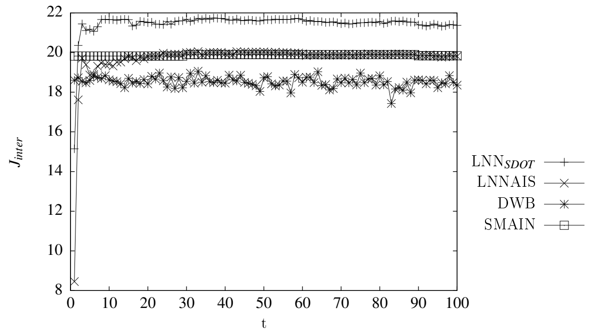
$\tilde{s}$	LNN$_{SDOT}$	DWB	SMAIN
1	$z = 6.646$ $p < 0.001$ Reject H_0	$z = 6.646$ $p < 0.001$ Reject H_0	$z = 6.409$ $p < 0.001$ Reject H_0
2	$z = 6.106$ $p < 0.001$ Reject H_0	$z = 6.646$ $p < 0.001$ Reject H_0	$z = 6.15$ $p < 0.001$ Reject H_0
3	$z = 6.646$ $p < 0.001$ Reject H_0	$z = 6.446$ $p < 0.001$ Reject H_0	$z = 6.402$ $p < 0.001$ Reject H_0
4	$z = 6.439$ $p < 0.001$ Reject H_0	$z = 6.646$ $p < 0.001$ Reject H_0	$z = 6.616$ $p < 0.001$ Reject H_0
5	$z = 6.402$ $p < 0.001$ Reject H_0	$z = 6.646$ $p < 0.001$ Reject H_0	$z = 6.416$ $p < 0.001$ Reject H_0



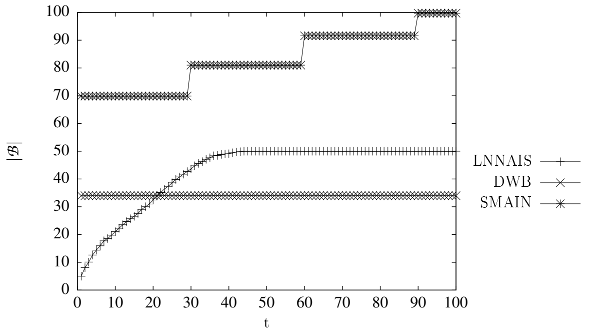
(a) Q_{ratio}



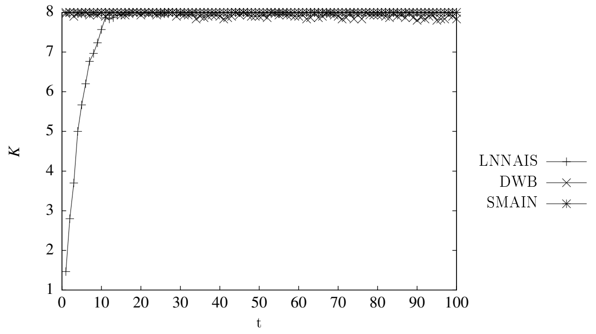
(b) J_{intra}



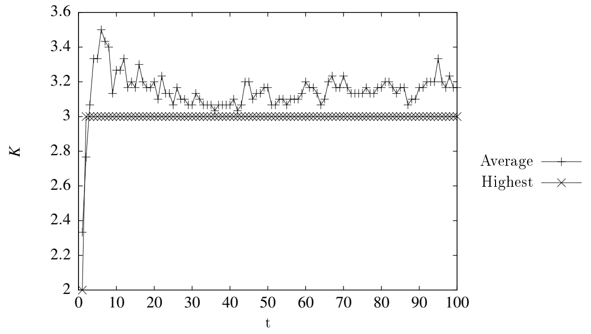
(c) J_{inter}



(d) $|\mathcal{B}|$



(e) K



(f) K (LNN$_{SDOT}$)

Figure 7.6 Pattern Migration ($N = 8, |C| = 25, \tilde{f} = 3, \tilde{s} = 3$): Quantifying each model's partitioning quality with regards to Q_{ratio}, $|\mathcal{B}|$, J_{intra}, J_{inter} and K

Although SMAIN tends to find clusters with a higher quality than LNNAIS for different dimensions (see table 7.17), cluster sizes (see table 7.18), frequencies of change (see table 7.19) and severities of change (see table 7.20), SMAIN utilises a larger ALC population size than LNNAIS for all pattern migration data sets. The larger ALC population size which is utilised by SMAIN is an indication of overfitting of the data which results in clusters of higher quality for pattern migration environments. This drawback can have a major impact in the scalability of the SMAIN model where the number of clusters changes (cluster migration environments) and where the centroids of clusters are non-stationary (centroid migration environments).

Table 7.17: Descriptive Statistics: Pattern migration at different dimensions ($|C|$=25; $\tilde{f}$=3; $\tilde{s}$=3)

N		LNN$_{SDOT}$	LNNAIS	DWB	SMAIN
	K	5.28	8	7.96	8
		(±)2.06	(±)0	(±)0.02	(±)0
	J_{intra}	5.318	1.826	3.245	1.667
		(±)2.787	(±)0.212	(±)0.08	(±)0.013
3	J_{inter}	21.349	21.616	20.095	21.643
		(±)0.782	(±)0.137	(±)0.117	(±)0.047
	$\hat{Q}$	0.368	0.248	1.578	0.151
		(±)0.135	(±)0.168	(±)0.083	(±)0.002
	$\|\mathcal{B}\|$	49.3	49.3	84	84.51
		(±)0.45	(±)0.45	(±)0	(±)2.3
	K	3.14	7.99	7.92	8
		(±)0.32	(±)0.01	(±)0.04	(±)0
	J_{intra}	5.997	2.863	4.004	2.793
		(±)0.348	(±)0.07	(±)0.117	(±)0.014
8	J_{inter}	21.557	19.887	18.53	19.869
		(±)0.219	(±)0.068	(±)0.32	(±)0.084
	$\hat{Q}$	0.335	0.586	1.286	0.436
		(±)0.029	(±)0.183	(±)0.094	(±)0.012
	$\|\mathcal{B}\|$	45.97	45.97	34	83
		(±)1.26	(±)1.26	(±)0	(±)2.89

Continued on next page

N		LNN$_{SDOT}$	LNNAIS	DWB	SMAIN
	K	4.75	7.97	7.92	7.84
		(±)0.74	(±)0.05	(±)0.06	(±)0.23
	J_{intra}	5.699	4.059	5.347	4.266
		(±)0.803	(±)0.064	(±)0.19	(±)0.068
15	J_{inter}	22.169	20.565	19.685	20.607
		(±)0.464	(±)0.106	(±)0.312	(±)0.224
	$\hat{Q}$	0.425	0.973	1.225	0.663
		(±)0.102	(±)0.117	(±)0.059	(±)0.015
	$\|\mathcal{B}\|$	43.79	43.79	29	84.18
		(±)1.67	(±)1.67	(±)0	(±)3.37

Table 7.18: Descriptive Statistics: Pattern migration at different cluster sizes (N=8; $\tilde{s}$=3; $\tilde{f}$=3)

$\|C\|$		LNN$_{SDOT}$	LNNAIS	DWB	SMAIN
	K	3.09	3.66	7.92	8
		(±)0.09	(±)0.07	(±)0.04	(±)0
	J_{intra}	6.508	5.99	3.619	2.582
		(±)0.157	(±)0.13	(±)0.055	(±)0.005
10	J_{inter}	21.098	19.888	18.606	19.972
		(±)0.125	(±)0.137	(±)0.197	(±)0.04
	$\hat{Q}$	0.497	0.874	1.194	0.408
		(±)0.173	(±)0.096	(±)0.038	(±)0.006
	$\|\mathcal{B}\|$	9.99	9.99	43	71.51
		(±)0.02	(±)0.02	(±)0	(±)1.53
	K	3.14	7.99	7.92	8
		(±)0.32	(±)0.01	(±)0.04	(±)0
	J_{intra}	5.997	2.863	4.004	2.793
		(±)0.348	(±)0.07	(±)0.117	(±)0.014
25	J_{inter}	21.557	19.887	18.53	19.869
		(±)0.219	(±)0.068	(±)0.32	(±)0.084
	$\hat{Q}$	0.335	0.586	1.286	0.436

Continued on next page

$\|C\|$		LNN$_{SDOT}$	LNNAIS	DWB	SMAIN
		(±)0.029	(±)0.183	(±)0.094	(±)0.012
	$\|\mathcal{B}\|$	45.97	45.97	34	83
		(±)1.26	(±)1.26	(±)0	(±)2.89
	K	3.11	8	7.95	8
		(±)0.31	(±)0	(±)0.03	(±)0
	J_{intra}	6.018	2.84	3.915	2.819
		(±)0.307	(±)0.019	(±)0.105	(±)0.017
50	J_{inter}	21.683	19.965	18.546	20.107
		(±)0.267	(±)0.035	(±)0.334	(±)0.107
	$\hat{Q}$	0.326	0.499	1.296	0.439
		(±)0.044	(±)0.043	(±)0.06	(±)0.015
	$\|\mathcal{B}\|$	39.7	39.7	44	72.02
		(±)0.28	(±)0.28	(±)0	(±)3.92

Table 7.19: Descriptive Statistics: Pattern migration at different frequencies of change (N=8; $|C|$=25; $\tilde{s}$=3)

$\tilde{f}$		LNN$_{SDOT}$	LNNAIS	DWB	SMAIN
	K	3.21	7.99	7.94	8
		(±)0.59	(±)0.02	(±)0.03	(±)0
	J_{intra}	6.021	2.891	4.077	2.891
		(±)0.519	(±)0.034	(±)0.077	(±)0.019
1	J_{inter}	21.934	20.158	18.772	20.226
		(±)0.342	(±)0.051	(±)0.199	(±)0.093
	$\hat{Q}$	0.335	0.559	1.287	0.435
		(±)0.049	(±)0.089	(±)0.054	(±)0.015
	$\|\mathcal{B}\|$	46.07	46.07	32	61.01
		(±)1.36	(±)1.36	(±)0	(±)2.78
	K	3.09	8	7.94	8
		(±)0.29	(±)0.01	(±)0.03	(±)0
	J_{intra}	6.118	2.861	3.87	2.825
		(±)0.313	(±)0.047	(±)0.071	(±)0.016

Continued on next page

$\tilde{f}$		LNN$_{SDOT}$	LNNAIS	DWB	SMAIN
2	J_{inter}	21.967	19.965	18.67	20.253
		$(\pm)0.255$	$(\pm)0.116$	$(\pm)0.225$	$(\pm)0.105$
	$\hat{Q}$	0.335	0.629	1.287	0.409
		$(\pm)0.082$	$(\pm)0.111$	$(\pm)0.054$	$(\pm)0.009$
	$\|\mathcal{B}\|$	38.4	38.4	40	64.06
		$(\pm)0.66$	$(\pm)0.66$	$(\pm)0$	$(\pm)2.77$
	K	3.14	7.99	7.92	8
		$(\pm)0.32$	$(\pm)0.01$	$(\pm)0.04$	$(\pm)0$
	J_{intra}	5.997	2.863	4.004	2.793
		$(\pm)0.348$	$(\pm)0.07$	$(\pm)0.117$	$(\pm)0.014$
3	J_{inter}	21.557	19.887	18.53	19.869
		$(\pm)0.219$	$(\pm)0.068$	$(\pm)0.32$	$(\pm)0.084$
	$\hat{Q}$	0.335	0.586	1.286	0.436
		$(\pm)0.029$	$(\pm)0.183$	$(\pm)0.094$	$(\pm)0.012$
	$\|\mathcal{B}\|$	45.97	45.97	34	83
		$(\pm)1.26$	$(\pm)1.26$	$(\pm)0$	$(\pm)2.89$
	K	3.05	7.99	7.94	8
		$(\pm)0.06$	$(\pm)0.02$	$(\pm)0.03$	$(\pm)0$
	J_{intra}	6.213	2.881	3.855	2.974
		$(\pm)0.07$	$(\pm)0.072$	$(\pm)0.075$	$(\pm)0.044$
4	J_{inter}	21.548	19.806	18.53	19.939
		$(\pm)0.11$	$(\pm)0.109$	$(\pm)0.248$	$(\pm)0.174$
	$\hat{Q}$	0.332	0.635	1.269	0.426
		$(\pm)0.02$	$(\pm)0.198$	$(\pm)0.073$	$(\pm)0.02$
	$\|\mathcal{B}\|$	46.19	46.19	44	37.09
		$(\pm)1.78$	$(\pm)1.78$	$(\pm)0$	$(\pm)2.45$
	K	3.1	7.98	7.95	8
		$(\pm)0.26$	$(\pm)0.03$	$(\pm)0.02$	$(\pm)0$
	J_{intra}	6.033	2.85	3.908	2.845
		$(\pm)0.24$	$(\pm)0.041$	$(\pm)0.062$	$(\pm)0.113$
5	J_{inter}	21.636	19.877	18.558	20.187
		$(\pm)0.185$	$(\pm)0.04$	$(\pm)0.171$	$(\pm)0.207$

Continued on next page

$\tilde{f}$		LNN$_{SDOT}$	LNNAIS	DWB	SMAIN
	$\hat{Q}$	0.323	0.601	1.274	0.437
		(±)0.028	(±)0.119	(±)0.048	(±)0.017
	$\lvert\mathcal{B}\rvert$	38.2	38.2	37	55.03
		(±)1.06	(±)1.06	(±)0	(±)2.9

Table 7.20: Descriptive Statistics: Pattern migration at different severities of change (N=8; $|C|$=25; $\tilde{f}$=3)

$\tilde{s}$		LNN$_{SDOT}$	LNNAIS	DWB	SMAIN
	K	3.02	7.99	7.95	8
		(±)0.03	(±)0.02	(±)0.02	(±)0
	J_{intra}	6.067	2.829	3.945	2.762
		(±)0.052	(±)0.051	(±)0.149	(±)0.017
1	J_{inter}	21.822	19.963	18.71	20.045
		(±)0.065	(±)0.066	(±)0.271	(±)0.096
	$\hat{Q}$	0.314	0.594	1.263	0.435
		(±)0.012	(±)0.146	(±)0.096	(±)0.018
	$\lvert\mathcal{B}\rvert$	45.42	45.42	34	68.13
		(±)1.67	(±)1.67	(±)0	(±)2.96
	K	3.17	7.99	7.96	8
		(±)0.56	(±)0.02	(±)0.02	(±)0
	J_{intra}	5.931	2.862	3.88	2.776
		(±)0.371	(±)0.072	(±)0.063	(±)0.015
2	J_{inter}	21.765	19.898	18.605	20.072
		(±)0.438	(±)0.097	(±)0.173	(±)0.067
	$\hat{Q}$	0.346	0.6	1.26	0.453
		(±)0.125	(±)0.142	(±)0.041	(±)0.016
	$\lvert\mathcal{B}\rvert$	45.76	45.76	37	73.46
		(±)1.74	(±)1.74	(±)0	(±)2.73
	K	3.14	7.99	7.92	8
		(±)0.32	(±)0.01	(±)0.04	(±)0
	J_{intra}	5.997	2.863	4.004	2.793

Continued on next page

$\tilde{s}$		LNN$_{SDOT}$	LNNAIS	DWB	SMAIN
		(±)0.348	(±)0.07	(±)0.117	(±)0.014
3	J_{inter}	21.557	19.887	18.53	19.869
		(±)0.219	(±)0.068	(±)0.32	(±)0.084
	$\hat{Q}$	0.335	0.586	1.286	0.436
		(±)0.029	(±)0.183	(±)0.094	(±)0.012
	$\|\mathcal{B}\|$	45.97	45.97	34	83
		(±)1.26	(±)1.26	(±)0	(±)2.89
	K	3.06	7.99	7.9	8
		(±)0.36	(±)0.02	(±)0.05	(±)0
	J_{intra}	6.202	2.898	3.977	2.81
		(±)0.641	(±)0.057	(±)0.107	(±)0.015
4	J_{inter}	21.648	19.941	18.497	20.008
		(±)0.398	(±)0.096	(±)0.388	(±)0.093
	$\hat{Q}$	0.337	0.598	1.275	0.438
		(±)0.047	(±)0.122	(±)0.095	(±)0.014
	$\|\mathcal{B}\|$	45.7	45.7	41	78.81
		(±)1.38	(±)1.38	(±)0	(±)3.21
	K	3.23	7.99	7.95	8
		(±)0.87	(±)0.03	(±)0.02	(±)0
	J_{intra}	6.023	2.891	3.944	2.82
		(±)0.898	(±)0.045	(±)0.053	(±)0.015
5	J_{inter}	21.649	19.976	18.578	20.163
		(±)0.471	(±)0.076	(±)0.208	(±)0.068
	$\hat{Q}$	0.339	0.601	1.269	0.444
		(±)0.058	(±)0.119	(±)0.04	(±)0.011
	$\|\mathcal{B}\|$	45.88	45.88	37	78.66
		(±)1.67	(±)1.67	(±)0	(±)2.71

7.5.2 Cluster Migration

Figure 7.7 illustrates the quality of partitioning by the different models over time for cluster migration. Similar trends as for pattern migration are illustrated for cluster quality (see figure 7.7(a)) and ALC population sizes (see figure 7.7(d)). As for pattern migration there is an increase in the ALC population size for the SMAIN model with every change in the data (figure 7.7(d) at $t = 30$, $t = 60$ and $t = 90$). The drawback of SMAIN to potentially overfit the data is emphasised with cluster migration environments, since it is expected to utilise a smaller ALC population size with a decrease in the number of clusters in the data (as illustrated in figures 7.7(d) and 7.7(e), the ALC population size of SMAIN increases even with a decrease in the number of clusters). Figure 7.7(e) illustrates that LNNAIS and SMAIN detected the change in the number of clusters at $t = 30$. The expected number of clusters for $t \geq 30$ is $K = 6$ which is correctly obtained by LNNAIS and SMAIN. DWB tends to cluster the data into slightly more than six clusters ($6 < K < 7$), because of the hybrid approach followed by DWB (using K-means clustering). The DWB model partitions the ALC population into the initial eight clusters (eight sub-nets) at each step in time. This results into an average of 6.97 clusters at $t \geq 30$ (as illustrated in figure 7.7(e)), which explains the lower quality of clusters found by DWB when compared to SMAIN and LNNAIS (as illustrated in figure 7.7(a)). Again, LNN$_{SDOT}$ did not detect the correct number of clusters, but did however detect a change in the data at $t = 30$, $t = 60$ and $t = 90$ (see figure 7.7(f)). Overall, there is no significant change in Q_{ratio}, J_{intra}, J_{inter} and K for all the models ($t > 40$) in cluster migration environments except for LNN$_{SDOT}$ where K fluctuates between $K = 2$ and $K = 3$ at every change in the data (explained in section 7.5.3).

The Mann-Whitney U statistical hypothesis test rejects H_0 that the mean clustering quality, $\bar{Q}$, are the same at a 0.05 level of significance between LNNAIS and LNN$_{SDOT}$, SMAIN and DWB for different dimensions (as summarised in table 7.21), different clusters sizes (as summarised in table 7.22), for all frequencies of change (as summarised in table 7.23), and all severities of change (as summarised in table 7.24). There is thus a statistical significant difference in the clustering quality of all the cluster migration data sets between LNNAIS and all the other models.

Table 7.21: Statistical Hypothesis Testing between LNNAIS and Other Models ($\alpha = 0.05$; with continuity correction; unpaired; non-directional) for Cluster migration at different dimensions ($|C|$=25; $\tilde{f}$=3; $\tilde{s}$=3)

N	LNN$_{SDOT}$	DWB	SMAIN
3	$z = 4.568$	$z = 6.646$	$z = 6.646$
3	$p < 0.001$	$p < 0.001$	$p < 0.001$
3	Reject H_0	Reject H_0	Reject H_0
8	$z = 6.646$	$z = 6.646$	$z = 6.646$

Continued on next page

N	LNN$_{SDOT}$	DWB	SMAIN
8	$p < 0.001$	$p < 0.001$	$p < 0.001$
8	Reject H_0	Reject H_0	Reject H_0
15	$z = 5.455$	$z = 5.862$	$z = 5.64$
15	$p < 0.001$	$p < 0.001$	$p < 0.001$
15	Reject H_0	Reject H_0	Reject H_0

Table 7.22: Statistical Hypothesis Testing between LNNAIS and Other Models ($\alpha = 0.05$; with continuity correction; unpaired; non-directional) for Cluster migration at different cluster sizes (N=8; $\tilde{s}$=3; $\tilde{f}$=3)

$\|C\|$	LNN$_{SDOT}$	DWB	SMAIN
	$z = 6.646$	$z = 6.527$	$z = 6.646$
10	$p < 0.001$	$p < 0.001$	$p < 0.001$
	Reject H_0	Reject H_0	Reject H_0
	$z = 6.646$	$z = 6.646$	$z = 6.646$
25	$p < 0.001$	$p < 0.001$	$p < 0.001$
	Reject H_0	Reject H_0	Reject H_0
	$z = 6.646$	$z = 6.646$	$z = 3.785$
50	$p < 0.001$	$p < 0.001$	$p < 0.001$
	Reject H_0	Reject H_0	Reject H_0

Table 7.23: Statistical Hypothesis Testing between LNNAIS and Other Models ($\alpha = 0.05$; with continuity correction; unpaired; non-directional) for Cluster migration at different frequencies of change (N=8; $|C|$=25; $\tilde{s}$=3)

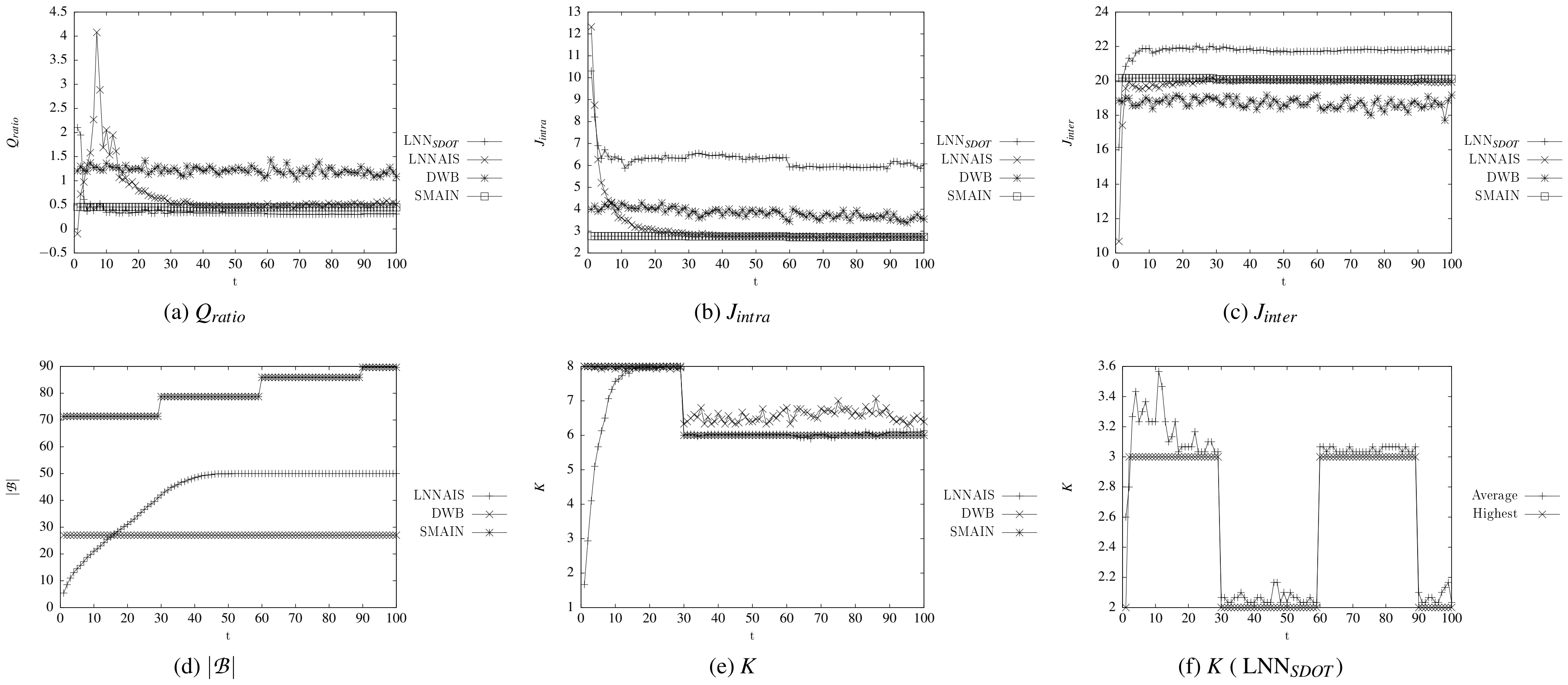
$\tilde{f}$	LNN$_{SDOT}$	DWB	SMAIN
	$z = 6.631$	$z = 6.631$	$z = 6.631$
1	$p < 0.001$	$p < 0.001$	$p < 0.001$
	Reject H_0	Reject H_0	Reject H_0
	$z = 5.877$	$z = 6.646$	$z = 6.616$

Continued on next page

$\tilde{f}$	LNN$_{SDOT}$	DWB	SMAIN
2	$p < 0.001$ Reject H_0	$p < 0.001$ Reject H_0	$p < 0.001$ Reject H_0
3	$z = 6.646$ $p < 0.001$ Reject H_0	$z = 6.646$ $p < 0.001$ Reject H_0	$z = 6.646$ $p < 0.001$ Reject H_0
4	$z = 6.646$ $p < 0.001$ Reject H_0	$z = 6.646$ $p < 0.001$ Reject H_0	$z = 6.631$ $p < 0.001$ Reject H_0
5	$z = 6.35$ $p < 0.001$ Reject H_0	$z = 6.276$ $p < 0.001$ Reject H_0	$z = 6.579$ $p < 0.001$ Reject H_0

Table 7.24: Statistical Hypothesis Testing between LNNAIS and Other Models ($\alpha = 0.05$; with continuity correction; unpaired; non-directional) for Cluster migration at different severities of change (N=8; $|C|$=25; $\tilde{f}$=3)

$\tilde{s}$	LNN$_{SDOT}$	DWB	SMAIN
1	$z = 6.291$ $p < 0.001$ Reject H_0	$z = 6.202$ $p < 0.001$ Reject H_0	$z = 6.646$ $p < 0.001$ Reject H_0
2	$z = 6.601$ $p < 0.001$ Reject H_0	$z = 6.646$ $p < 0.001$ Reject H_0	$z = 6.646$ $p < 0.001$ Reject H_0
3	$z = 6.646$ $p < 0.001$ Reject H_0	$z = 6.646$ $p < 0.001$ Reject H_0	$z = 6.646$ $p < 0.001$ Reject H_0
4	$z = 6.431$ $p < 0.001$ Reject H_0	$z = 6.527$ $p < 0.001$ Reject H_0	$z = 6.631$ $p < 0.001$ Reject H_0
5	$z = 6.328$ $p < 0.001$ Reject H_0	$z = 6.601$ $p < 0.001$ Reject H_0	$z = 6.32$ $p < 0.001$ Reject H_0



(a) Q_{ratio} (b) J_{intra} (c) J_{inter}

(d) $|\mathcal{B}|$ (e) K (f) K (LNN$_{SDOT}$)

Figure 7.7 Cluster Migration ($N = 8, |C| = 25, \tilde{f} = 3, \tilde{s} = 3$): Quantifying each model's partitioning quality with regards to Q_{ratio}, $|\mathcal{B}|$, J_{intra}, J_{inter} and K

Similar to the pattern migration environments, SMAIN utilises a larger ALC population size than LNNAIS and tends to overfit the data. This results in clusters of higher quality for cluster migration environments when compared to LNNAIS at different dimensions (see table 7.25), cluster sizes (see table 7.26), frequencies of change (see table 7.27) and severities of change (see table 7.28). The drawback of overfit is even more emphasised in cluster migration environments where the ALC population size of the SMAIN model does not scale with the number of clusters. LNNAIS delivers clusters of a higher quality than DWB and LNN$_{SDOT}$ for all cluster migration environments. LNNAIS also obtains the correct number of clusters at different severities of change with no significant change in the ALC population size (see table 7.28 where an increase in $\tilde{s}$ increases the number of clusters migrating and disappearing in the data, i.e. decreasing the number of clusters in the data).

Table 7.25: Descriptive Statistics: Cluster migration at different dimensions ($|C|$=25; $\tilde{f}$=3; $\tilde{s}$=3)

N		LNN$_{SDOT}$	LNNAIS	DWB	SMAIN
	K	4.38	6.46	7.25	6.58
		(±)1.31	(±)0.08	(±)0.08	(±)0
	J_{intra}	5.565	1.66	2.711	1.555
		(±)2.487	(±)0.132	(±)0.106	(±)0.008
3	J_{inter}	21.426	21.609	20.031	21.707
		(±)0.585	(±)0.147	(±)0.163	(±)0.053
	$\hat{Q}$	0.367	0.224	1.468	0.14
		(±)0.124	(±)0.102	(±)0.093	(±)0.002
	$\|\mathcal{B}\|$	49.15	49.15	95	83.15
		(±)0.47	(±)0.47	(±)0	(±)2.51
	K	2.62	6.43	6.97	6.58
		(±)0.18	(±)0.09	(±)0.2	(±)0
	J_{intra}	6.174	2.835	3.835	2.75
		(±)0.282	(±)0.046	(±)0.164	(±)0.015
8	J_{inter}	21.799	19.979	18.717	20.118
		(±)0.21	(±)0.085	(±)0.315	(±)0.074
	$\hat{Q}$	0.326	0.59	1.214	0.45
		(±)0.023	(±)0.105	(±)0.088	(±)0.011

Continued on next page

N		LNN$_{SDOT}$	LNNAIS	DWB	SMAIN
	$\|\mathcal{B}\|$	45.49	45.49	27	79.97
		(±)1.63	(±)1.63	(±)0	(±)3.83
	K	3.69	3.8	7.45	6.64
		(±)0.11	(±)0.06	(±)0.14	(±)0.11
	J_{intra}	6.705	6.845	4.93	4.034
		(±)0.225	(±)0.163	(±)0.183	(±)0.083
15	J_{inter}	21.52	20.84	19.678	20.739
		(±)0.155	(±)0.165	(±)0.344	(±)0.169
	$\hat{Q}$	0.553	0.856	1.219	0.633
		(±)0.13	(±)0.188	(±)0.046	(±)0.019
	$\|\mathcal{B}\|$	10	10	31	79.7
		(±)0	(±)0	(±)0	(±)3.19

Table 7.26: Descriptive Statistics: Cluster migration at different cluster sizes (N=8; $\tilde{s}$=3; $\tilde{f}$=3)

$\|C\|$		LNN$_{SDOT}$	LNNAIS	DWB	SMAIN
	K	3.11	3.7	7.2	6.58
		(±)0.09	(±)0.05	(±)0.13	(±)0
	J_{intra}	6.046	5.389	3.671	2.752
		(±)0.185	(±)0.109	(±)0.117	(±)0.008
10	J_{inter}	21.989	20.648	18.821	20.097
		(±)0.252	(±)0.149	(±)0.343	(±)0.054
	$\hat{Q}$	0.386	0.816	1.132	0.421
		(±)0.045	(±)0.105	(±)0.067	(±)0.009
	$\|\mathcal{B}\|$	10	10	35	71.79
		(±)0.01	(±)0.01	(±)0	(±)1.11
	K	2.62	6.43	6.97	6.58
		(±)0.18	(±)0.09	(±)0.2	(±)0
	J_{intra}	6.174	2.835	3.835	2.75
		(±)0.282	(±)0.046	(±)0.164	(±)0.015
25	J_{inter}	21.799	19.979	18.717	20.118

Continued on next page

$\lvert C\rvert$		LNN$_{SDOT}$	LNNAIS	DWB	SMAIN
		(±)0.21	(±)0.085	(±)0.315	(±)0.074
	$\hat{Q}$	0.326	0.59	1.214	0.45
		(±)0.023	(±)0.105	(±)0.088	(±)0.011
	$\lvert\mathcal{B}\rvert$	45.49	45.49	27	79.97
		(±)1.63	(±)1.63	(±)0	(±)3.83
	K	3.04	6.56	7.24	6.58
		(±)0.14	(±)0.13	(±)0.14	(±)0
	J_{intra}	5.565	2.851	3.808	2.864
		(±)0.231	(±)0.018	(±)0.108	(±)0.017
50	J_{inter}	21.576	19.938	18.27	20.021
		(±)0.153	(±)0.083	(±)0.44	(±)0.1
	$\hat{Q}$	0.295	0.522	1.236	0.452
		(±)0.023	(±)0.104	(±)0.068	(±)0.017
	$\lvert\mathcal{B}\rvert$	48.89	48.89	31	72
		(±)0.73	(±)0.73	(±)0	(±)3.05

Table 7.27: Descriptive Statistics: Cluster migration at different frequencies of change (N=8; $\lvert C\rvert$=25; $\tilde{s}$=3)

$\tilde{f}$		LNN$_{SDOT}$	LNNAIS	DWB	SMAIN
	K	2.95	6.26	7.03	6.25
		(±)0.2	(±)0.12	(±)0.12	(±)0.05
	J_{intra}	5.617	2.833	3.617	2.822
		(±)0.472	(±)0.041	(±)0.067	(±)0.017
1	J_{inter}	21.667	19.686	18.281	20.307
		(±)0.173	(±)0.156	(±)0.357	(±)0.075
	$\hat{Q}$	0.303	0.77	1.203	0.445
		(±)0.044	(±)0.167	(±)0.062	(±)0.016
	$\lvert\mathcal{B}\rvert$	37.97	37.97	29	69.26
		(±)0.73	(±)0.73	(±)0	(±)2.75
	K	3.19	6.38	7.2	6.38
		(±)0.37	(±)0.15	(±)0.11	(±)0

Continued on next page

$\tilde{f}$		LNN$_{SDOT}$	LNNAIS	DWB	SMAIN
	J_{intra}	5.764 (±)0.613	2.832 (±)0.046	3.681 (±)0.078	2.793 (±)0.015
2	J_{inter}	21.561 (±)0.679	19.94 (±)0.112	18.429 (±)0.323	20.145 (±)0.087
	$\hat{Q}$	0.366 (±)0.11	0.671 (±)0.141	1.217 (±)0.062	0.439 (±)0.016
	$\|\mathcal{B}\|$	45.68 (±)1.64	45.68 (±)1.64	31 (±)0	74.82 (±)2.84
	K	2.62 (±)0.18	6.43 (±)0.09	6.97 (±)0.2	6.58 (±)0
	J_{intra}	6.174 (±)0.282	2.835 (±)0.046	3.835 (±)0.164	2.75 (±)0.015
3	J_{inter}	21.799 (±)0.21	19.979 (±)0.085	18.717 (±)0.315	20.118 (±)0.074
	$\hat{Q}$	0.326 (±)0.023	0.59 (±)0.105	1.214 (±)0.088	0.45 (±)0.011
	$\|\mathcal{B}\|$	45.49 (±)1.63	45.49 (±)1.63	27 (±)0	79.97 (±)3.83
	K	2.95 (±)0.45	6.68 (±)0.12	7.61 (±)0.13	6.78 (±)0
	J_{intra}	5.691 (±)0.477	2.895 (±)0.046	3.702 (±)0.063	2.783 (±)0.022
4	J_{inter}	21.621 (±)0.372	19.878 (±)0.115	18.223 (±)0.205	19.944 (±)0.064
	$\hat{Q}$	0.324 (±)0.054	0.658 (±)0.136	1.221 (±)0.038	0.455 (±)0.015
	$\|\mathcal{B}\|$	45.96 (±)1.6	45.96 (±)1.6	32 (±)0	83.78 (±)2.24
	K	3.03 (±)0.28	6.96 (±)0.16	7.45 (±)0.24	6.98 (±)0
	J_{intra}	6.086 (±)0.728	2.932 (±)0.079	3.965 (±)0.129	2.828 (±)0.012

Continued on next page

$\tilde{f}$		LNN$_{SDOT}$	LNNAIS	DWB	SMAIN
5	J_{inter}	21.726	19.77	18.519	20.023
		(±)0.295	(±)0.181	(±)0.405	(±)0.095
	$\hat{Q}$	0.337	0.695	1.215	0.439
		(±)0.076	(±)0.208	(±)0.079	(±)0.012
	$\|\mathcal{B}\|$	46.05	46.05	27	80.23
		(±)1.77	(±)1.77	(±)0	(±)3.46

Table 7.28: Descriptive Statistics: Cluster migration at different severities of change (N=8; $|C|$=25; $\tilde{f}$=3)

$\tilde{s}$		LNN$_{SDOT}$	LNNAIS	DWB	SMAIN
	K	3.02	7.99	7.95	8
		(±)0.26	(±)0.02	(±)0.02	(±)0
	J_{intra}	6.336	2.892	3.908	2.807
		(±)0.69	(±)0.067	(±)0.103	(±)0.013
1	J_{inter}	21.911	20.105	18.692	20.178
		(±)0.303	(±)0.081	(±)0.251	(±)0.062
	$\hat{Q}$	0.342	0.609	1.262	0.422
		(±)0.065	(±)0.168	(±)0.063	(±)0.017
	$\|\mathcal{B}\|$	46.07	46.07	44	73.39
		(±)1.53	(±)1.53	(±)0	(±)2.82
	K	3.04	7.26	7.75	7.29
		(±)0.1	(±)0.18	(±)0.09	(±)0
	J_{intra}	6.118	2.747	3.722	2.705
		(±)0.251	(±)0.06	(±)0.072	(±)0.02
2	J_{inter}	21.617	19.827	18.552	20.051
		(±)0.228	(±)0.193	(±)0.201	(±)0.078
	$\hat{Q}$	0.328	0.616	1.27	0.384
		(±)0.027	(±)0.157	(±)0.047	(±)0.01
	$\|\mathcal{B}\|$	45.84	45.84	36	69.04
		(±)1.32	(±)1.32	(±)0	(±)3.22

Continued on next page

$\tilde{s}$		LNN$_{SDOT}$	LNNAIS	DWB	SMAIN
	K	2.62	6.43	6.97	6.58
		(±)0.18	(±)0.09	(±)0.2	(±)0
	J_{intra}	6.174	2.835	3.835	2.75
		(±)0.282	(±)0.046	(±)0.164	(±)0.015
3	J_{inter}	21.799	19.979	18.717	20.118
		(±)0.21	(±)0.085	(±)0.315	(±)0.074
	$\hat{Q}$	0.326	0.59	1.214	0.45
		(±)0.023	(±)0.105	(±)0.088	(±)0.011
	$\lvert\mathcal{B}\rvert$	45.49	45.49	27	79.97
		(±)1.63	(±)1.63	(±)0	(±)3.83
	K	3.06	5.75	7.29	5.87
		(±)0.11	(±)0.15	(±)0.07	(±)0
	J_{intra}	5.787	2.81	3.448	2.79
		(±)0.398	(±)0.036	(±)0.068	(±)0.017
4	J_{inter}	21.516	19.729	17.926	19.754
		(±)0.356	(±)0.171	(±)0.231	(±)0.11
	$\hat{Q}$	0.322	0.689	1.207	0.425
		(±)0.058	(±)0.211	(±)0.044	(±)0.014
	$\lvert\mathcal{B}\rvert$	45.6	45.6	36	63.18
		(±)1.64	(±)1.64	(±)0	(±)2.33
	K	3.11	4.86	6.4	5.13
		(±)0.29	(±)0.08	(±)0.11	(±)0.09
	J_{intra}	5.634	2.863	3.473	2.879
		(±)0.575	(±)0.042	(±)0.074	(±)0.088
5	J_{inter}	21.489	19.832	18.296	19.861
		(±)0.637	(±)0.109	(±)0.272	(±)0.148
	$\hat{Q}$	0.333	0.615	1.176	0.437
		(±)0.07	(±)0.142	(±)0.059	(±)0.021
	$\lvert\mathcal{B}\rvert$	45.49	45.49	39	65.09
		(±)1.45	(±)1.45	(±)0	(±)2.89

7.5.3 Centroid Migration

Figure 7.8 illustrates the quality of partitioning by the different models over time for centroid migration. Different to the pattern and cluster migration data sets, the centroids in the centroid migration data sets are non-stationary. Non-stationary centroids result in merging of clusters and division of clusters. Figure 7.8(e) illustrates that LNNAIS and SMAIN detected the change in the number of clusters at $t = 30$, $t = 60$ and $t = 90$. The expected number of clusters for $30 \leq t < 60$ and $60 \leq t < 90$ is $K = 6$ and $K = 4$ for $t \geq 90$ which is correctly obtained by LNNAIS and SMAIN. A similar drawback of DWB for cluster migration as for pattern and cluster migration is that DWB tends to cluster the data into slightly more clusters, because of the hybrid approach followed by DWB (using K-means clustering). The DWB model partitions the ALC population into the initial eight clusters (eight sub-nets) at each step in time. Different to pattern and cluster migration, LNN$_{SDOT}$ detected the change in the data at $t = 30$, $t = 60$ and $t = 90$ and determined the correct number of clusters. LNN$_{SDOT}$ did not determine the correct number of clusters for $t < 30$. Since the centroids in the centroid migration is non-stationary, the distances between these centroids change over time. This has a direct influence on the network affinity between the ALCs in LNNAIS, since the ALCs adapt to the clusters. As a result of the changes in the distance between the centroids, the sequential outlier technique detects more network affinities as outliers (utilised by LNN$_{SDOT}$ to dynamically determine the ALC network boundaries, as explained in section 6.2). Detecting more ALC network boundaries correctly determines the number of clusters in the data set. This highlights a potential drawback of the sequential deviation outlier detection technique used by LNN$_{SDOT}$ which is that if the centroids of clusters in a data set are uniformly distributed with equal distances, the ALC networks formed in LNNAIS will have equal network affinities between each other, resulting in no outlier network affinities. This is expected since the sequential deviation outlier detection technique will detect no outliers and therefore no boundaries between the ALC networks. This was the case for the pattern and cluster migration environments where the spatial positions of the centroids remain stationary (refer to figures 7.6(f) and figures 7.7(f) where $K \leq 3$ at all time steps). The same argument applies to centroid migration where $t < 30$ as illustrated in figure 7.8(f), since prior to this point in time none of the centroids have changed their spatial positions and only three boundaries between the ALC networks were detected by LNN$_{SDOT}$.

The drawback of SMAIN to potentially overfit the data is also highlighted with centroid migration environments, since it is expected to utilise a smaller ALC population size with a decrease in the number of clusters in the data (as illustrated in figures 7.8(d), 7.8(e) and 7.8(a); the

ALC population size of SMAIN increases even at a decrease in the number of clusters with no significant gain in the cluster quality). Furthermore the clusters found by SMAIN become less compact at each change (as illustrated in figure 7.8(b)). Figures 7.8(a) shows that the quality of clusters found by LNNAIS lowers at each change, but that LNNAIS succeeds to recover from the change and improve on the cluster quality as time progresses, even though the number of clusters changes and clusters become less compact (as illustrated in figures 7.8(b) and 7.8(e)). LNNAIS delivers clusters of a higher quality than DWB (for all t) and similar quality as LNN$_{SDOT}$ (for $t > 30$) for all centroid migration environments.

Table 7.29: Statistical Hypothesis Testing between LNNAIS and Other Models ($\alpha = 0.05$; with continuity correction; unpaired; non-directional) for Centroid migration at different dimensions ($|C|$=25; $\tilde{f}$=3; $\tilde{s}$=3)

N	LNN$_{SDOT}$	DWB	SMAIN
3	$z = 3.326$	$z = 6.646$	$z = 5.914$
3	$p < 0.001$	$p < 0.001$	$p < 0.001$
3	Reject H_0	Reject H_0	Reject H_0
8	$z = 4.805$	$z = 6.646$	$z = 6.646$
8	$p < 0.001$	$p < 0.001$	$p < 0.001$
8	Reject H_0	Reject H_0	Reject H_0
15	$z = 6.498$	$z = 6.646$	$z = 6.276$
15	$p < 0.001$	$p < 0.001$	$p < 0.001$
15	Reject H_0	Reject H_0	Reject H_0

Table 7.30: Statistical Hypothesis Testing between LNNAIS and Other Models ($\alpha = 0.05$; with continuity correction; unpaired; non-directional) for Centroid migration at different cluster sizes (N=8; $\tilde{s}$=3; $\tilde{f}$=3)

$\|C\|$	LNN$_{SDOT}$	DWB	SMAIN
	$z = 6.232$	$z = 6.209$	$z = 6.646$
10	$p < 0.001$	$p < 0.001$	$p < 0.001$
	Reject H_0	Reject H_0	Reject H_0
	$z = 4.805$	$z = 6.646$	$z = 6.646$

Continued on next page

$\lvert C\rvert$	LNN$_{SDOT}$	DWB	SMAIN
25	$p < 0.001$ Reject H_0	$p < 0.001$ Reject H_0	$p < 0.001$ Reject H_0
50	$z = 5.67$ $p < 0.001$ Reject H_0	$z = 6.646$ $p < 0.001$ Reject H_0	$z = 4.709$ $p < 0.001$ Reject H_0

Table 7.31: Statistical Hypothesis Testing between LNNAIS and Other Models ($\alpha = 0.05$; with continuity correction; unpaired; non-directional) for Centroid migration at different frequencies of change (N=8; $\lvert C\rvert$=25; $\tilde{s}$=3)

$\tilde{f}$	LNN$_{SDOT}$	DWB	SMAIN
1	$z = 3.674$ $p < 0.001$ Reject H_0	$z = 6.646$ $p < 0.001$ Reject H_0	$z = 6.646$ $p < 0.001$ Reject H_0
2	$z = 6.158$ $p < 0.001$ Reject H_0	$z = 6.646$ $p < 0.001$ Reject H_0	$z = 6.631$ $p < 0.001$ Reject H_0
3	$z = 4.805$ $p < 0.001$ Reject H_0	$z = 6.646$ $p < 0.001$ Reject H_0	$z = 6.646$ $p < 0.001$ Reject H_0
4	$z = 4.687$ $p < 0.001$ Reject H_0	$z = 6.646$ $p < 0.001$ Reject H_0	$z = 5.722$ $p < 0.001$ Reject H_0
5	$z = 5.241$ $p < 0.001$ Reject H_0	$z = 6.453$ $p < 0.001$ Reject H_0	$z = 6.646$ $p < 0.001$ Reject H_0

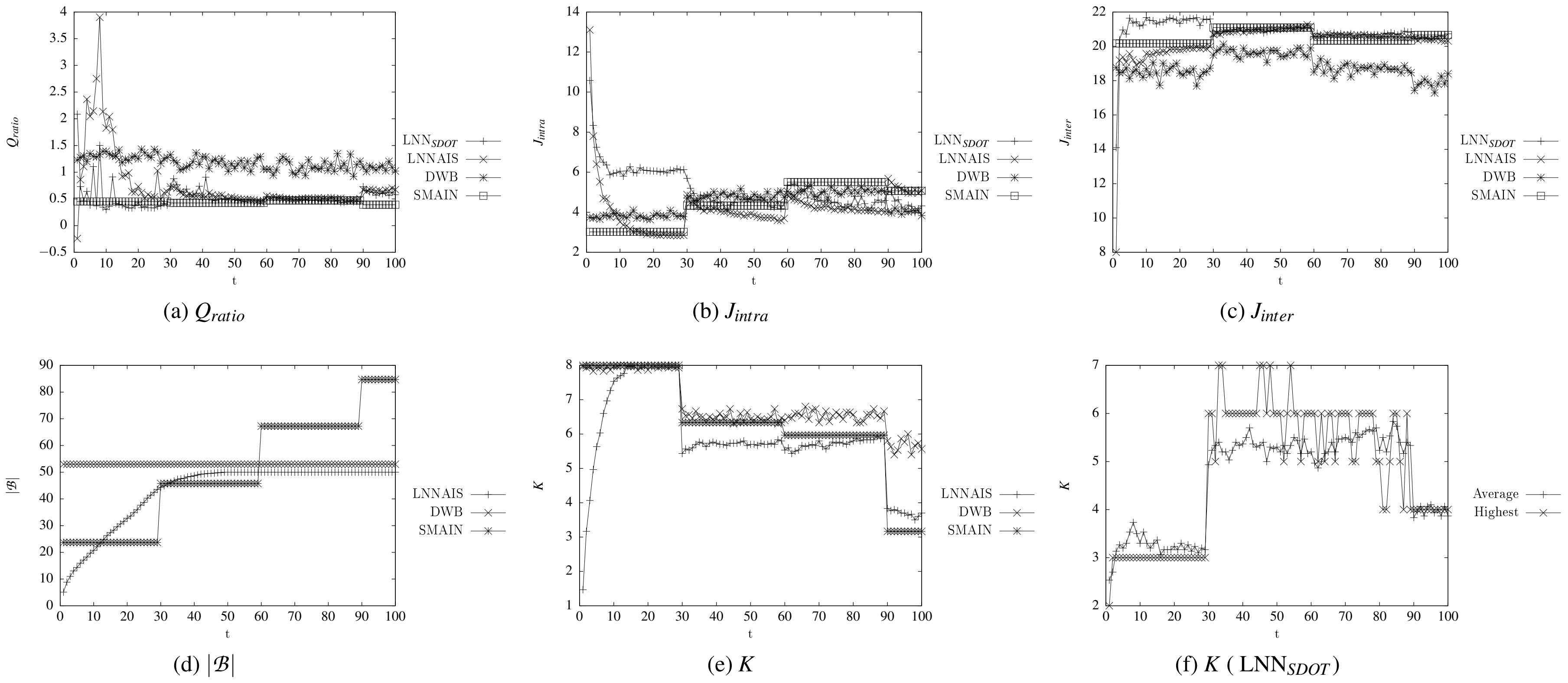


Figure 7.8 Centroid Migration ($N = 8, |C| = 25, \tilde{f} = 3, \tilde{s} = 3$): Quantifying each model's partitioning quality with regards to Q_{ratio}, $|\mathcal{B}|$, J_{intra}, J_{inter} and K

Table 7.32: Statistical Hypothesis Testing between LNNAIS and Other Models ($\alpha = 0.05$; with continuity correction; unpaired; non-directional) for Centroid migration at different severities of change (N=8; $|C|$=25; $\tilde{f}$=3)

$\tilde{s}$	LNN_{SDOT}	DWB	SMAIN
1	$z = 5.345$ $p < 0.001$ Reject H_0	$z = 6.646$ $p < 0.001$ Reject H_0	$z = 0.074$ $p = 0.941$ Accept H_0
2	$z = 5.98$ $p < 0.001$ Reject H_0	$z = 6.646$ $p < 0.001$ Reject H_0	$z = 6.527$ $p < 0.001$ Reject H_0
3	$z = 4.805$ $p < 0.001$ Reject H_0	$z = 6.646$ $p < 0.001$ Reject H_0	$z = 6.646$ $p < 0.001$ Reject H_0
4	$z = 4.273$ $p < 0.001$ Reject H_0	$z = 6.646$ $p < 0.001$ Reject H_0	$z = 5.781$ $p < 0.001$ Reject H_0
5	$z = 4.938$ $p < 0.001$ Reject H_0	$z = 6.646$ $p < 0.001$ Reject H_0	$z = 6.646$ $p < 0.001$ Reject H_0

The Mann-Whitney U statistical hypothesis test rejects H_0 that the mean clustering quality, $\bar{Q}$, are the same at a 0.05 level of significance between LNNAIS and LNN_{SDOT}, SMAIN and DWB for different dimensions (as summarised in table 7.29), different clusters sizes (as summarised in table 7.30), for all frequencies of change (as summarised in table 7.31) and severities of change greater than one (as summarised in table 7.32). There is thus a statistical significant difference in the clustering quality of all the centroid migration data sets between LNNAIS and all the other models except for $\tilde{s} = 1$ between LNNAIS and SMAIN where the Mann-Whitney U statistical hypothesis test accepted H_0.

Table 7.33: Descriptive Statistics: Centroid migration at different dimensions ($|C|$=25; $\tilde{f}$=3; $\tilde{s}$=3)

N		LNN$_{SDOT}$	LNNAIS	DWB	SMAIN
	K	3.4	5.69	6.73	5.63
		($\pm$)0.84	($\pm$)0.25	($\pm$)0.12	($\pm$)0.18
	J_{intra}	7.111	2.49	2.949	2.694
		($\pm$)2.193	($\pm$)0.246	($\pm$)0.078	($\pm$)0.163
3	J_{inter}	20.43	21.022	19.256	21.102
		($\pm$)1.01	($\pm$)0.3	($\pm$)0.22	($\pm$)0.127
	$\hat{Q}$	0.493	0.392	1.238	0.253
		($\pm$)0.1	($\pm$)0.121	($\pm$)0.067	($\pm$)0.015
	$\lvert\mathcal{B}\rvert$	49.18	49.18	95	60.01
		($\pm$)0.62	($\pm$)0.62	($\pm$)0	($\pm$)2.13
	K	4.73	5.94	6.84	6.36
		($\pm$)0.55	($\pm$)0.18	($\pm$)0.08	($\pm$)0.2
	J_{intra}	4.919	4.01	4.5	4.386
		($\pm$)0.621	($\pm$)0.198	($\pm$)0.093	($\pm$)0.11
8	J_{inter}	20.933	20.522	18.828	20.548
		($\pm$)0.316	($\pm$)0.196	($\pm$)0.245	($\pm$)0.136
	$\hat{Q}$	0.507	0.62	1.18	0.443
		($\pm$)0.101	($\pm$)0.092	($\pm$)0.065	($\pm$)0.011
	$\lvert\mathcal{B}\rvert$	45.97	45.97	53	50.09
		($\pm$)1.6	($\pm$)1.6	($\pm$)0	($\pm$)2.82
	K	5.56	5.98	6.66	6.06
		($\pm$)0.31	($\pm$)0.41	($\pm$)0.18	($\pm$)0.38
	J_{intra}	4.973	4.885	5.639	7.728
		($\pm$)0.258	($\pm$)0.315	($\pm$)0.118	($\pm$)1.248
15	J_{inter}	20.601	20.069	18.641	20.45
		($\pm$)0.29	($\pm$)0.51	($\pm$)0.281	($\pm$)0.775
	$\hat{Q}$	0.529	0.772	1.184	1.28
		($\pm$)0.056	($\pm$)0.114	($\pm$)0.064	($\pm$)0.43
	$\lvert\mathcal{B}\rvert$	28.94	28.94	68	28.22
		($\pm$)0.64	($\pm$)0.64	($\pm$)0	($\pm$)3.32

Table 7.34: Descriptive Statistics: Centroid migration at different cluster sizes (N=8; $\tilde{s}$=3; $\tilde{f}$=3)

$\|C\|$		LNN$_{SDOT}$	LNNAIS	DWB	SMAIN
	K	3.05	3.62	6.81	5.73
		(±)0.1	(±)0.09	(±)0.13	(±)0.14
	J_{intra}	7.143	6.531	4.562	4.61
		(±)0.178	(±)0.175	(±)0.084	(±)0.091
10	J_{inter}	21.198	19.536	18.374	19.971
		(±)0.252	(±)0.257	(±)0.222	(±)0.137
	$\hat{Q}$	0.511	0.894	1.131	0.479
		(±)0.113	(±)0.107	(±)0.048	(±)0.014
	$\|\mathcal{B}\|$	10	10	53	45.68
		(±)0.01	(±)0.01	(±)0	(±)2.22
	K	4.73	5.94	6.84	6.36
		(±)0.55	(±)0.18	(±)0.08	(±)0.2
	J_{intra}	4.919	4.01	4.5	4.386
		(±)0.621	(±)0.198	(±)0.093	(±)0.11
25	J_{inter}	20.933	20.522	18.828	20.548
		(±)0.316	(±)0.196	(±)0.245	(±)0.136
	$\hat{Q}$	0.507	0.62	1.18	0.443
		(±)0.101	(±)0.092	(±)0.065	(±)0.011
	$\|\mathcal{B}\|$	45.97	45.97	53	50.09
		(±)1.6	(±)1.6	(±)0	(±)2.82
	K	4.62	5.79	6.75	5.37
		(±)0.46	(±)0.26	(±)0.13	(±)0.19
	J_{intra}	4.774	3.958	4.492	4.532
		(±)0.593	(±)0.218	(±)0.087	(±)0.119
50	J_{inter}	21.195	20.542	18.53	20.706
		(±)0.283	(±)0.212	(±)0.321	(±)0.144
	$\hat{Q}$	0.441	0.542	1.241	0.489
		(±)0.05	(±)0.062	(±)0.049	(±)0.021
	$\|\mathcal{B}\|$	39.72	39.72	53	59.29
		(±)0.18	(±)0.18	(±)0	(±)2.83

Table 7.35: Descriptive Statistics: Centroid migration at different frequencies of change (N=8; $|C|$=25; $\tilde{s}$=3)

$\tilde{f}$		LNN_{SDOT}	LNNAIS	DWB	SMAIN
	K	5.28	5.66	5.88	5.24
		(±)0.46	(±)0.33	(±)0.17	(±)0.32
	J_{intra}	4.588	4.154	5.012	7.429
		(±)0.618	(±)0.254	(±)0.099	(±)0.437
1	J_{inter}	19.49	19.046	18.411	20.122
		(±)0.741	(±)0.54	(±)0.312	(±)0.615
	$\hat{Q}$	0.55	0.636	1.131	1.071
		(±)0.084	(±)0.096	(±)0.056	(±)0.172
	$\|\mathcal{B}\|$	29.5	29.51	95	29.31
		(±)0.4	(±)0.39	(±)0	(±)2.43
	K	4.51	5.36	6.3	4.81
		(±)0.35	(±)0.34	(±)0.14	(±)0.36
	J_{intra}	4.815	4.128	4.591	5
		(±)0.381	(±)0.301	(±)0.098	(±)0.289
2	J_{inter}	20.159	19.389	17.782	20.035
		(±)0.45	(±)0.691	(±)0.355	(±)0.452
	$\hat{Q}$	0.483	0.65	1.178	0.455
		(±)0.073	(±)0.105	(±)0.046	(±)0.019
	$\|\mathcal{B}\|$	29.51	29.52	53	56.85
		(±)0.36	(±)0.35	(±)0	(±)5.91
	K	4.73	5.94	6.84	6.36
		(±)0.55	(±)0.18	(±)0.08	(±)0.2
	J_{intra}	4.919	4.01	4.5	4.386
		(±)0.621	(±)0.198	(±)0.093	(±)0.11
3	J_{inter}	20.933	20.522	18.828	20.548
		(±)0.316	(±)0.196	(±)0.245	(±)0.136
	$\hat{Q}$	0.507	0.62	1.18	0.443
		(±)0.101	(±)0.092	(±)0.065	(±)0.011
	$\|\mathcal{B}\|$	45.97	45.97	53	50.09
		(±)1.6	(±)1.6	(±)0	(±)2.82

Continued on next page

$\tilde{f}$		LNN$_{SDOT}$	LNNAIS	DWB	SMAIN
	K	5.56	6.69	7.09	6.82
		(±)0.35	(±)0.26	(±)0.13	(±)0.3
	J_{intra}	4.535	3.807	5.03	3.949
		(±)0.434	(±)0.243	(±)0.103	(±)0.389
4	J_{inter}	21.091	20.623	19.153	20.657
		(±)0.201	(±)0.129	(±)0.215	(±)0.208
	$\hat{Q}$	0.448	0.544	1.251	0.415
		(±)0.134	(±)0.106	(±)0.069	(±)0.028
	$\|\mathcal{B}\|$	38.18	38.18	40	32.22
		(±)0.72	(±)0.72	(±)0	(±)4.88
	K	4.71	7.38	7.34	7.47
		(±)0.75	(±)0.24	(±)0.08	(±)0.09
	J_{intra}	5.581	3.49	4.634	3.521
		(±)0.919	(±)0.185	(±)0.074	(±)0.114
5	J_{inter}	21.024	20.229	18.954	20.36
		(±)0.209	(±)0.253	(±)0.159	(±)0.12
	$\hat{Q}$	0.469	0.652	1.203	0.42
		(±)0.096	(±)0.144	(±)0.044	(±)0.018
	$\|\mathcal{B}\|$	38.12	38.12	68	51.92
		(±)0.58	(±)0.58	(±)0	(±)3.76

Table 7.36: Descriptive Statistics: Centroid migration at different severities of change (N=8; $|C|$=25; $\tilde{f}$=3)

$\tilde{s}$		LNN$_{SDOT}$	LNNAIS	DWB	SMAIN
	K	4.59	7.81	7.47	7.35
		(±)0.86	(±)0.21	(±)0.11	(±)0.12
	J_{intra}	6.045	3.162	4.934	4.121
		(±)1.106	(±)0.143	(±)0.097	(±)0.083
1	J_{inter}	21.012	20.319	18.815	20.031
		(±)0.345	(±)0.139	(±)0.24	(±)0.182
	$\hat{Q}$	0.409	0.523	1.288	0.496

Continued on next page

$\tilde{s}$		LNN$_{SDOT}$	LNNAIS	DWB	SMAIN
		(±)0.031	(±)0.096	(±)0.066	(±)0.02
	$\lvert\mathcal{B}\rvert$	37.89	37.89	46	46.68
		(±)0.95	(±)0.95	(±)0	(±)2.43
	K	5.68	6.53	6.91	6.02
		(±)0.59	(±)0.31	(±)0.07	(±)0.35
	J_{intra}	4.576	3.806	4.842	4.441
		(±)0.677	(±)0.232	(±)0.097	(±)0.288
2	J_{inter}	20.175	19.618	18.485	19.986
		(±)0.38	(±)0.307	(±)0.177	(±)0.264
	$\hat{Q}$	0.461	0.623	1.21	0.456
		(±)0.054	(±)0.086	(±)0.047	(±)0.024
	$\lvert\mathcal{B}\rvert$	37.81	37.81	69	39.93
		(±)0.91	(±)0.91	(±)0	(±)3.75
	K	4.73	5.94	6.84	6.36
		(±)0.55	(±)0.18	(±)0.08	(±)0.2
	J_{intra}	4.919	4.01	4.5	4.386
		(±)0.621	(±)0.198	(±)0.093	(±)0.11
3	J_{inter}	20.933	20.522	18.828	20.548
		(±)0.316	(±)0.196	(±)0.245	(±)0.136
	$\hat{Q}$	0.507	0.62	1.18	0.443
		(±)0.101	(±)0.092	(±)0.065	(±)0.011
	$\lvert\mathcal{B}\rvert$	45.97	45.97	53	50.09
		(±)1.6	(±)1.6	(±)0	(±)2.82
	K	5.38	5.92	6.66	5.88
		(±)0.51	(±)0.21	(±)0.1	(±)0.24
	J_{intra}	4.778	4.422	5.043	5.282
		(±)0.61	(±)0.219	(±)0.099	(±)0.27
4	J_{inter}	20.589	20.286	18.759	20.661
		(±)0.337	(±)0.271	(±)0.24	(±)0.18
	$\hat{Q}$	0.491	0.585	1.204	0.466
		(±)0.097	(±)0.091	(±)0.06	(±)0.025
	$\lvert\mathcal{B}\rvert$	38.42	38.42	69	47.99

Continued on next page

$\tilde{s}$		LNN$_{SDOT}$	LNNAIS	DWB	SMAIN
		$(\pm)0.72$	$(\pm)0.72$	$(\pm)0$	$(\pm)2.84$
	K	4.91	6.1	6.59	5.67
		$(\pm)0.45$	$(\pm)0.25$	$(\pm)0.14$	$(\pm)0.15$
	J_{intra}	4.586	3.728	4.743	4.305
		$(\pm)0.426$	$(\pm)0.187$	$(\pm)0.075$	$(\pm)0.15$
5	J_{inter}	20.852	20.309	18.845	20.672
		$(\pm)0.348$	$(\pm)0.24$	$(\pm)0.252$	$(\pm)0.122$
	$\hat{Q}$	0.503	0.635	1.222	0.441
		$(\pm)0.076$	$(\pm)0.09$	$(\pm)0.046$	$(\pm)0.015$
	$\lvert\mathcal{B}\rvert$	38.15	38.15	55	56.85
		$(\pm)0.78$	$(\pm)0.78$	$(\pm)0$	$(\pm)3.54$

Again, due to the tendency of SMAIN to overfit the data, the quality of clusters found by SMAIN for centroid migration environments generally tends to be higher than the quality of the clusters found by LNNAIS (see tables 7.33 - 7.36 for centroid migration environments with different dimensions, clusters sizes, frequencies of change and severities of change). Note that in cases where the ALC population of SMAIN has a similar size as the ALC population of LNNAIS, LNNAIS tends to deliver cluster of a higher quality than SMAIN. This is shown in table 7.33 for $N = 15$ where $|\mathcal{B}| \approx 28$ and $K \approx 6$ for LNN$_{SDOT}$, LNNAIS and SMAIN. Note that with these parameter values LNNAIS tends to deliver clusters with a higher quality than SMAIN and LNN$_{SDOT}$ tends to deliver clusters of a higher quality than both SMAIN and LNNAIS. The advantage of LNN$_{SDOT}$ is that the clusters were dynamically determined. This is also shown in table 7.35 for $\tilde{f} = 1$ where $|\mathcal{B}| \approx 29$ and $K \approx 5$ for LNN$_{SDOT}$, LNNAIS and SMAIN. LNNAIS tends to deliver clusters with a higher quality than SMAIN and LNN$_{SDOT}$ tends to deliver clusters of a higher quality than both SMAIN and LNNAIS. In general, LNNAIS delivers clusters of a higher quality than DWB for all centroid migration environments. LNNAIS also obtains the correct number of clusters at different severities of change with no significant change in the ALC population size (see table 7.36 where an increase in $\tilde{s}$ increases the ratio of centroid migration, i.e. decreasing the number of clusters in the data). In general, where LNN$_{SDOT}$ obtained the correct number of clusters, the quality of the clusters tends to be higher than those clusters delivered by LNNAIS at different dimensions, cluster sizes, frequencies of change and severities of change (see tables 7.33- 7.36). Furthermore, as discussed above, LNN$_{SDOT}$ also tends to deliver clusters of higher quality than SMAIN and DWB in cases where the data is not overfitted by SMAIN and a similar number of clusters are obtained.

7.6 Conclusion

The chapter discussed and investigated different data migration types in a non-stationary environment. These migration types were pattern migration, cluster migration and centroid migration. A procedure to generate artificial non-stationary data sets with different environment parameters and migration types was proposed. Also, clustering performance measures for a non-stationary environment were proposed. The proposed clustering performance measures were used for comparison between four network based artificial immune system models for clustering of the generated artificial non-stationary data sets. These models were LNNAIS, LNN$_{SDOT}$, DWB and SMAIN. A sensitivity analysis of the LNNAIS parameters was done on the different artificial non-stationary data sets for each of the defined data migration types.

A sensitivity analysis of the LNNAIS parameters shows that for all migration types, LNNAIS utilises small population sizes with small cluster sizes and larger population sizes for large cluster sizes. There is also no effect on $\mathcal{B}_{max}$ with different frequencies or severities of change. The clustering quality of LNNAIS is the lowest at high frequencies and high severities of change for all of the migration environments at different dimensions and cluster sizes. The clustering quality of LNNAIS improves with an increase in the cluster size at different dimensions. Increasing the number of dimensions lowers the clustering quality of LNNAIS at different cluster sizes. A difference between the migration types is that LNNAIS utilises small and large population sizes at different dimensions for centroid migration environments; but, for the other migration types LNNAIS utilises small population sizes for high dimensional environments with small cluster sizes. Also, the frequency and severity of change in high dimensional centroid migration environments have a smaller effect on the clustering performance of LNNAIS when compared to pattern and cluster migration environments.

Overall, the SMAIN model tends to find clusters of a higher quality for all types of data migration environments (at the cost of overfitting the data), followed by LNNAIS. The higher quality of clusters found by SMAIN is due to a larger ALC population size which is utilised by SMAIN. The drawback of overfit by SMAIN is even more emphasised in cluster and centroid migration environments where the ALC population size of the SMAIN model does not scale with the number of clusters, since it is expected to utilise a smaller ALC population size with a decrease in the number of clusters in the data. LNNAIS delivers clusters of a higher quality than DWB and LNN$_{SDOT}$ for all pattern and cluster migration environments. In centroid migration environments, LNNAIS succeeds to recover from any changes and improve on the cluster quality as time progresses, even though the number of clusters changes and clusters become less compact. LNNAIS delivers clusters of a higher quality than DWB for centroid migration environments and in cases where the ALC population of SMAIN has a similar size as the ALC population of LNNAIS, LNNAIS also tends to deliver clusters of a higher quality than SMAIN. LNNAIS also obtains the correct number of clusters at different severities of change with no significant change in the ALC population size. Wherever LNN$_{SDOT}$ obtained the correct number of clusters, the quality of the clusters tends to be higher than those clusters delivered by LNNAIS at different dimensions, clusters sizes, frequencies of change and severities of change. Furthermore, LNN$_{SDOT}$ also tends to deliver clusters of higher quality than SMAIN and DWB in cases where the data is not overfitted by SMAIN and a similar number of clusters are obtained.

An advantage of LNNAIS and LNN$_{SDOT}$, compared to the other models, is that both models have less user specified parameters and are computationally less expensive since neither follows a hybrid approach like SMAIN and DWB to determine the number of ALC networks. A further advantage of LNN$_{SDOT}$ is that the clusters are dynamically determined. A drawback of the SMAIN model in non-stationary environments is the increase in the ALC population size with each change in the data. This drawback has a major impact on the scalability of the SMAIN model. A drawback of the LNN$_{SDOT}$ model is in cases where there are no outlier network affinities between ALC networks. The lack in outlier network affinities results in less network boundaries and therefore less ALC network (cluster) formations.

From the results presented, it can be concluded that LNNAIS, having a small set of control parameters, is an efficient clustering model for different non-stationary environments. LNN$_{SDOT}$ is most suitable for centroid migration environments where the number of clusters in the data is not known and needs to be quantified over time.

Chapter 8

Conclusion

This chapter briefly highlights the findings and contributions of this thesis and discusses directions for future research.

8.1 Summary

This thesis investigated the application of a network theory inspired artificial immune model to data clustering problems in stationary and non-stationary environments.

Chapter 5 presented a new network based artificial immune model, namely the local network neighbourhood AIS (LNNAIS). The proposed model utilises an index based network topology to determine the network connectivity between the artificial lymphocytes (ALCs). The application of LNNAIS to data clustering problems in stationary environments was investigated. The clustering performance of the LNNAIS model was compared against classical clustering algorithms (K-means clustering and CPSO) and existing network based AIS models (SMAIN, DWB and Opt-aiNet). In most cases, LNNAIS produced better or similar results with reference to the clustering quality, compactness and separation of the clusters. Although SMAIN tends to deliver clusters of a higher quality than LNNAIS, further investigation into the size of the ALC populations showed that SMAIN utilised a larger ALC population to cluster the data. This explained the superior clustering quality of SMAIN but also highlighted a potential drawback of SMAIN that tend to overfit the data. Compared to SMAIN in view of these findings, the LNNAIS model delivers clusters of high quality without overfitting the data. A sensitivity analysis was done on the parameters of LNNAIS. The results suggest that an increase in the ALC population size increases diversity which obtains the required number of clusters and improves the clustering quality. Smaller neighbourhood sizes deliver more compact and more separated clusters when compared to larger neighbourhood sizes, and also tend to obtain the required number of clusters. Therefore small neighbourhood sizes deliver clusters of a higher quality. Furthermore, the clonal level threshold influences the compactness of the clusters and is problem specific.

Chapter 6 presented two different techniques which can be used by LNNAIS to dynamically determine the number clusters in a data set. These techniques are the iterative pruning technique (IPT) and the sequential deviation outlier technique (SDOT). Both of these techniques are computationally less expensive than the multiple execution approaches to dynamically determine the number of clusters in a data set. The IPT technique is computationally slightly more expensive than SDOT since IPT needs to iterate through all possible edges (to a maximum of $\mathcal{B}_{max}$). A range for K can be specified, but this makes IPT parameter dependant. An advantage of IPT is that the technique can use any cluster validity index to determine the number of clusters. The SDOT technique neither uses a cluster validity index nor does it require any boundary constraints on K. SDOT is a non-parametric technique. This is an advantage, since it is not always feasible to visually inspect the formed clusters and a specified range for K might not contain the optimum number of clusters. Both techniques were applied on different data sets to determine the optimal number of clusters. These results were compared to the results obtained from K-means clustering which used the multiple execution approach to determine the optimal number of clusters in each data set. In general, LNNAIS using SDOT tends to deliver clusters of similar or higher quality for all data sets, followed by LNNAIS using IPT and K-means clustering. The influence of the different LNNAIS parameters (using SDOT) was then investigated.

Chapter 7 presented and investigated different data migration types in a non-stationary environment. These migration types were pattern migration, cluster migration and centroid migration. A procedure to generate artificial non-stationary data sets with different environment parameters and migration types was proposed. Also, clustering performance measures for a non-stationary environment were proposed. The proposed clustering performance measures were used for comparison between four network based artificial immune system models for clustering of the generated artificial non-stationary data sets. A sensitivity analysis of the LNNAIS parameters shows that for all migration types, LNNAIS utilises small population sizes with small cluster sizes and larger population sizes for large cluster sizes. The clustering quality of LNNAIS is the lowest at high frequencies and high severities of change for all of the migration environments at different dimensions and cluster sizes. The clustering quality of LNNAIS improves with an increase in the cluster size at different dimensions. Increasing the number of dimensions lowers the clustering quality of LNNAIS at different cluster sizes. The clustering performance of the LNNAIS model and the enhanced version utilising SDOT (LNN$_{SDOT}$) were compared against the clustering performance of SMAIN and DWB in non-stationary environments. The higher quality of clusters found by SMAIN when compared to LNNAIS is due to a larger ALC population size which is utilised by SMAIN and overfits the data. This is more emphasised in cluster and centroid migration environments where the ALC population size of the SMAIN model does not scale with the number of clusters. LNNAIS delivers clusters of a higher quality than DWB and LNN$_{SDOT}$ for all pattern and cluster migration environments. In centroid migration environments, LNNAIS succeeds to recover from any changes and improve on the cluster quality as time progresses, even though the number of clusters changes and clusters become less compact. LNNAIS delivers clusters of a higher quality than DWB for centroid migration environments and in cases where the data is not overfitted by SMAIN and the ALC population has similar sizes, LNNAIS also tends to deliver clusters of a higher quality than SMAIN. LNNAIS also obtains the correct number of clusters at different severities of change with no significant change in the ALC population size. Wherever LNN$_{SDOT}$ obtained the correct number of clusters, the quality of the clusters tends to be higher than those clusters delivered by LNNAIS at different dimensions, clusters sizes, frequencies of change and severities of change. Furthermore, LNN$_{SDOT}$ also tends to deliver clusters of higher quality than SMAIN and DWB in cases where the data is not overfitted by SMAIN and a similar number of clusters are obtained.

From the results presented in this thesis, it can be concluded that LNNAIS and LNN$_{SDOT}$ are efficient clustering models for different stationary and non-stationary environments. This is achieved even in light of the smaller set of control parameters compared to other network based AIS models. LNN$_{SDOT}$ can dynamically determine the number of clusters in a stationary data set and is most suitable for centroid migration non-stationary environments where the number of clusters in the data is not known and needs to be tracked over time.

8.2 Future Research

Several new directions for future research are briefly summarised below.

Decreasing neighbourhood sizes: Although the clustering performance of LNNAIS is the best at small neighbourhood sizes, a hybrid approach of a linear decrementing neighbourhood size needs to be investigated. An initial large neighbourhood size ($\rho = \mathcal{B}_{max}$) will have a more *greedy* approach to adapt the ALCs as one ALC network to the data patterns. More ALC networks are formed by linearly decreasing the neighbourhood size, which results into a more refined and specific search to the clusters in the data by different ALC networks. The model might initially prematurely adapt to the data, but eventually converge to different cluster centroids with the final set of ALC networks.

Alternative network neighbourhood topologies: The proposed LNNAIS in this thesis utilises a ring topology to determine the network connections between the ALCs. Although the clustering performance of LNNAIS at different neighbourhood sizes was investigated, future research needs to investigate the clustering performance of LNNAIS utilising different network topologies which includes rectangular grid (used in SOM), star and wheel (used in PSO) and Caylee trees. In addition to the investigation of the clustering performance of LNNAIS with different network topologies, the time of convergence and the coverage of the search space by the ALCs need to be investigated.

Hierarchical grouping: The clusters obtained by LNNAIS are represented by the formed ALC networks. The ALC networks are determined by pruning the network links between those ALCs with the lowest network affinity until the required number of clusters are obtained (or in the case of LNN$_{SDOT}$ the number of clusters is dynamically determined by the outlier network affinities). There is a potential risk that at the time of pruning the network links to determine the ALC network boundaries, an ALC might have been in the process of adapting to a neighbouring ALC. This can result into an ALC which has a low network affinity between the ALC's predecessor and an even lower network affinity with the ALC's successor in the population. When the adapting ALC is grouped with an ALC network, the calculated mean of the ALCs in that network might not represent the most appropriate centroid of the cluster in the data, since the ALC is becoming an outlier to the network of ALCs. This will have an impact on the clustering performance of LNNAIS. Therefore, a hierarchical agglomerative approach needs to be investigated to determine whether there is less influence of adapting ALCs to the calculated centroid of an ALC network. Another potential risk is that the adapting ALC can have equal network affinities between its neighbouring ALCs. These network affinities might be the lowest in the population resulting in an ALC network which consists of a single ALC and which does not contain any data patterns. These risks of the behaviour of ALCs need to be investigated.

LNN$_{SDOT}$ with IPT: Since the sequential deviation outlier technique (SDOT) used by LNNAIS depends on outlier network affinities between the ALC networks in order to dynamically determine the ALC network boundaries, a hybrid approach of SDOT and the iterative pruning technique (IPT) needs to be investigated. SDOT can be used to determine the initial number of clusters as a starting value of K for IPT. IPT can then increment and/or decrement the value of K with each iteration. The stopping criteria depend on whether the validity index used by IPT is a monotonic increasing or decreasing function. In the case of a monotonic increasing function, if the validity index decreases with an increment or decrement in the value of K, the search terminates. In the case of a monotonic decreasing function, if the validity index increases with an increment or decrement in the value of K, the search terminates. The search continues in both cases until the stopping criteria are met. The hybrid approach will dynamically determine the number of clusters with SDOT if outlier network affinities exist, otherwise the IPT technique is initialised with the result of SDOT and the search continues with the IPT technique.

Generating non-stationary environments: The generated non-stationary environments in this thesis contained clusters with fixed and equal spreads. For all the migration types defined, further investigation is needed into the clustering performance of the models on non-stationary environments where the spread of clusters changes with migrating patterns. This means that with each migrated pattern joining a cluster, the spread of the cluster should increase by a certain ratio. The same reasoning should be followed for patterns migrating from a cluster. The spread of clusters from which patterns migrate should decrease by a certain ratio. The dynamic spread of clusters will result in non-stationary environments with clusters which not only have different sizes (as those used in this thesis), but also different spreads and densities.

Image segmentation and classification problems: The proposed LNNAIS can be applied to the problem of image segmentation and classification problems. Since LNNAIS is an unsupervised learning algorithm, no changes are necessary to apply LNNAIS to image segmentation problems. The pixels of an image are then seen as the data set of antigen patterns. The ALC population of LNNAIS will adapt to these antigen patterns by forming ALC networks and eventually cluster the pixels of the image. Each cluster represents a segment of the image. In the context of non-stationary environments, a sequence of images of specific scenery can be segmented to identify any moving objects in the image. Focusing on classification problems, LNNAIS needs to be changed in such a way that ALCs are labeled with the same class labels as in the antigen set of patterns. This means that ALCs can then only adapt to antigen patterns of the same class. Eventually each of the formed ALC networks will represent a specific class in the data set of antigen patterns. This is a more semi-supervised learning approach of LNNAIS for classification problems.

Bibliography

[1] C. Aggarwal, A. Hinneburg, and D. Keim, "On the surprising behavior of distance metrics in high dimensional space," *Lecture Notes in Computer Science*, vol. 1973/2001, pp. 420–434, 2001.

[2] U. Aickelin, P. J. Bentley, S. Cayzer, J. Kim, and J. McLeod, "Danger theory: The link between ais and ids?" in *ICARIS*, vol. 2787. Edinburgh, UK: Springer, Sept. 2003, pp. 147–155.

[3] M. R. Anderberg, "Cluster analysis for applications," *Academic Press*, 1973.

[4] M. J. Antunes and M. E. Correia, "Towards a new immunity-inspired intrusion detection framework," Computer Science Department, Faculty of Science, University of Porto, Porto, Portugal, Tech. Rep. DCC-2006-4, Oct. 2006.

[5] A. Arning, R. Agrawal, and P. Raghavan, "A linear method for deviation detection in large databases," *Proceedings of Knowledge Discovery and Data Mining*, pp. 164–169, 1996.

[6] A. Asuncion and D. Newman, "UCI machine learning repository," 2007.

[7] M. Ayara, J. Timmis, R. de Lemos, L. N. de Castro, and R. Duncan, "Negative selection: How to generate detectors," *Proceedings of the 1st International Conference on Artificial Immune Systems (ICARIS)*, vol. 1, pp. 89–98, 2002.

[8] F. R. Bach and M. I. Jordan, "Learning spectral clustering," *Neural Information Processing Systems*, vol. 16, 2003.

[9] S. J. Baek, B. K. Jeon, D. Lee, and K. M. Sung, "Fast clustering algorithm for vector quantisation," *Electronics Letters*, vol. 34, pp. 151–152, 1998.

[10] G. H. Ball and D. J. Hall, "A clustering technique for summarizing multivariate data," *Behavioral Science*, vol. 12, no. 2, pp. 153–155, 1967.

[11] J. Balthrop, F. Esponda, S. Forrest, and M. Glickman, "Coverage and generalization in an artificial immune system," *Proceedings of GECCO*, pp. 3–10, 2002.

[12] J. Banchereau and R. M. Steinman, "Dendritic cells and the control of immunity," *Nature*, vol. 392, pp. 245–252, Mar. 1998.

[13] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty, *Nonlinear programming: theory and algorithms*. Wiley-interscience, 2006.

[14] R. Bellman, "Dynamic programming," *NJ: Princeton University Press*, 1957.

[15] P. Berkhin, "Survey of clustering data mining techniques, accrue software," *Inc. TR, San Jose, USA*, 2002.

[16] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*. Kluwer Academic Publishers Norwell, MA, USA, 1981.

[17] S. K. Bhatia, "Adaptive k-means clustering," *Proceedings of Florida Artificial Intelligence Research Symposium*, 2004.

[18] R. J. D. Boer, *In: Theoretical Immunology, Part Two*, A. Perelson, Ed. Addison-Wesley, Redwood City, CA, 1988.

[19] N. Bolshakova and F. Azuaje, "Estimating the number of clusters in dna microarray data," *Methods Inf. Med*, vol. 45, pp. 153–157, 2006.

[20] P. Bretscher and M. Cohn, "A theory of self-nonself discrimination," *Science (New York, N.Y.)*, vol. 169, pp. 1042–9, Sept. 1970.

[21] J. M. Buhmann, "Data clustering and learning," *The Handbook of Brain Theory and Neural Networks*, pp. 278–281, 2003.

[22] F. M. Burnet, *The Clonal Selection Theory of Acquired Immunity*. Nashville: Vanderbilt University Press, 1959.

[23] R. Canham, A. H. Jackson, and A. Tyrrell, "Robot error detection using an artificial immune system," *Proceedings of the 2003 NASA/DoD Conference on Evolvable Hardware*, 2003.

[24] B. Chen and C. Zang, “A hybrid immune model for unsupervised structural damage pattern recognition,” *Expert Systems with Applications*, vol. 38, no. 3, pp. 1650–1658, Mar. 2011.

[25] Y. Cheng and G. M. Church, “Biclustering of expression data,” in *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology table of contents*, 2000, pp. 93–103.

[26] Y. M. Cheung, “k*-means: A new generalized k-means clustering algorithm,” *Pattern Recognition Letters*, vol. 24, pp. 2883–2893, 2003.

[27] G. Coelho and F. V. Zuben, “A Concentration-Based artificial immune network for multi-objective optimization,” in *Evolutionary Multi-Criterion Optimization*, 2011, p. 343–357.

[28] G. P. Coelho, F. O. Franca, and F. J. Zuben, “A multi-objective multipopulation approach for biclustering,” in *Lecture Notes in Computer Science*, vol. 5132, 2008, pp. 71–82.

[29] ——, “Improving a Multi-Objective multipopulation artificial immune network for biclustering,” in *IEEE Congress on Evolutionary Computation*, 2009, pp. 2748–2755.

[30] D. E. Cooke and J. E. Hunt, “Recognising promoter sequences using an artificial immune system.” *Proc Int Conf Intell Syst Mol Biol*, vol. 3, pp. 89–97, 1995.

[31] D. L. Davies and D. W. Bouldin, “A cluster separation measure,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 1, pp. 224–227, 1979.

[32] L. N. de Castro and F. J. V. Zuben, “An evolutionary immune network for data clustering,” in *IEEE SBRN*, Rio de Janeiro, Nov. 2000, pp. 84–89.

[33] L. de Castro and J. Timmis, “An artificial immune network for multimodal function optimization,” in *Proceedings of the 2002 Congress on Evolutionary Computation CEC2002*. IEEE Press, 2002, pp. 699–704.

[34] ——, *Artificial Immune Systems: A New Computational Approach*. London, UK: Springer-Verlag, 2002.

[35] L. de Castro and F. V. Zuben, “The clonal selection algorithm with engineering applications,” in *Proceedings of the Genetic and Evolutionary Computational Conference (GECCO’00)*, 2000, pp. 36–37.

[36] ——, *AiNet: An Artificial Immune Network for Data Analysis*. USA: Idea Group Publishing, 2001, pp. 231–259.

[37] ——, “An immunological approach to initialize centers of radial basis function neural networks,” in *Proceedings of the Fifth Brazilian Conference on Neural Networks (CBRN’01)*, 2001, pp. 79–84.

[38] ——, “Learning and optimization using the clonal selection principle,” *IEEE Transactions on Evolutionary Computation, Special Issue on Artificial Immune Systems*, vol. 6, pp. 239–251, 2002.

[39] F. O. de França and F. J. V. Zuben, “A dynamic artificial immune algorithm applied to challenging benchmarking problems,” in *Proceedings of the 2009 conference on evolutionary computation*, 2009, pp. 423–430.

[40] F. O. de França, F. J. V. Zuben, and L. N. de Castro, “An artificial immune network for multimodal function optimization on dynamic environments,” in *Proceedings of the 2005 conference on Genetic and evolutionary computation*, 2005, pp. 289–296.

[41] A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum likelihood from incomplete data via the em algorithm,” *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 39, pp. 1–38, 1977.

[42] P. D’haeseleer, S. Forrest, and P. Helman, “An immunological approach to change detection: Algorithms, analysis and implications,” *Proceedings of the 1996 IEEE Symposium on Computer Security and Privacy*, vol. 1, pp. 110–120, 1996.

[43] M. d’Ocagne, “Coordonnées parallèles et axiales: Méthode de transformation géométrique et procédé nouveau de calcul graphique déduits de la considération des coordonnées parallèlles,” *Paris Gauthier-Villars*, 1885.

[44] J. C. Dunn, “Well separated clusters and optimal fuzzy partitions,” *Journal of Cybernetics*, vol. 4, pp. 95–104, 1974.

[45] R. Eberhart, P. Simpson, and R. Dobbins, *Computational intelligence PC tools*. Academic Press Professional, Inc. San Diego, CA, USA, 1996.

[46] R. C. Eberhart and Y. Shi, “Tracking and optimizing dynamic systems with particle swarms,” in *Proceedings of the 2001 Congress on Evolutionary Computation, 2001*, 2001, pp. 94–100.

[47] A. P. Engelbrecht, *Fundamentals of Computational Swarm Intelligence*. Wiley, 2005.

[48] ——, *Computational Intelligence: An Introduction*, 2nd ed. John Wiley & Sons, Oct. 2007.

[49] J. D. Farmer, N. H. Packard, and A. S. Perelson, "The immune system, adaptation, and machine learning," *Physica D*, vol. 22, no. 2, pp. 187–204, 1986.

[50] W. Feng, T. Brune, L. Chan, M. Chowdhury, C. K. Kuek, and Y. Li, "Benchmarks for testing evolutionary algorithms," in *Conference on Control and Measurement*, Dunhuang, P. R. China, 1998, pp. 134–138.

[51] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Annals of Eugenics*, vol. 7, pp. 179–188, 1936.

[52] E. Forgy, "Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications," *Biometrics*, vol. 21, p. 768, 1965.

[53] S. Forrest, A. S. Perelson, L. Allen, and R. Cherukuri, "Self-nonself discrimination in a computer," in *Proceedings of the 1994 IEEE Symposium on Research in Security and Privacy*. Oakland, CA: IEEE Computer Society Press, 1994, pp. 202–212.

[54] N. Franken, "Visual exploration of algorithm parameter space," in *2009 IEEE Congress on Evolutionary Computation*, Trondheim, Norway, 2009, pp. 389–398.

[55] T. Fukuda, K. Mori, and M. Tsukiyama, *Parallel search for multi-modal function optimization with diversity and learning of immune algorithm*. Springer, 1998.

[56] G. Fung, "A comprehensive overview of basic clustering algorithms," in *IEEE, June*, 2001.

[57] J. Gao, "A novel artificial immune system for solving multiobjective scheduling problems subject to special process constraint," *Computers & Industrial Engineering*, vol. 58, no. 4, pp. 602–609, May 2010.

[58] A. Gaspar and P. Collard, "Two models of immunization for time dependent optimization," *Systems, Man, and Cybernetics, 2000 IEEE International Conference on*, vol. 1, 2000.

[59] A. Gaspar and B. Hirsbrunner, "From optimization to learning in changing environments: The pittsburgh immune classier system," in *1st International Conference on Artificial Immune Systems*, vol. 1, Sept. 2002.

[60] F. Gonzalez, D. Dasgupta, and R. Kozma, “Combining negative selection and classification techniques for anomaly detection,” in *Proceedings of the Congress on Evolutionary Computation*, Hawaii, May 2002.

[61] A. J. Graaff, “The artificial immune system with evolved lymphocytes,” M.Sc. dissertation, University of Pretoria, 2003.

[62] A. J. Graaff and A. P. Engelbrecht, “Using a threshold function to determine the status of lymphocytes in the artificial immune system,” *Proceedings of the 2003 annual research conference of the South African institute of computer scientists and information technologists on Enablement through technology*, pp. 268–274, 2003.

[63] ——, “Optimised coverage of non-self with evolved lymphocytes in an artificial immune system,” *International Journal of Computational Intelligence Research*, vol. 2, pp. 127–150, 2006.

[64] ——, “A local network neighbourhood artificial immune system for data clustering,” in *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on*, 2007, pp. 260–267.

[65] L. Hagen and A. B. Kahng, “New spectral methods for ratio cut partitioning and clustering,” *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 11, no. 9, pp. 1074–1085, 2002.

[66] M. Halkidi, Y. Batistakis, and M. Vazirgiannis, “On clustering validation techniques,” *Journal of Intelligent Information Systems*, vol. 17, pp. 107–145, 2001.

[67] ——, “Cluster validity methods: part i,” *ACM SIGMOD Record*, vol. 31, pp. 40–45, 2002.

[68] ——, “Clustering validity checking methods: part ii,” *ACM SIGMOD Record*, vol. 31, pp. 19–27, 2002.

[69] M. Halkidi and M. Vazirgiannis, “Clustering validity assessment: Finding the optimal partitioning of a data set,” in *Proceedings of the 2001 IEEE International Conference on Data Mining*, 2001, pp. 187–194.

[70] G. Hamerly and C. Elkan, “Alternatives to the k-means algorithm that find better clusterings,” *Proceedings of the eleventh international conference on Information and knowledge management*, pp. 600–607, 2002.

[71] ——, "Learning the k in k-means. seventeenth annaul conference on neural information processing systems (nips)," *British Columbia, Canada*, 2003.

[72] ——, "Learning the k in K-Means," *The Seventh Annual Conference on Neural Information Processing Systems*, vol. 17, 2003.

[73] R. W. Hamming, "Error detecting and error correcting codes," *Bell Syst. Tech. J*, vol. 29, pp. 147–160, 1950.

[74] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 2001.

[75] D. Hanisch, A. Zien, R. Zimmer, and T. Lengauer, *Co-clustering of biological networks and gene expression data*, ser. BioInformatics. Oxford Univ Press, 2002, vol. 18.

[76] E. Hart, "Immunology as a metaphor for computational information processing: Fact or fiction," Ph.D. dissertation, University of Edinburgh, 2002.

[77] E. Hart and P. Ross, *Clustering moving data with a modified immune algorithm*. Springer: Boers E, 2001, pp. 394–404.

[78] ——, "Exploiting the analogy between immunology and sparse distributed memories: A system for clustering non-stationary data," *Proceedings of the First International Conference on Artificial Immune Systems (ICARIS-2002)*, pp. 49–58, 2002.

[79] E. Hart, P. Ross, and J. Nelson, "Producing robust schedules via an artificial immune system," *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, pp. 464–469, 1998.

[80] J. A. Hartigan, "Direct clustering of a data matrix," *Journal of the American Statistical Association*, pp. 123–129, 1972.

[81] V. Hautamäki, S. Cherednichenko, I. Kärkkäinen, T. Kinnunen, and P. Fränti, "Improving k-means by outlier removal," *Lecture notes in computer science*, pp. 978–987, 2005.

[82] T. A. Hely, D. J. Willshaw, and G. M. Hayes, "A new approach to kanerva's sparse distributed memory," *Neural Networks, IEEE Transactions on*, vol. 8, pp. 791–794, 1997.

[83] J. Herrero, A. Valencia, and J. Dopazo, *A hierarchical unsupervised growing neural network for clustering gene expression patterns*, ser. BioInformatics. Oxford Univ Press, 2001, vol. 17.

[84] J. Hiernaux, "Some remarks on the stability of the idiotypic network," *Immunochemistry*, vol. 14, pp. 733–739, Dec. 1977.

[85] S. Hofmeyr, "An immunological model of distributed detection and its application to computer security," Ph.D. dissertation, University of New Mexico, 1999.

[86] P. W. Holland, J. Garcia-Fernandez, N. A. Williams, and A. Sidow, "Gene duplications and the origins of vertebrate development." *Development (Cambridge, England). Supplement*, p. 125, 1994.

[87] K. Y. Huang, "A synergistic automatic clustering technique (SYNERACT) for multispectral image analysis," *Photogrammetric engineering and remote sensing*, vol. 68, no. 1, pp. 33–40, 2002.

[88] L. Hubert and J. Schultz, "Quadratic assignment as a general data analysis strategy," *British Journal of Mathematical and Statistical Psychology*, vol. 29, pp. 190–241, 1976.

[89] J. Hunt, C. King, and D. Cooke, "Immunising against fraud," in *Knowledge Discovery and Data Mining and IEE Colloquium*. IEEE, 1996, pp. 38–45.

[90] J. Hunt, J. Timmis, D. Cooke, M. Neal, and C. King, "Jisys: The development of an artificial immune system for real world applications," *Artificial Immune Systems and their Applications. Ed. D. Dasgupta*, pp. 157–186, 1998.

[91] J. E. Hunt, D. E. Cooke, and H. Holstein, "Case memory and retrieval based on the immune system [m]. in (veloso mm, aamodt a., eds.) case-based reasoning research and development (proc. iccbr-95)," *Springer, LNAI*, vol. 1010, pp. 205–216, 1995.

[92] J. E. Hunt and A. Fellows, "Introducing an immune response into a cbr system for data mining," *BCS ESG'96 Conference and published as Research and Development in Expert Systems XIII*, pp. 35–42, 1996.

[93] J. E. Hunt and D. E. Cooke, "Learning using an artificial immune system," *Journal of Network and Computer Applications*, vol. 19, pp. 189–212, 1996.

[94] A. Inselberg, "The plane with parallel coordinates," *Visual Computer 1*, vol. 4, pp. 69–91, 1985.

[95] A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*. Englewood Cliffs, New Jersey: Prentice-Hall, 1988.

[96] A. Jain, M. Murty, and P. Flynn, "Data clustering: a review," *ACM Comput. Surv.*, vol. 31, pp. 264–323, Sept. 1999.

[97] N. K. Jerne, "Towards a network theory of the immune system," *Annals of Immunology (Inst. Pasteur)*, vol. 125C, pp. 373–89, 1974.

[98] ——, "The generative grammar of the immune system," *The European Molecular Biology Organization journal*, vol. 4, pp. 847–52, Apr. 1985.

[99] Z. Ji and D. Dasgupta, "Real-valued negative selection algorithm with variable-sized detectors," *LNCS*, vol. 3102, pp. 287–298, 2004.

[100] S. Jianbo and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[101] Y. Jin and B. Sendhoff, *Constructing Dynamic Optimization Test Problems Using the Multi-objective Optimization Concept*. Springer, 2004, pp. 525–536.

[102] K. A. D. Jong, "An analysis of the behavior of a class of genetic adaptive systems." Ph.D. dissertation, University of Michigan, 1975.

[103] S. Jonnalagadda and R. Srinivasan, "An information theory approach for validating clusters in microarray data." in *Joint Conference Intelligent Systems for Molecular Biology (ISMB) and European Conference on Computational Biology (ECCB)*, Glasgow, UK, 2004.

[104] J. Jun, D. Lee, and K. Sim, "Realization of cooperative swarm behavior in distributed autonomous robotic systems using artificial immune system," in *Proceedings of IEEE international conference on systems, man and cybernetics*, vol. 6, Tokyo, Japan, Nov. 1999, pp. 614–619.

[105] P. Kanerva, *Sparse Distributed Memory*. MIT Press, 1988.

[106] R. E. Kass and L. Wasserman, "A reference bayesian test for nested hypotheses and its relationship to the schwarz criterion," *Journal of the American Statistical Association*, vol. 90, no. 431, pp. 928–934, 1995.

[107] L. Kaufman and P. J. Rousseeuw, "Finding groups in data: An introduction to cluster analysis," *New York*, 1990.

[108] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Neural Networks, 1995. Proceedings., IEEE International Conference on*, vol. 4, 1995, pp. 1942–1948.

[109] J. F. Kennedy, R. C. Eberhart, and Y. Shi, *Swarm Intelligence*. NetLibrary, Incorporated, 2001.

[110] J. Kim and P. Bentley, "Negative selection and niching by an artificial immune system for network intrusion detection," in *Genetic and Evolutionary Computation Conference (GECCO '99)*, Orlando, Florida, July 1999, pp. 149–158.

[111] ——, "An evaluation of negative selection in an artificial immune system for network intrusion detection," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, San Francisco, California, USA, 2001, pp. 1330–1337.

[112] ——, "Immune memory in the dynamic clonal selection algorithm," in *Proceedings of the 1st International Conference on Artificial Immune Systems (ICARIS)*, vol. 1, University of Kent at Canterbury, Sept. 2002, pp. 59–67.

[113] ——, "A model of gene library evolution in the dynamic clonal selection algorithm," in *Proceedings of the 1st International Conference on Artificial Immune Systems (ICARIS)*, vol. 1, University of Kent at Canterbury, Sept. 2002, pp. 182–189.

[114] ——, "Towards an artificial immune system for network intrusion detection: An investigation of dynamic clonal selection," in *Proceedings of Congress on Evolutionary Computation*, 2002, pp. 1015–1020.

[115] T. Knight and J. Timmis, "Aine: An immunological approach to data mining," *IEEE International Conference on Data Mining*, pp. 297–304, 2001.

[116] ——, "A multi-layered immune inspired approach to data mining," in *RASC*, Nottingham, UK., Dec. 2002, pp. 266–271.

[117] E. M. Knorr and R. T. Ng, "Algorithms for mining distance-based outliers in large datasets," in *Proceedings of the 24th International Conference on Very Large Data Bases*. Morgan Kaufmann Publishers Inc., 1998, pp. 392–403.

[118] T. Kohonen, *Self-Organizing Maps*. Springer, 2001.

[119] F. Kovács, C. Legány, and A. Babos, "Cluster validity measurement techniques," in *6th International Symposium of Hungarian Researchers on Computational Intelligence*, Budapest, Nov. 2005.

[120] K. J. Lafferty and A. J. Cunningham, "A new analysis of allogeneic interactions," *Immunol Cell Biol*, vol. 53, pp. 27–42, Feb. 1975.

[121] C. Laurentys, G. Ronacher, R. Palhares, and W. Caminhas, "Design of an artificial immune system for fault detection: A negative selection approach," *Expert Systems with Applications*, vol. 37, no. 7, pp. 5507–5513, July 2010.

[122] C. Y. Lee and E. K. Antonsson, "Dynamic partitional clustering using evolutionary strategies," *Proc. of the 3rd Asia-Pacific Conference on Simulated Evolution and Learning*, 2000.

[123] Y. J. Liu, G. D. Johnson, J. Gordon, and I. C. MacLennan, "Germinal centres in t-cell-dependent antibody responses." *Immunol Today*, vol. 13, pp. 17–21, 1992.

[124] P. Matzinger, "Essay 1: The danger model in its historical context," *Scandinavian Journal of Immunology*, vol. 54, pp. 4–9, July 2001.

[125] ——, "The real function of the immune system or tolerance and the four d's (danger, death, destruction and distress)," 2004.

[126] S. McCormick, M. Santosuosso, X. Z. Zhang, and Z. Xing, "Manipulation of dendritic cells for host defence against intracellular infections," *Biochemical Society transactions*, vol. 34, pp. 283–286, Apr. 2006.

[127] B. Meyer, H. Meij, S. Grey, and A. Meyer, *Fisiologie van die Mens - Biochemiese, Fisiese en Fisiologiese Begrippe*, 1st ed. Cape Town: Kagiso Tersier, 1996.

[128] A. W. Moore and D. Pelleg, "X-means: Extending k-means with efficient estimation of the number of clusters," in *Proceedings of the Seventeenth International Conference on Machine Learning*. San Francisco, CA: Morgan Kaufmann, 2000, pp. 727–734.

[129] K. Mori, M. Tsukiyama, and T. Fukuda, "Multi-optimization by immune algorithm with diversity and learning," *Proc. Second Int. Conf. Multiagent Systems*, pp. 118–123, 1996.

[130] ——, "Adaptive scheduling system inspired by immune system," *Systems, Man, and Cybernetics, 1998. 1998 IEEE International Conference on*, vol. 4, 1998.

[131] R. W. Morrison, "Performance measurement in dynamic environments," *GECCO Workshop on Evolutionary Algorithms for Dynamic Optimization Problems, J. Branke, Ed*, pp. 5–8, 2003.

[132] F. Murtagh, "Clustering in massive data sets," *Handbook of Massive Data Sets*, pp. 401–545, 2002.

[133] N. Nanas and A. Roeck, "A review of evolutionary and immune-inspired information filtering," *Natural Computing*, vol. 9, no. 3, pp. 545–573, 2009.

[134] O. Nasraoui, C. Cardona, C. Rojas, and F. Gonzalez, "Mining evolving user profiles in noisy web clickstream data with a scalable immune system clustering algorithm," in *Workshop Notes of WEBKDD 2003: Web Mining as Premise to Effective&Intelligent*, 2003, pp. 71–81.

[135] O. Nasraoui, C. Cardona-Uribe, and C. Rojas-Coronel, "Tecno-streams: Tracking evolving clusters in noisy data streams with a scalable immune system learning model," in *IEEE International Conference on Data Mining*, Melbourne, Florida, Nov. 2003.

[136] O. Nasraoui, D. Dasgupta, and F. Gonzalez, "A novel artificial immune system approach to robust data mining," *Late Breaking Papers of the Genetic and Evolutionary Computation Conference (GECCO),(New York)*, pp. 356–363, 2002.

[137] ——, "The promise and challenges of artificial immune system based web usage mining: Preliminary results," *proceedings of The SIAM Workshop on Web Analytics*, pp. 29–39, 2002.

[138] O. Nasraoui, F. Gonzalez, C. Cardona, C. Rojas, and D. Dasgupta, "A scalable artificial immune system model for dynamic unsupervised learning," in *Genetic and Evolutionary Computation – GECCO-2003*, vol. 2723. Chicago: Springer-Verlag, July 2003, pp. 219–230.

[139] O. Nasraoui, F. Gonzalez, and D. Dasgupta, "The fuzzy artificial immune system: motivations, basic concepts, and application to clustering and web profiling," *Fuzzy Systems, 2002. FUZZ-IEEE'02. Proceedings of the 2002 IEEE International Conference on*, vol. 1, 2002.

[140] M. Neal, "An artificial immune system for continuous analysis of time-varying data," in *ICARIS*, vol. 1. University of Kent, 2002, pp. 76–85.

[141] M. Neal, J. Hunt, and J. Timmis, "Augmenting an artificial immune network," *Systems, Man, and Cybernetics, 1998. 1998 IEEE International Conference on*, vol. 4, pp. 3821–3826, 1998.

[142] M. Neal, "Meta-stable memory in an artificial immune network," *In Artificial Immune Systems: Proceedings of ICARIS 2003*, pp. 168–180, 2003.

[143] A. Ng, M. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Advances in Neural Information Processing Systems 14: Proceeding of the 2001 Conference*, 2001, pp. 849–856.

[144] S. Ohno, *Evolution by gene duplication.* Springer, 1970.

[145] M. Omran, A. Engelbrecht, and A. Salman, "Particle swarm optimization method for image clustering," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 19, pp. 297–322, 2005.

[146] J. K. Percus, O. E. Percus, and A. S. Perelson, "Predicting the size of the t-cell receptor and antibody combining region from consideration of efficient self-nonself discrimination." *Proceedings of the National Academy of Sciences of the United States of America*, vol. 90, p. 1691, 1993.

[147] A. S. Perelson and G. F. Oster, "Theoretical studies of clonal selection: minimal antibody repertoire size and reliability of self-non-self discrimination," *Journal of Theoretical Biology*, vol. 81, pp. 645–670, Dec. 1979.

[148] A. S. Perelson, "Immune network theory," *Immunological Review*, vol. 110, pp. 5–36, 1989.

[149] A. S. Perelson and G. Weisbuch, "Immunology for physicists," *Reviews of Modern Physics*, vol. 69, p. 1219, Oct. 1997.

[150] M. Potter and K. D. Jong, "The coevolution of antibodies for concept learning," in *Proceedings of the Fifth International Conference on Parallel Problem Solving from Nature (PPSN V)*, 1998, pp. 530–539.

[151] S. Ray and R. H. Turi, "Determination of number of clusters in k-means clustering and application in colour image segmentation," *The 4th International Conference on Advances in Pattern Recognition and Digital Techniques, Calcuta*, 1999.

[152] P. Richter, “A network theory of the immune system.” *European Journal of Immunology*, vol. 5, pp. 350–354, 1975.

[153] ——, *The network idea and the immune response*. New York: Marcel Dekker, 1978, pp. 539–569.

[154] C. Rosenberger and K. Chehdi, “Unsupervised clustering method with optimal estimation of the number of clusters: Application to image segmentation,” in *International Conference on Pattern Recognition*, vol. 1, 2000, pp. 1656–1659.

[155] S. Sarafijanovic and J.-Y. L. Boudec, “An artificial immune system for misbehavior detection in mobile ad-hoc networks with virtual thymus, clustering, danger signal, and memory detectors,” in *ICARIS*, ser. Lecture Notes in Computer Science, vol. 3239. Catania, Sicily, Italy: Springer, Sept. 2004, pp. 342–356.

[156] S. M. Savaresi, D. L. Boley, S. Bittanti, and G. Gazzaniga, “Cluster selection in divisive clustering algorithms,” *Proceedings of the 2 ndSIAM*, vol. 1401, pp. 299–314, 2002.

[157] L. Schindler, D. K. M.S., J. Kelly, and B. Hollen, “Understanding the immune system - national cancer institute,” 2002.

[158] A. Secker, A. A. Freitas, and J. Timmis, “A danger theory inspired approach to web mining,” in *ICARIS*, ser. Lecture Notes in Computer Science, vol. 2787. Edinburgh, UK: Springer, Sept. 2003, pp. 156–167.

[159] A. Sehgal and M. S. Berger, “Basic concepts of immunology and neuroimmunology,” *Neurosurg Focus*, vol. 9, pp. 1–6, 2000.

[160] Y. Shi and R. Eberhart, *Parameter selection in particle swarm optimization*. Springer, 1998, pp. 591–600.

[161] D. J. Smith, S. Forrest, and A. S. Perelson, “Immunological memory is associative,” *Workshop Notes, Workshop 4: Immunity Based Systems, Intnl. Conf. on Multiagent Systems*, pp. 62–70, 1998.

[162] J. A. Snyman, *Practical Mathematical Optimization*. Springer, 2005.

[163] R. M. Steinman, “Dendritic cells: versatile controllers of the immune system,” *Nat Med*, vol. 13, pp. 1155–1159, Oct. 2007.

[164] N. Tang and V. R. Vemuri, “An artificial immune system approach to document clustering,” in *SAC ’05: Proceedings of the 2005 ACM symposium on Applied computing*. New York, NY, USA: ACM Press, 2005, pp. 918–922.

[165] J. Timmis, “Artificial immune systems: A novel data analysis technique inspired by the immune network theory,” Ph.D. dissertation, University of Wales, Aug. 2000.

[166] ——, “aivis: Artificial immune network visualisation,” *EuroGraphics UK 2001 Conference Proceedings*, pp. 61–69, 2001.

[167] J. Timmis and C. Edmonds, “A comment on opt-ainet: An immune network algorithm for optimisation,” *Lecture Notes in Computer Science*, pp. 308–317, 2004.

[168] J. Timmis and T. Knight, “Artificial immune systems: Using the immune system as inspiration for data mining,” *Data Mining: A Heuristic Approach*, pp. 209–230, 2002.

[169] J. Timmis and M. Neal, “A resource limited artificial immune system for data analysis,” in *Research and Development in Intelligent Systems XVII*, vol. 14. Cambridge, UK.: Springer, Dec. 2000, pp. 19–32.

[170] M. D. G. Toledo, “A comparison in cluster validation techniques,” Master Thesis, University of Puerto Rico, Dec. 2005.

[171] C. Tomasi, “Estimating gaussian mixture densities with em - a tutorial,” *Duke University*, 2005.

[172] J. T. Tou, “DYNOC—A dynamic optimal cluster-seeking technique,” *International Journal of Parallel Programming*, vol. 8, no. 6, pp. 541–547, 1979.

[173] R. H. Turi, “Clustering-based colour image segmentation,” Ph.D. dissertation, Monash University, Australia, 2001.

[174] D. W. van der Merwe and A. P. Engelbrecht, “Data clustering using particle swarm optimization,” *Evolutionary Computation, 2003. CEC’03. The 2003 Congress on*, vol. 1, 2003.

[175] C. J. Veenman, M. J. Reinders, and E. Backer, “A maximum variance cluster algorithm,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 9, pp. 1273–1280, 2002.

[176] C. S. Wallace and D. L. Dowe, “Intrinsic classification by MML-the snob program,” in *Proceedings of the 7th Australian Joint Conference on Artificial Intelligence*, Armidale, NSW, Australia, 1994, pp. 37–44.

[177] Y. Watanabe, A. Ishiguro, and Y. Uchikawa, *Decentralized Behavior Arbitration Mechanism for Autonomous Mobile Robot using Immune Network*. Springer, 1998, pp. 187–209.

[178] A. Watkins and J. Timmis, “Artificial immune recognition system (airs): Revisions and refinements,” in *Proceedings of the 1st International Conference on Artificial Immune Systems (ICARIS)*, vol. 1, University of Kent at Canterbury, Sept. 2002, pp. 173–181.

[179] A. Watkins, J. Timmis, and L. Boggess, “Artificial immune recognition system (airs): An immune-inspired supervised machine learning algorithm,” *Genetic Programming and Evolvable Machines*, vol. 5, pp. 291–317, Sept. 2004.

[180] K. Weicker, “Performance measures for dynamic environments,” *Parallel Problem Solving from Nature-Ppsn VII: 7th International Conference, Granada, Spain, September 7-11, 2002: Proceedings*, 2002.

[181] G. Weisbuch, R. J. D. Boer, and A. S. Perelson, “Localized memories in idiotypic networks,” *Journal of Theoretical Biology*, vol. 146, pp. 483–499, Oct. 1990.

[182] S. T. Wierchoń, “Generating antibody string in an artificial immune system,” Polish Academy of Sciences, 1999.

[183] ——, “Discriminative power of the receptors activated by k-contiguous bits rule,” *Journal of Computer Science and Technology*, vol. 1, pp. 1–13, 2000.

[184] S. Wierchoń and U. Kużelewska, “Stable clusters formation in an artificial immune system,” in *Proceedings of the 1st International Conference on Artificial Immune Systems (ICARIS)*, vol. 1, University of Kent at Canterbury, Sept. 2002, pp. 68–75.

[185] Z. Xiang, “Color image quantization by minimizing the maximum intercluster distance,” *ACM Transactions on Graphics (TOG)*, vol. 16, pp. 260–276, 1997.

[186] C. Xie, Z. Chen, and X. Yu, “Sequence outlier detection based on chaos theory and its application on stock market,” *Fuzzy Systems and Knowledge Discovery*, pp. 1221–1228, 2006.

[187] L. V. Yang, L. I. U. Ping, and T. Qi-feng, "3D model retrieval algorithm based on artificial immune clustering," *Modern Computer*, 2011.

[188] L. A. Zadeh, "Fuzzy sets," *Information Control*, vol. 8, pp. 338–353, 1965.

[189] Y. Zhao and G. Karypis, "Empirical and theoretical comparisons of selected criterion functions for document clustering," *Machine Learning*, vol. 55, no. 3, pp. 311–331, 2004.

Appendix A

List of Symbols

This appendix lists symbols used within this thesis.

Chapter 2 – Clustering and Quality Measures

P	set of patterns that needs to be partitioned	9
N	number of dimensions	9
$\mathbf{p}_i$	i-th pattern in P	9
I	size of P	9
K	number of clusters	9
C_k	k-th cluster	9
σ	distance based similarity measure	10
Γ	cosine similarity measure	11
$\mathbf{Z}$	covariance matrix for the distribution of feature vectors	11
$\mathbf{p}_i^T$	transpose of $\mathbf{p}_i$	11
$\mathbf{p}_{i,n}$	n-th attribute of i-th pattern in P	12
$\oplus$	exclusive-or between two bits	12
ℓ	linking distance measure	14
$\mathbf{c}_k$	centroid of cluster C_k	16
J_X	function measuring different quantization errors	15
$\mathbf{M}$	matrix of membership degrees	17
$m_{i,k}$	function measuring the degree of membership	17
ϕ	radius of influence	18
C	set of K centroids	18

G	sum of individual densities	18
g	Gaussian density function	19
χ_k	mixing probability for C_k	19
Ω_k	mean vector of C_k	19
ξ_k	distribution parameters for C_k. $\xi_k = (\rho_k, \mu_k, \mathbf{Z}_k)$	19
Ξ	set of ξ	19
Q_X	cluster validity index of X	22
υ	diameter of a cluster	22
E_i	i-th execution of an algorithm	22
C_k^i	k-th cluster in E_i	22
inf	information gain/loss	23
M	magnitude of change	23
d	direction of M	23
$\mathcal{E}$	ascending ordered set of feature vector pair distances	24
e	$e \in \mathcal{E}$; feature vector pair distance	25
$\mathcal{S}_{min}$	sum of a number of smallest feature vector pair distances	24
$\mathcal{S}_{max}$	sum of a number of largest feature vector pair distances	24
ψ	variance of a set of vectors	25
ι	average standard deviation of all clusters	26
n	function which determines whether a feature vector is within the neighbourhood of another vector	26
$inter_{min}$	minimum separation between clusters	27
ω	control parameters for non-stationary environment	29
f	objective (fitness) function	32
$best\,(t)$	best candidate solution in a population at time t	32
$f_{max}\,(t)$	best fitness value in search space at time t	32
$f_{min}\,(t)$	worst fitness value in search space at time t	32
$accuracy\,(t)$	accuracy of $best\,(t)$ at time t	32
$stab\,(t)$	stability of an algorithm at time t	32
$sens\,(t)$	sensitivity of an algorithm to a change at time t	33
$react_{\varepsilon}\,(t)$	time taken to relocate $best\,(t)$ within a minimum time-ratio of ε	33
T	number of time steps	33
$currentBest\,(t)$	fitness of $best\,(t)$	34
$currentBestOff\,(t)$	maximum of $currentBest\left(t^{'}\right)$ where $1 \leq t^{'} \leq t$	34
$currentAvg\,(t)$	the average accuracy of a population at time t	34

$windowAcc(t)$	the *best* (t) within a certain time-span	34
cmf	collective mean fitness	35
o	set of outlier feature vectors	36
S_o	subset of potential outlier feature vectors	38
sf	smoothing factor	38
D	function of dissimilarity	38
S_e	exception set of feature vectors with highest sf value	38
S	swarm or population of particles	39
$\mathbf{x}_i$	current feature vector or position of the i-th particle in S	39
$\mathbf{b}_i$	the *personal best position* of the i-th particle in S	39
$\mathbf{v}_i$	current velocity of the i-th particle in S	39
$\mathbf{V}_{max}$	the maximum allowed velocity of any particle in S	39
$\mathbf{g}_i$	position of the *best* particle in the *neighbourhood* of the i-th particle in S	39
w	inertia weight	40
c_1, c_2	acceleration constants	40
δ	ratio of $\mathbf{V}_{max}$	41
$\mathbf{w}_{rc}$	codebook vector at row r and column c	43
$\mathbf{w}^{'}$	best matching neuron	44
Λ	neighbourhood of $\mathbf{w}^{'}$	44
γ	learning rate	44

Chapter 4 – Artificial Immune Systems

$\mathcal{B}$	ALC/ARB/antibody population	67
$\mathcal{A}$	antigen population	67
$\mathcal{S}$	set of selected ALCs ($\mathcal{S} \subseteq \mathcal{B}$)	68
$\mathcal{H}$	set of selected ALCs with highest affinity ($\mathcal{H} \subseteq \mathcal{S}$)	68
ϖ	shape space (search space)	70
V	total volume of ϖ	70
$\mathbf{b}$	ALC in $\mathcal{B}$	70
$\mathbf{a}$	antigen pattern in $\mathcal{A}$	70
r	antigen affinity threshold	70

v	average volume in ϖ with radius ϕ 70
$\mathbf{x}$	antigen detector 74
Υ	set of self patterns 74
κ	alphabet size 75
$\mathbf{x}^*$	length-limited/crossover hole 74
$W_i^{\mathbf{x}}$	template window of pattern **x** starting at position *i* in **x** 74
$\mathbf{s}$	self pattern 76
Θ	multiplying factor 82
$\mathcal{F}$	free-antibodies population 84
$\mathcal{M}$	memory ALC population 84
$\mathbf{b}^*$	clone of **b** 85
$\mathbf{b}'$	mutated $\mathbf{b}^*$ 85
η	number of clones 85
σ_{max}	maximum distance/affinity between two patterns 85
$\mathbf{m}_{min}$	memory ALC with lowest σ 85
ν	concentration level 87
NAT	network affinity threshold 90
ϑ	stimulation level 90
ps	antigen stimulation level 90
ns	network stimulation level 90
nn	network suppression level 90
$\alpha_{\mathbf{b}}$	set of antigen stimulation levels with ALC **b** 90
$\lambda_{\mathbf{b}}$	set of network stimulation levels with ALC **b** 90
R	number of resources allocated to an ALC 92
R_k	resource constant 92
ϑ'	normalised stimulation level 92
R_γ	decaying rate of resources 94
R_Λ	mortality threshold 94
$\mathbf{h}$	highest stimulated ALC 94
$\mathcal{B}_{init}$	initial size of $\mathcal{B}$ 95
R_{init}	initial value of R 95
R_{max}	maximum number of available resources 96
R_{decay}	decayed resource level 96
s	density of patterns in a set 97

τ	rate of affinity contribution	100
ϕ_{init}	initial value of ϕ	100
$k_{compress}$	number of sub-nets	102
α	co-stimulation coefficient	102
β	network suppression coefficient	102
m_{min}	minimum degree for activation	103
age_i	age of i-*th* ALC	103
a_{min}	minimum age of ALC	103
a_{max}	maximum age of ALC	103
k_{clone}	maximum number of clones	103
$\mathcal{B}_{max}$	maximum size of $\mathcal{B}$	103
ς	mutation rate	104
ε_{death}	affinity death threshold	104
$\varepsilon_{network}$	network suppression threshold	105
φ	number (ratio) of ALCs in $\mathcal{B}$ with the lowest affinity	105
$\varepsilon_{fitness}$	ALC fitness threshold	105
l	number of levels in an idiotypic network of ALCs	116
z	number of network connections in an idiotypic network of ALCs	117
ν^j	concentration level of an ALC in level j of an idiotypic network	118
ϑ^i	stimulation level of an ALC in level i of an idiotypic network	118

Chapter 5 – A Local Network Neighbourhood Artificial Immune System with Application to Unsupervised Data Clustering

$\mathbf{b}_h$	ALC with the highest antigen affinity at position h in $\mathcal{B}$	127
$\mathbf{a}'$	antigen mutated clone	127
$\mathcal{C}_h$	set of antigen mutated clones of ALC $\mathbf{b}_h$	127
$\lvert\mathcal{C}_h\rvert$	clonal level of ALC $\mathbf{b}_h$	127
ε_{clone}	clonal level threshold	127
σ^*	normalised antigen affinity	127
θ	vector difference	128

$\mathcal{N}$	ALC network neighbourhood	129
ρ	window size of $\mathcal{N}$, i.e. neighbourhood size	129
μ	average network affinity between ALCs	129
i^*	index position in $\mathcal{B}$ with the highest average network affinity	131
ρ_r	ρ expressed as a ratio of $\mathcal{B}_{max}$	151

Chapter 7 – Data Clustering in Uncertain Environments using a Local Network Neighbourhood Artificial Immune System

$\bar{Q}_r$	mean clustering quality for run r	208
$\hat{Q}$	collective mean clustering quality	208
$Q_{best}(t)$	clustering quality at time t	208
$P^{(t)}$	data set P at time t	208
$K^{(t)}$	number of clusters in data set $P^{(t)}$ at time t	208
$\tilde{s}$	severity of change in P	208
$\tilde{f}$	frequency of change in P	209

Appendix B

Derived Publications

This appendix provides a list of articles which were derived from the work introduced in this thesis. These articles have been published or are currently being reviewed.

Book Chapters

A.J. Graaff and A.P. Engelbrecht. Chapter 18: Natural Immune System. *Computational Intelligence: An Introduction*, 2nd Edition, A.P. Engelbrecht (Author), John Wiley & Sons, October 2007.

A.J. Graaff and A.P. Engelbrecht. Chapter 19: Artificial Immune Models. *Computational Intelligence: An Introduction*, 2nd Edition, A.P. Engelbrecht (Author), John Wiley & Sons, October 2007.

Journal Publications

A.J. Graaff and A.P. Engelbrecht. Optimised Coverage of Non-self with Evolved Lymphocytes in an Artificial Immune System. *International Journal of Computational Intelligence Research*, vol. 2, no. 2, pp. 127–150, 2006.

A.J. Graaff and A.P. Engelbrecht. Clustering Data in an Uncertain Environment using an Artificial Immune System. *Pattern Recognition Letters*, vol. 32, no. 2, pp. 342–351, January 2011.

A.J. Graaff and A.P. Engelbrecht. Using sequential deviation to dynamically determine the number of clusters found by a local network neighbourhood artificial immune system. *Applied Soft Computing*, vol. 11, pp. 2698–2713, March 2011.

A.J. Graaff and A.P. Engelbrecht. Clustering Data in Stationary Environments with a Local Network Neighborhood Artificial Immune System. *International Journal of Machine Learning and Cybernetics*, submitted May 2011.

Conference Publications

A.J. Graaff and A.P. Engelbrecht. A local network neighbourhood artificial immune system for data clustering. In *IEEE Congress on Evolutionary Computation*, CEC 2007., pp. 260–267, 2007.

A.J. Graaff and A.P. Engelbrecht. Towards a self regulating local network neighbourhood artificial immune system for data clustering. In *IEEE Congress on Evolutionary Computation*, CEC 2008.(IEEE World Congress on Computational Intelligence), pp. 633–640, 2008.