DifFUZZY: A novel clustering algorithm for systems biology

Ornella Cominetti Allende

St John's College

University of Oxford

Supervised by Philip K. Maini,
Radek Erban, Helen Byrne and Philip Murray

A thesis submitted for the degree of

Doctor of Philosophy

Hilary Term 2012

A mi papá,

a mi mamá,

a Ádám

y

a la memoria de mis Nonnos

Acknowledgements

I would like to express my deepest gratitude to my supervisors Philip K. Maini, Radek Erban, Helen Byrne and Philip Murray for all their support and guidance throughout these past terms. I am indebted to them, for their help, encouragement and constant input which has enabled me to develop this work. I know how fortunate I am for having been mentored and endorsed by each one of them.

I also want to heartily thank many great collaborators in specific projects, who have contributed greatly to the completion of this research:

- To Anastasios Matzavinos (Department of Mathematics, Iowa State University), without whom DifFUZZY would not have been born and for having been a wonderful host in Iowa.
- To Sandhya Samarasinghe and Don Kulasiri (Centre for Advanced Computational Solutions C-fACS, Lincoln University, New Zealand) for sharing their expertise and for all their support and time in the beginnings of the clustering project, as well as for having been fantastic hosts and guides in New Zealand.
- To Professor Kwang-Hyun Cho and all the members of the Laboratory for Systems Biology and Bio-Inspired Engineering (KAIST, South Korea) for their hospitality and kindness.
- To Angela and George Shiflet (Wofford College, South Carolina), for their friendship and kindness, and for the excellent pair-programming sessions we shared.
- To Climent Casals-Pascual for teaching me much about malaria and clinical research, while keeping my motivation up during the last year of my DPhil. I am very grateful to him for allowing me to participate in such an interesting study and for his great and inspiring scientific discussions.
- To Nick Jones for being a great transfer and confirmation examiner and for bringing me into the malaria project, where his input was highly appreciated.
- To David Smith, whose fantastic visualisation algorithm was key in a clear presentation of the results.

I would like to thank the Oxford University Press and the University of Oxford for financial support through the Clarendon Fund, the Systems Biology Doctoral Training Centre and the Centre for Mathematical Biology, which ultimately made this work possible. A number of other funding sources have allowed me to complete my work and to disseminate my research through a number of conferences and academic visits, and I am deeply honored and grateful for their generosity. These are: the International Society for Computational Biology, the Society for Mathematical Biology, the European Society for Mathematical and Theoretical Biology, the Marie Curie Network, the Google Anita Borg Memorial Scholarship, the Sidney Perry Foundation, the London Mathematical Society, the Mathematical

Biosciences Institute (The Ohio State University), the Mathematical Institute (Oxford University), St John's College (Oxford University), the Centre for Mathematical Modeling (University of Chile), Beca Presidente de la República (Gobierno de Chile), the President's Fund Award of the Federation of University Women, The Carlton House Charitable Trust and the Oxford Centre for Integrative Systems Biology.

A thousand thanks go to the Chaste team, in particular to Alex and Ozzy, for all their help and patience.

I also want to extend my gratitude to many friends at CMB, DTC and St John's, who made my years at Oxford a wonderful time to remember. I will miss them all.

Finally many thanks to my loving, multitudinous and supportive family: Fio, Rossana, Sandy, Randy, nephews and nieces, cousins, aunts and uncles.

To my parents, Paolo and Myriam, for their love, support and example, to whom all my achievements belong.

To Ádám, for absolutely everything, especially for his constant encouragement, companionship and unconditional love.

DifFUZZY: A novel clustering algorithm for systems biology

Ornella Cominetti Allende
St John's College – University of Oxford
Hilary, 2012

Abstract of the dissertation submitted for the Degree of Doctor of Philosophy

Abstract

Current studies of the highly complex pathobiology and molecular signatures of human disease require the analysis of large sets of high-throughput data, from clinical to genetic expression experiments, containing a wide range of information types. A number of computational techniques are used to analyse such high-dimensional bioinformatics data.

In this thesis we focus on the development of a novel soft clustering technique, DifFUZZY, a fuzzy clustering algorithm applicable to a larger class of problems than other soft clustering approaches. This method is better at handling datasets that contain clusters that are curved, elongated or are of different dispersion. We show how DifFUZZY outperforms a number of frequently used clustering algorithms using a number of examples of synthetic and real datasets. Furthermore, a quality measure based on the diffusion distance developed for DifFUZZY is presented, which is employed to automate the choice of its main parameter.

We later apply DifFUZZY and other techniques to data from a clinical study of children from The Gambia with different types of severe malaria. The first step was to identify the most informative features in the dataset which allowed us to separate the different groups of patients. This led to us reproducing the World Health Organisation classification for severe malaria syndromes and obtaining a reduced dataset for further analysis. In order to validate these features as relevant for malaria across the continent and not only in The Gambia, we used a larger dataset for children from different sites in Sub-Saharan Africa. With the use of a novel network visualisation algorithm, we identified pathobiological clusters from which we made and subsequently verified clinical hypotheses.

We finish by presenting conclusions and future directions, including image segmentation and clustering time-series data. We also suggest how we could bridge data modelling with bioinformatics by embedding microarray data into cell models. Towards this end we take as a case study a multiscale model of the intestinal crypt using a cell-vertex model.

Table of Contents

Chapter 1 Motivation and Introduction **1**

1.1 Clustering in Systems Biology 1

1.1.1 High-throughput data 2

1.1.2 Methods for systems biology 3

1.1.3 Cluster analysis: an overview 4

1.1.4 Network clustering 6

1.1.4.1 Network visualisation 7

1.2 Outline of this thesis 8

Chapter 2 Clustering Algorithms and Validation **10**

2.1 Clustering Algorithms 10

2.1.1 Hard clustering algorithms 11

2.1.1.1 K-means clustering 11

2.1.1.2 Hierarchical clustering 14

2.1.1.3 Spectral clustering 21

2.1.1.4 Self-organising maps 27

2.1.2 Fuzzy clustering algorithms 32

2.1.2.1 Fuzzy C-means clustering 32

2.1.2.2 Possibilistic C-means clustering 37

2.1.3 Summary of clustering methods 40

2.2 Cluster validation 42

2.2.1 Similarity 42

2.2.2 Hard clustering 44

2.2.2.1 Silhouette width 44

2.2.2.2 Rand index (RI) 46

2.2.3 Fuzzy clustering 46

2.2.3.1 Fuzzy silhouette width 46

2.2.4 Clustering evaluation when true clusters are known 47

2.2.4.1 Classification error 48

2.2.4.2 ROC curves 49
2.3 Discussion 51

Chapter 3 DifFUZZY: A Novel Fuzzy Clustering Algorithm **52**
3.1 The algorithm 52
3.1.1 Identification of the core of clusters 53
3.1.2 Computation of auxiliary matrices **W**, **D** and **P** 55
3.1.3 The membership values of soft data points 58
3.1.4 Geometric and graph interpretation of DifFUZZY 58
3.2 DifFUZZY's performance over test datasets 60
3.2.1 Exploring the effect of DifFUZZY's internal parameters 63
3.3 Exploring the $F(\sigma)$ function 65
3.4 About the identification of the cluster cores 69
3.5 Automatic selection of parameter M 73
3.5.1 DifFUZZY silhouette width 73
3.6 Discussion 77

Chapter 4 DifFUZZY: Testing and Validation **79**
4.1 Benchmark datasets 79
4.1.1 Description of the benchmark datasets 79
4.1.1.1 *CRC* dataset 80
4.1.1.2 *Iris* dataset 81
4.1.1.3 *Leukemia* dataset 83
4.1.1.4 *Ruspini* dataset 84
4.1.1.5 *WDBC* dataset 85
4.1.1.6 *Wine* dataset 86
4.1.1.7 *Zoo* dataset 88
4.1.2 Summary of results over benchmark datasets 91
4.2 Synthetic test datasets 93
4.2.1 *Globular clusters* dataset 93
4.2.2 *Concentric rings* dataset 94
4.2.3 *Elongated clusters* dataset 95
4.2.4 *3-D globular clusters* dataset 96
4.2.5 *Half moons* dataset 97
4.2.6 *Interlaced rings* dataset 98
4.2.7 *Stochastic dynamical system* dataset 100
4.3 Discussion 102

Chapter 5 Clustering Clinical Data: Severe Malaria **105**
5.1 Background . . . 105
5.1.1 About severe malaria . . . 105
5.1.1.1 Clinical features of severe malaria in children . . . 106
5.1.2 About the data . . . 107
5.2 Methodology . . . 108
5.2.1 Pre-processing . . . 110
5.2.1.1 Mixed data . . . 110
5.2.1.2 Data cleaning . . . 111
5.2.1.3 Missing data imputation . . . 112
5.2.1.4 Normalisation . . . 114
5.2.2 Feature selection . . . 114
5.2.3 Constructing the distance matrix . . . 117
5.2.4 Thresholding . . . 118
5.2.5 Network visualisation . . . 121
5.3 Results . . . 122
5.3.1 Network cluster validation . . . 126
5.3.2 Network validation . . . 128
5.4 Discussion . . . 132

Chapter 6 Conclusions and Future Directions **133**

Conclusions and Future Directions **133**
6.1 Conclusions . . . 133
6.2 Future Directions . . . 135
6.2.1 Clustering in cellular modelling . . . 135
6.2.1.1 Biological motivation . . . 137
6.2.1.2 Sub-cellular model . . . 138
6.2.2 Clustering in medical imaging . . . 139
6.2.3 Clustering time series . . . 141
6.2.4 Discussion . . . 142

Appendices **143**

Appendix A Clustering Algorithms and Validation **144**
A.1 Fuzzy C-means clustering . . . 144
A.1.1 Minimisation of the objective function J . . . 144

Appendix B DifFUZZY **146**

B.1 Random walk and diffusion . . . 146
B.2 Schematic representation of DifFUZZY . . . 149
B.3 Exploring the effect of DifFUZZY's internal parameters . . . 150
B.4 Implementation of DifFUZZY . . . 151
B.5 DifFUZZY: testing and validation . . . 153
B.5.1 Benchmark datasets . . . 153
B.5.2 ROC curves . . . 154
B.5.3 Random clustering . . . 154

Appendix C Clustering Clinical Data: Severe Malaria **155**
C.1 The Gambia dataset . . . 155
C.2 Feature selection . . . 156
C.2.1 DifFUZZY MV plots . . . 156
C.2.1.1 CM and RD . . . 156
C.2.1.2 CM, CMRD and RD . . . 157
C.3 Thresholding . . . 158
C.4 Network validation . . . 159
C.4.1 SMAC dataset . . . 159
C.4.1.1 SMAC site-specific networks . . . 159

Appendix D Conclusions and Future Directions **160**
D.1 Cellular modelling . . . 160
D.1.1 Cell-vertex model . . . 160
D.1.1.1 Implementation of the cell-vertex model . . . 161
D.1.1.2 T1 vertices swaps . . . 162
D.1.1.3 Cell sorting . . . 163
D.1.2 The Crypt Model . . . 167
D.1.2.1 Crypt simulations . . . 169
D.1.2.2 Modelling migration in the crypt . . . 171
D.1.3 Chaste . . . 172
D.1.4 Equations of motion for vertices in the cell-vertex model . . . 173

References **177**

List of Figures

Chapter 1 Motivation and Introduction **1**

Figure 1.1 Schematic of clustering 5
Figure 1.2 Schematic outline of this thesis. 9

Chapter 2 Clustering Algorithms and Validation **10**

Figure 2.1 Test datasets 11
Figure 2.2 K-means clustering on the *Globular clusters* dataset 12
Figure 2.3 K-means clustering on the *Concentric rings* dataset 13
Figure 2.4 K-means clustering on the *Elongated clusters* dataset 13
Figure 2.5 Hierarchical clustering on the *Globular clusters* dataset 16
Figure 2.6 Hierarchical clustering on the *Globular clusters* dataset 16
Figure 2.7 Hierarchical clustering on the *Globular clusters* dataset 16
Figure 2.8 Hierarchical clustering on the *Concentric rings* dataset 18
Figure 2.9 Hierarchical clustering on the *Concentric rings* dataset 18
Figure 2.10 Hierarchical clustering on the *Concentric rings* dataset 18
Figure 2.11 Hierarchical clustering on the *Elongated clusters* dataset 19
Figure 2.12 Hierarchical clustering on the *Elongated clusters* dataset 19
Figure 2.13 Hierarchical clustering on the *Elongated clusters* dataset 19
Figure 2.14 Spectral clustering on the *Globular clusters* dataset 25
Figure 2.15 Spectral clustering on the *Concentric rings* dataset 25
Figure 2.16 Spectral clustering on the *Elongated clusters* dataset 26
Figure 2.17 Examples of SOM neighbourhoods 28
Figure 2.18 Training of a SOM 29
Figure 2.19 SOM clustering on the *Globular clusters* dataset 30
Figure 2.20 SOM clustering on the *Concentric rings* dataset 30
Figure 2.21 SOM clustering on the *Elongated clusters* dataset 30
Figure 2.22 FCM on the *Globular clusters* dataset 34
Figure 2.23 FCM on the *Concentric rings* dataset 35

Figure 2.24 FCM on the *Elongated clusters* dataset 36
Figure 2.25 PCM on the *Globular clusters* dataset 37
Figure 2.26 PCM on the *Concentric rings* dataset 38
Figure 2.27 PCM on the *Elongated clusters* dataset 39
Figure 2.28 Silhouette widths of K-means on the test datasets 45

Chapter 3 DifFUZZY: A Novel Fuzzy Clustering Algorithm 52
Figure 3.1 First step of DifFUZZY 54
Figure 3.2 Plot of $F(\sigma)$ 54
Figure 3.3 *Globular clusters* dataset 57
Figure 3.4 DifFUZZY on the *Globular clusters* dataset 60
Figure 3.5 DifFUZZY on the *Concentric rings* dataset 61
Figure 3.6 DifFUZZY on the *Elongated clusters* dataset 62
Figure 3.7 DifFUZZY on the *Elongated clusters* dataset 64
Figure 3.8 DifFUZZY on the *Elongated clusters* dataset 64
Figure 3.9 DifFUZZY on the *Elongated clusters* dataset 64
Figure 3.10 *'3-5 clusters'* dataset 66
Figure 3.11 σ^* versus M for the *'3-5 clusters'* dataset 66
Figure 3.12 DifFUZZY cluster cores of the *'3-5 clusters'* dataset 67
Figure 3.13 *Iris* dataset 69
Figure 3.14 DifFUZZY and FCM on the *Iris* dataset 70
Figure 3.15 DifFUZZY on the *Iris* dataset 70
Figure 3.16 *Iris* dataset cluster cores 71
Figure 3.17 Scaled *Iris* dataset cluster cores 71
Figure 3.18 Scaled *Iris* dataset 72
Figure 3.19 '*X*' dataset 73
Figure 3.20 FCM on the *Concentric rings* dataset 75
Figure 3.21 Automatic selection of parameter M 76

Chapter 4 DifFUZZY: Testing and Validation 79
Figure 4.1 *Colorectal cancer* dataset 80
Figure 4.2 ROC curves for the *Colorectal cancer* dataset 81
Figure 4.3 DifFUZZY and FCM on the *Iris* dataset 81
Figure 4.4 ROC curves of DifFUZZY and FCM on test datasets 82
Figure 4.5 DifFUZZY and FCM on the *Leukemia* dataset 83
Figure 4.6 ROC curves of DifFUZZY and FCM on the *Leukemia* dataset 84
Figure 4.7 *Ruspini* dataset 84

Figure 4.8 DifFUZZY and FCM on the *Ruspini* dataset . . . 85
Figure 4.9 *WDBC* dataset . . . 86
Figure 4.10 *Wine* dataset . . . 87
Figure 4.11 DifFUZZY and FCM on the *Zoo* dataset . . . 89
Figure 4.12 DifFUZZY clusters from the *Zoo* dataset . . . 90
Figure 4.13 DifFUZZY and FCM on the *Globular clusters* dataset . . . 93
Figure 4.14 DifFUZZY and FCM on the *Concentric rings* dataset . . . 94
Figure 4.15 DifFUZZY and FCM on the *Elongated clusters* dataset . . . 95
Figure 4.16 DifFUZZY and FCM on the *3-D globular clusters* dataset . . . 96
Figure 4.17 DifFUZZY and FCM on the *Half moons* dataset . . . 97
Figure 4.18 ROC curves for the *Half moons* dataset . . . 98
Figure 4.19 DifFUZZY and FCM on the *Interlaced rings* dataset . . . 99
Figure 4.20 Stochastic dynamical system realisation . . . 100
Figure 4.21 DifFUZZY and FCM on the *Stoch. dynamical sys.* dataset . . . 101

Chapter 5 Clustering Clinical Data: Severe Malaria **105**
Figure 5.1 Severe malaria groups . . . 107
Figure 5.2 Flowchart of malaria data analysis . . . 109
Figure 5.3 Discriminant variables . . . 116
Figure 5.4 Clustering coefficient versus threshold . . . 119
Figure 5.5 Sparsity pattern plots . . . 120
Figure 5.6 Global network for The Gambia dataset . . . 123
Figure 5.7 RD clusters from The Gambia network . . . 124
Figure 5.8 Correspondence of clusters from global network . . . 126
Figure 5.9 Map of SMAC network countries . . . 128
Figure 5.10 Banjul (Gambia) and Blantyre (Malawi) networks . . . 129
Figure 5.11 Kumasi (Ghana) and Kilifi (Kenya) networks . . . 130
Figure 5.12 SMAC network . . . 131

Conclusions and Future Directions **133**
Figure 6.1 Colorectal cancer as a multiscale disease . . . 136
Figure 6.2 DifFUZZY on an image segmentation task . . . 140

Appendices **144**

Appendix A Clustering Algorithms and Validation **144**

Appendix B DifFUZZY **146**

Figure B.3 Solution of the 2-D diffusion equation 148
Figure B.4 Solution of the 2-D diffusion equation on a U lattice 148
Figure B.5 Schematic diagram of DifFUZZY 149
Figure B.6 DifFUZZY on the *Elongated clusters* dataset 150
Figure B.7 ROC curves of DifFUZZY and FCM on the *Wine* dataset 154
Figure B.8 Random clustering on test datasets 154

Appendix C Clustering Clinical Data: Severe Malaria **155**
Figure C.9 DifFUZZY MV plots on selected variables of The Gambia dataset 156
Figure C.10 DifFUZZY MV plots on selected variables of The Gambia dataset 157
Figure C.11 Size of largest cluster versus threshold 158
Figure C.12 Clustering coefficient versus threshold for the SMAC dataset 158
Figure C.13 Lambaréné (Gabon) and Libreville (Gabon) networks 159

Appendix D Conclusions and Future Directions **160**
Figure D.14 Schematic representation of a cell 160
Figure D.15 T1 swaps on the cell-vertex model 162
Figure D.16 Simulation of preferential adhesion on a monolayer of cells 164
Figure D.17 Simulation of repulsion on a monolayer of cells 164
Figure D.18 Simulation of adhesion on a monolayer of cells with cell birth 165
Figure D.19 Simulation of repulsion on a monolayer of cells with cell birth 166
Figure D.20 Cell types transitions 168
Figure D.21 Crypt cell-vertex simulation with preferential adhesion 170
Figure D.22 Crypt cell-vertex model simulation 171
Figure D.23 Schematic representation of a vertex 174

List of Tables

Chapter 2 Clustering Algorithms and Validation **10**
2.1 Summary of clustering methods 40
2.2 Confusion matrix 49

Chapter 4 DifFUZZY: Testing and Validation **79**
4.1 *Zoo* dataset (extract) 88
4.2 Summary comparison clustering methods over benchmark datasets 92

Chapter 5 Clustering Clinical Data: Severe Malaria **105**
5.1 Syndromes distribution on The Gambia dataset 107
5.2 RD clusters 125
5.3 Confusion Matrix of RD clusters in HC and Original network 127
5.4 SMAC RD clusters 130

Appendices **144**

Appendix B DifFUZZY **146**
B.1 Time complexity of DifFUZZY 151
B.2 *Ruspini* dataset 153

Appendix C Clustering Clinical Data: Severe Malaria **155**
C.3 The Gambia dataset variables summary (extract) 155
C.4 SMAC dataset number of cases of severe malaria syndromes 159

Appendix D Conclusions and Future Directions **160**
D.5 Crypt cell-vertex model parameters 169
D.6 Notation of equations of motion for vertices in the cell-vertex model 176

List of Acronyms

ALL	Acute Lymphoblastic Leukemia	82
AML	Acute Myeloid Leukemia	82
AUC	Area Under the Curve	50
BMU	Best Matching Unit	27
BPCA	Bayesian PCA	113
CD	Case Deletion	112
CM	Cerebral Malaria	106
CRC	Colorectal Cancer	79
D	Differentiated	167
DM	Distance Matrix	117
DSW	DifFUZZY Silhouette Width	74
EM	Expectation Maximisation	113
FCM	Fuzzy C-Means	31
FPR	False Positive Rate	49
FSW	Fuzzy Silhouette Width	46
GMM	Gaussian Mixture Models	41
HAART	Highly Active Anti Retroviral Therapy	31
HC	Hierarchical Clustering	14
HCT	Hard Clusters by Threshold	34
HIV	Human Immunodeficiency Virus	31
kNN	k-Nearest Neighbours	21
MAR	Missing At Random	112
MC	Matrix of Costs	48
MI	Mutual Information	116
MV	Missing Value	34

PCA	Principal Component Analysis	2
PCM	Possibilistic fuzzy C-Means	37
RAVG	Row Average	112
RD	Respiratory Distress	106
RI	Rand Index	45
ROC	Receiver Operating Characteristic	49
S	Stem	167
SM	Severe Malaria	105
SMA	Severe Malarial Anemia	106
SMAC	Severe Malaria in African Children	128
SOM	Self Organising Map	26
SSA	Stochastic Simulation Algorithm	99
SW	Silhouette Width	44
TA	Transit Amplifying	167
TPR	True Positive Rate	49
UCI	University of California, Irvine	177
WDBC	Wisconsin Diagnostic Breast Cancer	85
WHO	World Health Organization	106

Variables and Parameters

a	Number of pairs of points clustered together in partitionings $\mathbf{C}^{(1)}$ and $\mathbf{C}^{(2)}$
a_i	Average similarity of data point $\mathbf{x}_i$
α	DifFUZZY internal parameter
$\alpha(t)$	SOM learning function
$\mathbf{A}$	Dataset
b	Number of pairs of points co-clustered either in partitioning $\mathbf{C}^{(1)}$ or $\mathbf{C}^{(2)}$
b_i	Minimum similarity between the data point $\mathbf{x}_i$ and all other data points points that do not belong to $\mathbf{x}_i$'s cluster
β	DifFUZZY internal parameter
β^*	DifFUZZY optimal β value
$\overline{\beta}$	Number of β values
$\mathbf{C}_k$	Cluster number k
$\lvert\mathbf{C}_k\rvert$	Number of elements in k^{th} cluster
$\mathbf{c}_k$	Centroid of k^{th} cluster
$\mathbf{C}$	Partitioning
$\widetilde{\mathbf{C}}$	True partitioning
d	Number of pairs of points not clustered together in partitionings $\mathbf{C}^{(1)}$ or $\mathbf{C}^{(2)}$
D	Diffusion coefficient
$\mathbf{D}$	Diagonal matrix with $D_{i,j}$ entries
d	Dimensionality of data points
dt	Random walk time step duration
$\delta_1, \delta_2, \ldots \delta_4$	Correlation similarity measures

$\mathrm{dist}(\mathbf{x}_s, k)$	Diffusion distance between soft data point $\mathbf{x}_s$ and k^{th} cluster
$\mathbf{e}$	Indicator vector
$e_{\mathbf{x}_i}$	Diffusion distance between $\mathbf{x}_i$ and its second closest cluster
η	Manifold dimensionality
$f(x,y,t)$	Density of diffusing particles at (x,y) at time t
$F(\sigma)$	Number of σ-neighbourhood graph components of order larger than M
$\gamma_1, \gamma_2, \gamma_3$	DifFUZZY internal parameters
$h(k)$	Index of the most similar cluster in the true partitioning to the cluster C_k
$\mathbf{h}_i$	Spectral clustering auxiliary vectors
I	Identity matrix
J	Overall objective function
J_i	Objective function for the i^{th} data point
$\mathbf{J}$	Adhesion-free energy matrix
K	Number of clusters
$k_1, k_2, \ldots k_4$	Stochastic dynamical system rate constants
κ_i	i^{th} Lagrangian multiplier
$\mathbf{L}$	Laplacian matrix
$L(\beta)$	DifFUZZY auxiliary function
L_1	Manhattan distance
Λ	Overall objective function with Lagrangian multipliers
Λ_i	Objective function with Lagrangian multipliers for the i^{th} data point
λ_i	i^{th} eigenvalue
M	DifFUZZY external parameter
N	Total number of data points
N_{und}	Total number of soft data points
n	Total number of eigenvalues
n_k	Number of points in the k^{th} cluster core
p	Random walk jump probability

$\mathbf{P}$	Transition matrix with $P_{i,j}$ entries
$\overline{\mathbf{P}}$	Auxiliary transition matrix with $\overline{P_{i,j}}$ entries
$\boldsymbol{\phi}_0$	Stationary distribution
Q	Number of selected genes
q	Fuzziness parameter
$\mathbf{S}$	Similarity matrix with $S_{i,j}$ entries
$s_{\mathbf{x}_i}$	Diffusion distance between the data point $\mathbf{x}_i$ and its closest cluster
σ	Neighbourhood radious
$\sigma*$	Smallest σ value which maximises $F(\sigma)$
$\overline{\sigma}$	Number of σ values
σ_t	Neighbourhood radious at time t
$\overline{\text{SW}}$	Average silhouette width
t	Time
$t_{\max}$	SOM number of time steps
θ	SOM Gaussian neighbourhood function
$u_{i,a}$	Highest membership value of data point $\mathbf{x}_i$ in all clusters
$u_{i,b}$	Second highest membership value of data point $\mathbf{x}_i$ in all clusters
$u_{i,k}$	Membership value of data point $\mathbf{x}_i$ in k^{th} cluster
$\mathbf{v}_1$	Smallest eigenvector
$\mathbf{v}_n$	n^{th} smallest eigenvector
$\mathbf{W}$	Weights matrix with $W_{i,j}$ entries
$\overline{\mathbf{W}}$	Auxiliary weights matrix with $\overline{W}_{i,j}$ entries
$\mathbf{x}_i$	Data point number i
x_{il}	l^{th} component of data point number i
$\mathbf{x}_{k(s)}$	Hard point in the k^{th} core which is closest to $\mathbf{x}_s$
$\mathbf{x}_s$	Soft data point
y_i	Auxiliary vectors of spectral clustering
z	Threshold

Chapter 1

Motivation and Introduction

In this chapter we will introduce and motivate the role played by cluster analysis in systems biology, and in particular how it can be used to study high-throughput data. We begin by introducing the problem along with the key concepts that are needed as background for this thesis.

1.1 Clustering in Systems Biology

Over the past two decades there has been rapid and dramatic development in the fields of bioinformatics, systems biology and biotechnology. This growth has led to an exponential accumulation of biological data that requires intense processing and detailed analysis [Howe et al., 2008]. Handling all these complex and large datasets now drives the development of another area of science: data mining [Berthold and Hand, 1999]. Building a link between the fields of bioinformatics and data mining is now imperative.

The main focus of this thesis is the development of a novel clustering technique published as: "DifFUZZY: A fuzzy clustering algorithm for complex data sets" (Int. J. Comput. Intelligence in Bioinformatics and Systems Biology, Vol. 1, No. 4, 402-417, 2010) and freely available for download from `http://www.maths.ox.ac.uk/cmb/diffuzzy`, and its applications to the fields of exploratory data mining and data analysis. Conceptually, cluster analysis is a simple exercise that consists of recognising sets or clusters of elements that are somewhat more similar to each other than to the members of other distinct clusters [Bezdek et al., 1983]. However, difficulties arise when analysing datasets with complex

structure and when the underlying number of clusters and structure is unknown and often overlapping, which is generally the case with biological datasets. We will aim to explore and tackle clustering issues related to biological assays such as microarray data.

1.1.1 High-throughput data

Via the simultaneous measurement of large numbers of biological components, current *omics* technologies, such as genomics, proteomics and metabolomics, produce scores of measurements per sample. These output measurements are known as *high-throughput data* and one of the major challenges for systems biology is to analyse and integrate these diverse types of overwhelmingly large amounts of data [Baggerly and Coombes, 2006; Shah et al., 2007].

Furthermore, data are produced at a range of different scales, from genomic to proteomic to population level. Without automatic identification and grouping of these different scales, data analysis becomes unfeasible. When dealing with such large, diverse datasets, the use of good visualisation methods is essential. There are a number of techniques to analyse large and/or high-dimensional datasets, and clustering and network visualisation play a very important role [Samarasinghe, 2006].

Microarrays are the most commonly used apparatus for high-throughput experiment in the biological sciences [Wang et al., 2008]. They rely on the hybridisation of labelled probes of nucleic acids to specific targets (complimentary nucleic acids). They allow us to understand genetic information by measuring the expression levels of thousands of genes in a single sample [Brown and Botstein, 1999; Baggerly and Coombes, 2006].

Other examples of high-throughput data are: large clinical datasets with different types of measurements and variables, large biological networks, among many others.

1.1.2 Methods for systems biology

Data exploration, dimensionality reduction and feature extraction are important first steps in the analysis of high-throughput data. A problem arises when the data are multidimensional, complex, inter-dependent and noisy, such as genetic expression data, because standard techniques such as principal component analysis (PCA) [Jolliffe, 2002] and K-means clustering [MacQueen, 1967], are inadequate. Furthermore, hard clustering methods such as K-means may be inadequate since clusters in genetic expression data are often fuzzy, given that genes can be involved in more than one cellular process and interact with a number of different genes, as well as being coordinated by different regulatory mechanisms [Dembélé and Kastner, 2003].

However, visualisation can provide qualitative clues on key issues such as the importance of certain variables and their interactions, or it can help identify non-linearities and trends in the data. Among common visualisation methods are scatter plots, histograms, correlation graphs, parallel visualisation and projections to reduce dimensions, such as PCA [Samarasinghe, 2006].

Analysis of large amounts of data obtained through these approaches requires techniques that can uncover associations in high dimensional datasets. One such tool, and a key methodology, is cluster analysis. Besides a list of commonly used techniques there are many more methods (self-organising maps (SOMs), Boolean networks and PCA [Kohane et al., 2003]) that are used for studying large and complex datasets.

Clustering analysis of microarray data can give us information on which genes are co-expressed, or show similar expression profiles under similar conditions. It can also help us reduce the complexity of the dataset being studied, by identifying outliers[1] or large groups of data within the set, which should later be analysed independently. By identifying genes that are co-expressed, we are one step closer to obtaining a preliminary map of interac-

[1] Outliers can be defined as data points that have a very small probability of belonging to the dataset [van Hulle and Davis, 1998].

tions that are present within genes under similar conditions, such as being associated with a specific cancer-type.

1.1.3 Cluster analysis: an overview

The need to interpret and extract useful information from high-dimensional datasets has led over the past decades to the development of dimensionality reduction and data clustering techniques. Clustering is the partitioning of a dataset into clusters or subsets of objects that are similar in some way, that share common characteristics or that are close together according to some distance measure [Kaufman and Rousseeuw, 1990; Martínez and Martínez, 2004].

In Fig. 1.1 we use pears and apples to depict clustering. This classification of objects into different clusters can be partitional (most commonly known as hard clustering), hierarchical or overlapping [Martínez and Martínez, 2004]. Partitional clustering (Fig. 1.1 (c)) corresponds to clustering where the clusters are determined at the same time and where each object can be assigned to one and only one cluster, whereas in hierarchical clustering (Fig. 1.1 (b)), the clusters are found successively using clusters that were previously identified. Soft or fuzzy clustering (Fig. 1.1 (d)) includes clustering techniques that allow elements to be assigned to more than one cluster [Kaufman and Rousseeuw, 1990; Martínez and Martínez, 2004].

The goal of clustering is "to determine the intrinsic clustering in a dataset of unlabelled objects" [Trujillo et al., 2005], and is "one of the most widely used techniques for exploratory data analysis, with applications ranging from statistics, computer science, biology to social sciences or psychology" [Luxburg, 2007]. For instance, clustering can be used to detect genes that have similar expression profiles, such as being over- or under-expressed in certain experimental conditions, tissues or patients, or to identify families of proteins using amino acid sequences.

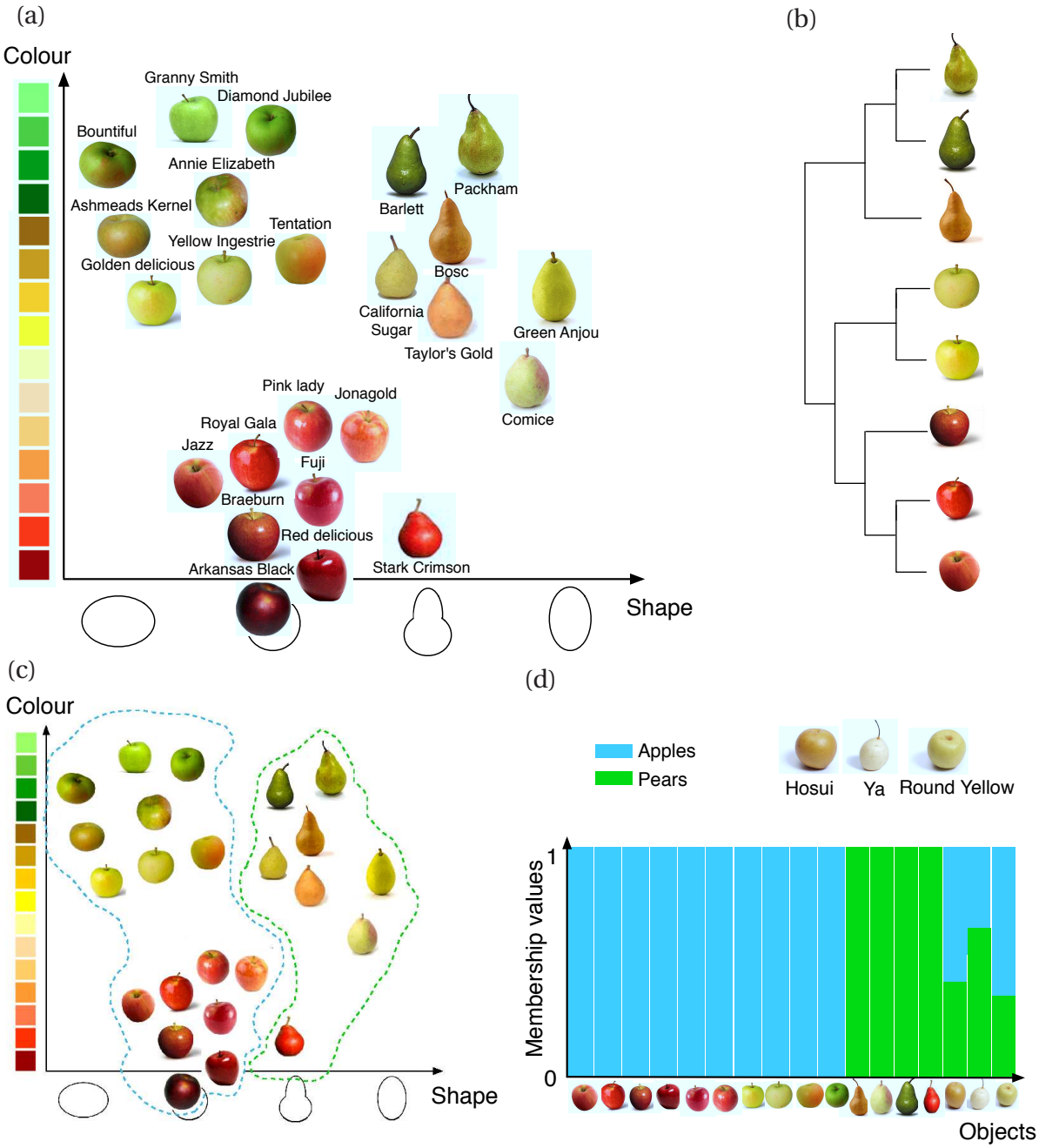


Figure 1.1: Schematic of clustering using two pome fruits: pears and apples. (a) A dataset of pears and apples represented by features of colour and shape. (b) Hierarchical clustering of a subset of the original pears and apples. Depending on where the tree is cut, apples are separated from pears (cutting over the first, most central branching) and also green/yellow apples from red apples (further cutting the apple cluster). (c) Crisp representation of clustering over mutually exclusive clusters of apples and pears, and (d) Fuzzy clustering schematic representation. Three additional pome fruits are added to the dataset, Asian pear varieties Ya, Hosui and Round Yellow, all three are hybrids between apples and pears. Fruit objects which belong 100% to the apples (cyan cluster), will have a membership value of 1 in that cluster, and zero in the other, whereas objects with fuzzy membership will have an intermediate membership degree in both clusters, such that their membership values in all clusters sum to 1. In this example, a crisp partition would allocate the Hosui and Round Yellow apple-pear types to the apples and the Ya would be assigned to the pears.

1.1.4 Network clustering

Network clustering is the identification of clusters or communities in a graph, where a graph is a set of objects or nodes with a set of pairwise connections, also known as edges. A network can be defined as a graph with additional information on the nodes or the edges [de Nooy et al., 2005], but in this thesis we will use both terms interchangeably.

Networks are said to be weighted or unweighted, directed or undirected if their edges have weighs or directions, respectively. Fully connected networks are those for which every pair of nodes is connected by an edge [Newman, 2010].

Clusters in networks are also known as communities or modules and represent clusters of nodes which are more densely connected to each other than to nodes in other clusters [Palla et al., 2005].

A quantitative measure on a network that reflects how well connected are its nodes is the clustering coefficient C_i;

$$C_i = \frac{\rho_i}{\binom{r_i}{2}}, \tag{1.1}$$

where r_i is the number of neighbours of node i, and ρ_i is the number of neighbours of node i that are connected through an edge.

The average of the clustering coefficient over all nodes on a network is used as a quality measure of the compactness of the clusters in the network, and will be used in Chapter 5, Section 5.2.4. This term should not be confused with data clustering in general, as it is a specific quality measure for networks.

There is a vast number of algorithms and heuristic approaches specialised to detect communities in networks, many of which make use of weights, directions and other intrinsic properties of networks [Newman, 2006; Leskovec et al., 2010]. A large proportion of these techniques are topology driven, have poor scalability or make statistical assumptions that are not always justified, such as the stochastic equivalence of all nodes. Other methods are

based on statistical models of networks [Zhao et al., 2011], and assess the quality of the partitioning by calculating how much the network differs from a graph generated by a random process [Good et al., 2010]. A partitioning over a network is thus based on the structure or topology of the network rather than on the attributes of the individual data points [Pujol et al., 2006].

While it is not within the scope of this thesis to study the community structure of networks *per se*, we will use networks as a tool in intermediate steps of our clustering algorithm (see DifFUZZY in Section 3.1.1 and clustering of clinical malaria data in Section 5.2).

1.1.4.1 Network visualisation

Given that systems biology datasets are large, ranging from hundreds of protein complexes to thousands of gene products to tens of thousands of interactions [Gehlenborg et al., 2010], we are concerned with large scale network visualisation, which involves drawing thousands of nodes and vertices. Such data is best visualised using force directed algorithms, also known as spring embedders [Di Battista et al., 1994, 1999]. These algorithms represent edges as springs that exert mechanical forces on the nodes and aim to minimise the total energy of the system. Additionally, electrical or gravitational fields, as well as magnetic forces can be incorporated to further constrain the layout in order to improve its resolution [Di Battista et al., 1994, 1999].

For very large datasets, clustering techniques may be applied to network visualisation. This allows a hierarchical representation of the data with super-nodes and nodes, where super-nodes are clusters of data points. Such condensation of information allows an easier understanding of the general structure of the data [Huang and Eades, 1998].

1.2 Outline of this thesis

This thesis is organised in the following manner: we begin in Chapter 2 by giving an overview of the most commonly used clustering algorithms, illustrating their application to three selected test datasets and describing some commonly used cluster validation indices; in Chapter 3, we present and study the performance of a new fuzzy clustering algorithm, DifFUZZY. We describe the three main steps that make up the method, and then give an intuitive explanation of how DifFUZZY works; then in Chapter 4 we compare how DifFUZZY performs on a number of test datasets, both synthetic and real biological sets versus the standard soft clustering algorithm Fuzzy C-means (FCM); in Chapter 5 we present a clustering analysis and network visualisation carried out over a dataset of children with different severe malaria syndromes; finally in Conclusions and Future Directions we present general comments and conclusions, we examine potential applications of DifFUZZY in multiscale modelling, medical image segmentation, time series clustering and we introduce preliminary work done along these lines.

A schematic representation of the structure of this thesis is shown in Fig. 1.2.

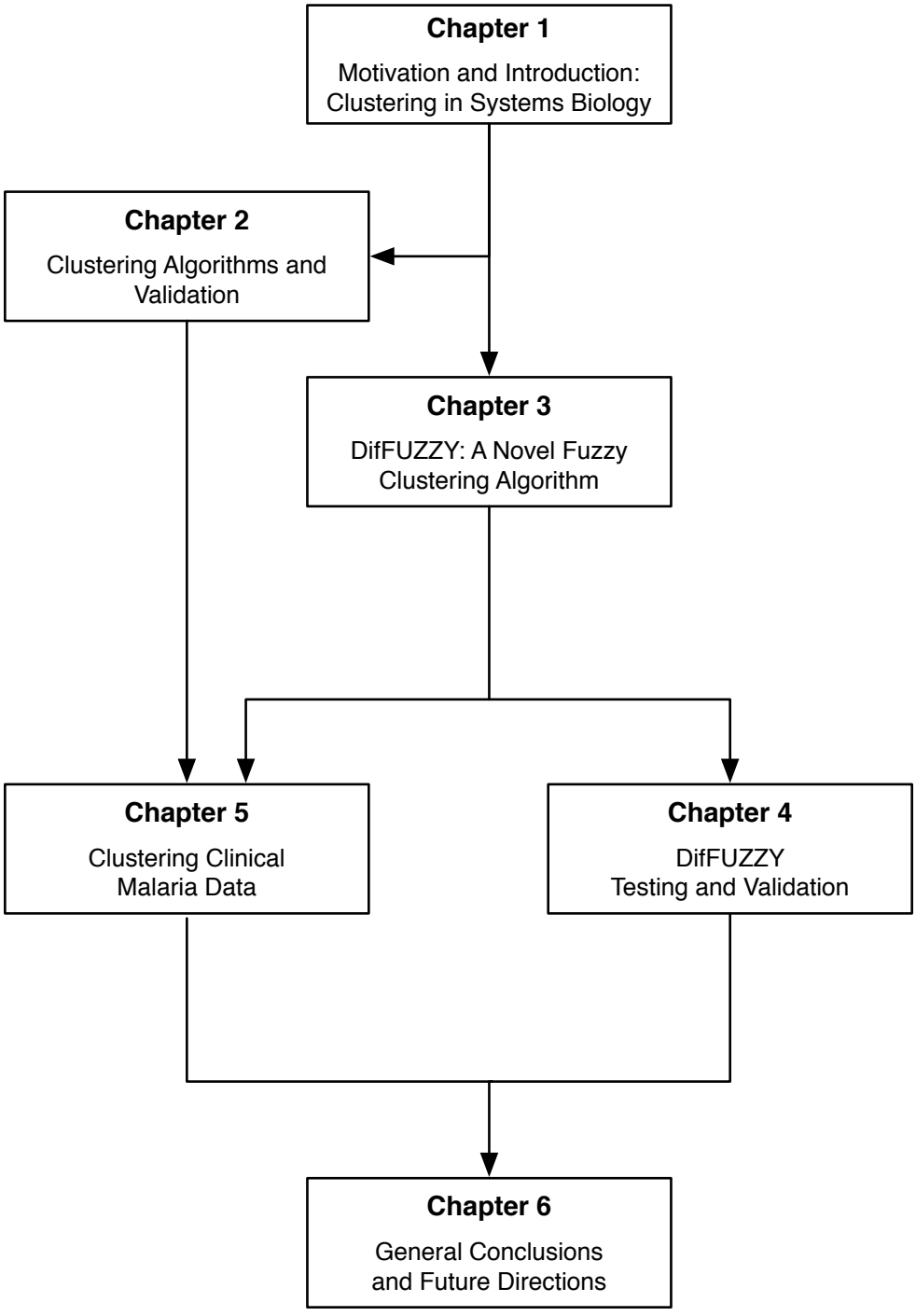


Figure 1.2: Schematic outline of this thesis.

Chapter 2

Clustering Algorithms and Validation

In this chapter we review several standard clustering methods. We discuss some of the most widely used algorithms which provide a foundation for hundreds of other techniques that either combine and/or improve on the more traditional partitioning methods [Wilson et al., 2002]. These methods are classified into hard (crisp) and soft (fuzzy) methods. The former class assigns every data point to only one cluster, whereas the latter allows data points to belong to more than one cluster by membership values which reflect the degree of belonging to each cluster. We also present cluster validation measures that are used to assess the quality of clustering results.

2.1 Clustering Algorithms

There is a vast array of different clustering methods available, yet there are very few which have prevailed and continue to be in widespread usage: these are K-means, hierarchical clustering (HC), Fuzzy C-means (FCM) and its modified version Possibilistic Fuzzy C-means (PCM), self organising maps (SOMs) and spectral clustering [Jain, 2008].

In order to compare different types of clustering methods we first construct three computer generated test 2-D datasets, shown in Fig. 2.1. These datasets have well-defined geometric properties, and form well-defined shapes such as globular clusters (Fig. 2.1 (a)), concentric rings (Fig. 2.1 (b)) or elongated clusters (Fig. 2.1 (c)). The clusters of the *Concentric rings* dataset share the same centroid, so a traditional clustering algorithm that assigns data points to their nearest centroid would fail when attempting to identify such rings as

distinct clusters. In the case of the *Elongated Clusters* dataset, points from different clusters in the vicinity of (0.8,0.6) are closer in Euclidean distance to each other than to the majority of data points in their own clusters. Here too, some clustering techniques fail when attempting to represent a cluster by its mean vector and then assigning data points to the nearest centroid, regardless of the positions of the rest of the data points. In Chapter 4 we compare all the clustering methods presented using additional synthetic test datasets and several real benchmark datasets.

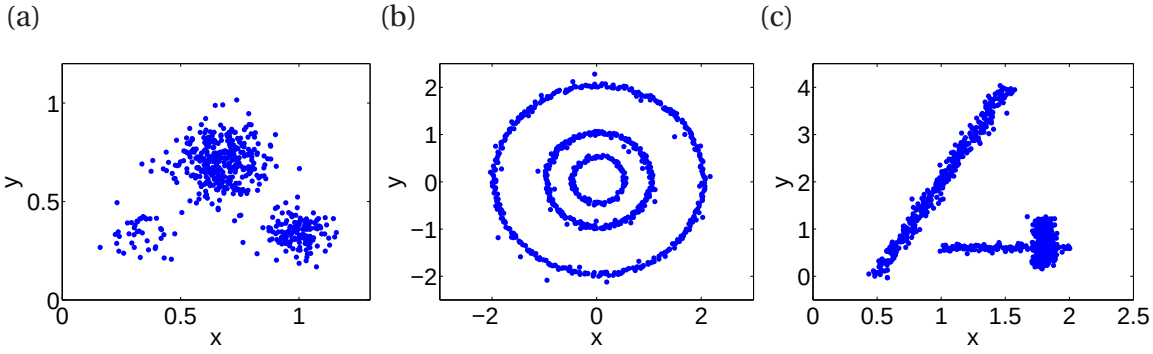


Figure 2.1: Computer generated test datasets. (a) *Globular clusters* dataset. (b) *Concentric rings* dataset. (c) *Elongated clusters* dataset. Every point is described by coordinates (x,y).

2.1.1 Hard clustering algorithms

2.1.1.1 K-means clustering

K-means clustering is a simple and widespread partitional (hard) method that aims to minimise the within-cluster sum of squares, i.e. the sum of the squared distance of every data point to its cluster centre [Azuaje and Dopazo, 2005]. It is the most commonly used clustering method of all [Jain, 2008] and divides the objects into K clusters (K corresponds to the number of clusters to determine, which is provided by the user along with the data) such that a certain metric relative to the centroids of the clusters is minimised [Jensen et al., 2007]. We illustrate the method as follows:

Consider the dataset $\mathbf{A} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_N\}$ where $\mathbf{x}_i \in \mathbb{R}^d, i = 1, 2, \ldots N$. The goal is to find $K \leq N$

sets $\mathbf{C}_k \neq \emptyset$, $\mathbf{C}_k \subset \mathbf{A}$, $k = 1,2,\ldots K$, such that $\mathbf{C}_k \cap \mathbf{C}_l = \emptyset$ for all $l = 1,2,\ldots K$, $l \neq k$, and

$$\mathrm{J} = \sum_{k=1}^{K} \sum_{\mathbf{x}_i \in \mathbf{C}_k} ||\mathbf{x}_i - \mathbf{c}_k||^2 \tag{2.1}$$

is minimised, where $\mathbf{c}_k$ is the mean vector or centroid, which corresponds to the central point of all the observations of that cluster:

$$\mathbf{c}_k = \frac{1}{|\mathbf{C}_k|} \sum_{\mathbf{x}_i \in \mathbf{C}_k} \mathbf{x}_i. \tag{2.2}$$

This is achieved by the following steps [MacQueen, 1967; Azuaje and Dopazo, 2005]:

1. Initialise the mean vectors $\mathbf{c}_k$, $k = 1,2,\ldots K$, for the K clusters by random[1] or by manual assignment.
2. Assign each data point to the cluster represented by its closest mean vector, i.e.
$$\mathbf{C}_k = \left\{ \mathbf{x}_i \in A \,\middle|\, \min_{l=1,2,\ldots K} \{||\mathbf{x}_i - \mathbf{c}_l||\} = ||\mathbf{x}_i - \mathbf{c}_k|| \right\}.$$
3. Recalculate the mean vectors for all the clusters using Eq. (2.2).
4. Repeat steps (2) and (3) until the mean vectors of the clusters remain stationary.

We tested the K-means clustering method with random initialisation of the centroids on three datasets whose intrinsic structure or clustering was known (*Globular clusters*, *Concentric rings* and *Elongated clusters* datasets shown in Fig. 2.1).

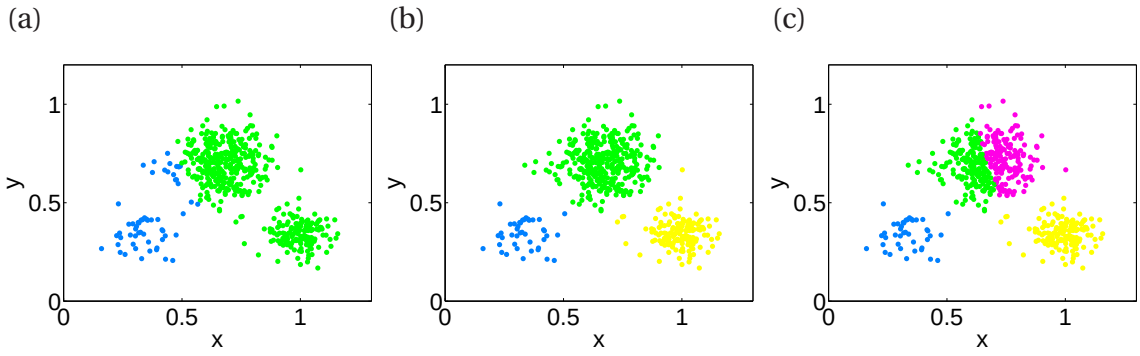


Figure 2.2: K-means clustering applied to the *Globular clusters* dataset (Fig. 2.1 (a)). (a) $K = 2$, (b) $K = 3$ and (c) $K = 4$. Different colours denote the identified clusters.

[1] Examples of random centroid assignments are: sampling from a uniform distribution on the feature space (the strategy used throughout this thesis); randomly selecting feature vectors from the data set [Wang, 2001]; generating the centroids from a Gaussian with mean and variance estimated from the data.

Fig. 2.2 (b) shows that the K-means algorithm successfully identified all three clusters for the dataset shown in Fig. 2.1 (a) when K was set at 3 (in other words, when it was specifically asked to partition the dataset into three clusters). When $K = 2$, it was able to separate the smallest globular cluster from the other two, but misassigned a few data points (Fig. 2.2 (a)). In the case of $K = 4$, it partitioned one cluster into two (Fig. 2.2 (c)).

Results on the other two examples (Figs. 2.3 and 2.4) indicate that K-means clustering is not adequate for certain datasets (such as the ones shown in Figs. 2.1 (b) and 2.1 (c)), regardless of what K value is chosen.

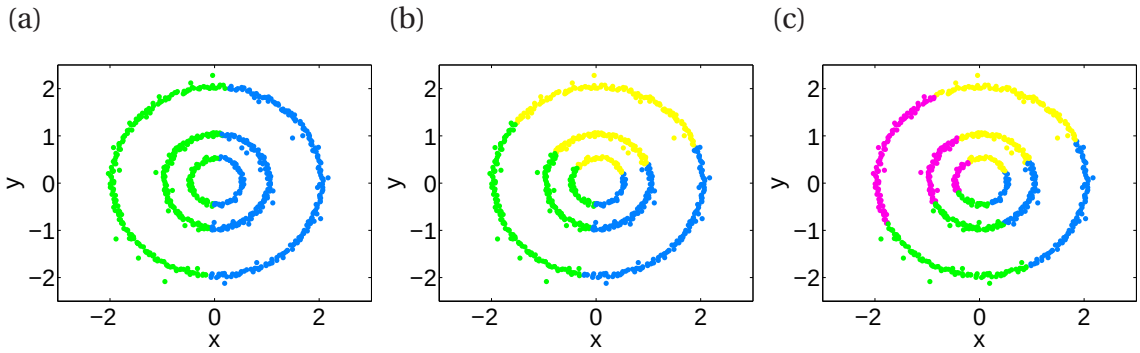


Figure 2.3: K-means clustering applied to the *Concentric rings* dataset (Fig. 2.1 (b)). (a) $K = 2$, (b) $K = 3$ and (c) $K = 4$.

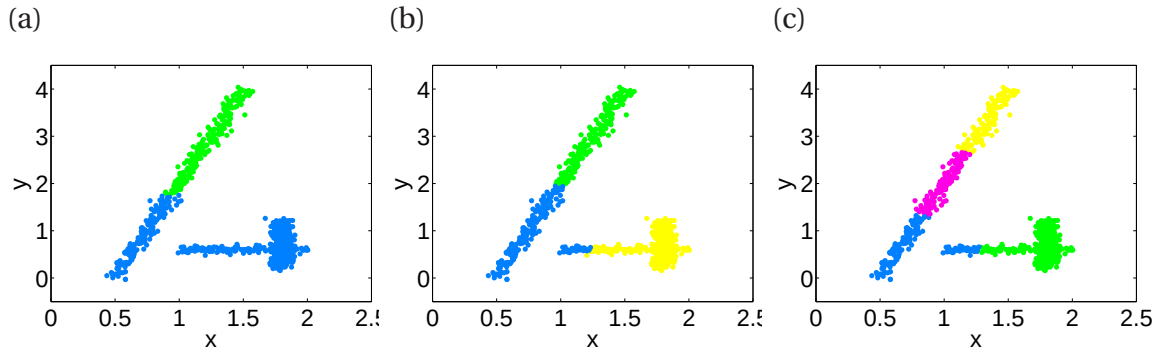


Figure 2.4: K-means clustering applied to the *Elongated clusters* dataset (Fig. 2.1 (c)). (a) $K = 2$, (b) $K = 3$ and (c) $K = 4$.

Strengths and weaknesses. K-means has a number of advantages and disadvantages, yet it remains one of the most widespread and standard choices of clustering [Jain, 2008]. It

is a fast and easy-to-understand clustering method which requires the parameter K to be specified in advance. It works very well for compact, well-separated clusters with an appropriate K value, but it fails if the clusters are not linearly separable [Harel and Koren, 2001]. It is very sensitive to the initialisation of the centroids, and can become trapped in local minima of the objective function, requiring a number of runs to be performed. However, this drawback can be overcome by more effective algorithms to minimise the objective function in Eq. (2.1), such as simulated annealing, guaranteeing a global solution of the optimisation problem [Shokri and Alsultan, 1991]. K-means can also fail when the clusters have dissimilar dispersion and size [Kaufman and Rousseeuw, 1990].

Our synthetic 2-D test datasets here show that K-means is able to identify globular clusters but is unable to identify clusters which are elongated, have centres that are not representative of the clusters, or are non-linearly separable. Also, we observe that all the clusters identified using K-means are of similar size. We will show the results of K-means over a number of real benchmark datasets in Section 4.1.

2.1.1.2 Hierarchical clustering

Hierarchical clustering (HC) is the recursive identification of clusters among the data points resulting in a hierarchy of partitions. It does not require the pre-specification of the number of clusters K [Yao and Li, 2010] and the resulting cluster structure is more informative than a non-hierarchical or "flat" set of clusters [Johnson, 1967; Kaufman and Rousseeuw, 1990].

Hierarchical methods can be agglomerative or divisive. The first type merges clusters iteratively starting with every single data point as a cluster and ending in one large cluster comprising all data points ("bottom up" approach) [Matteucci, 2004], whereas divisive hierarchical clustering corresponds to starting with all the objects in one cluster and subdividing them into smaller clusters ("top down" approach) [Johnson, 1967].

Given the dataset $\mathbf{A} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_N\}$, where $\mathbf{x}_i \in \mathbb{R}^d$, $i = 1, 2, \ldots N$ are the objects to be clus-

tered. We will denote by $\mathbf{C}_k$ the cluster numbered k out of a total of K clusters, which, using this algorithm, corresponds to the clusters obtained after the $(N-K)^{\text{th}}$ step or generation. A hierarchical agglomerative clustering algorithm is described by the steps below [Johnson, 1967]

1. Assign each item to a unique cluster so that there are N clusters, $\mathbf{C}_k$, $k = 1,2,\ldots N$, each containing just one item, $\mathbf{C}_k = \{\mathbf{x}_k, k = 1,2,\ldots N\}$. The initial similarities or distances between the clusters are defined by the distances between their single items.

2. Find the closest (or most similar) pair of clusters and merge them into a single cluster. Now the total number of clusters decreases by one unit (after the first step this value is $N-1$).

3. Calculate the distances between the new cluster and each of the rest of the previously identified clusters using one of the following alternatives:

   (a) single or minimum or connectedness linkage: the distance between one cluster and another cluster is the shortest distance (or greatest similarity) from any member of one cluster to any member of the other.

   (b) complete or maximum or diameter linkage: the distance between one cluster and another cluster is the greatest distance (or smallest similarity) from any member of one cluster to any member of the other.

   (c) average linkage: the distance between one cluster and another cluster is the average distance (or average similarity) from any member of one cluster to any member of the other.

4. Repeat steps 2 and 3 until all items are clustered into a single cluster of size N.

Once the hierarchical tree is complete, which can be visualised as a dendrogram[2], it can be cut at different levels in order to obtain different numbers of clusters. More specifically, if

[2] An example of a dendrogram is shown in Fig. 1.1 (b).

we want K clusters we have to cut at the $(N-K)^{\text{th}}$ merge, those that during the building of the hierarchy were created at the $(N-K)^{\text{th}}$ step of the agglomeration process.

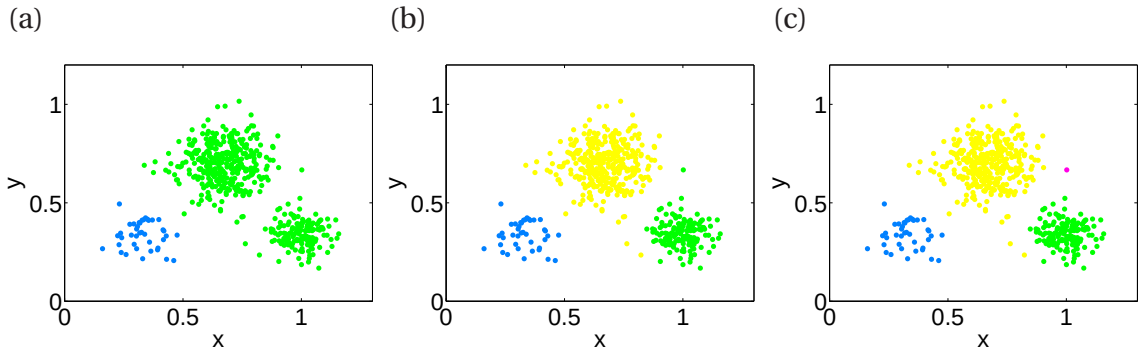


Figure 2.5: Hierarchical clustering applied to the *Globular clusters* dataset (Fig. 2.1 (a)) (Euclidean distance, average linkage). (a) $K=2$, (b) $K=3$ and (c) $K=4$.

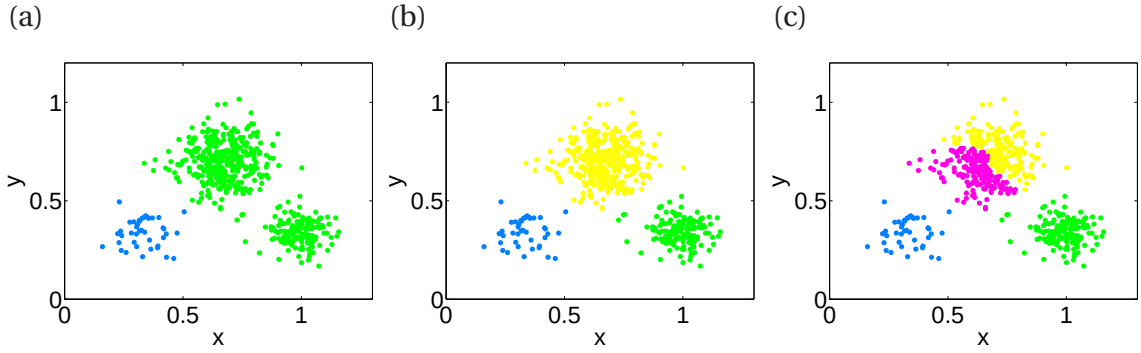


Figure 2.6: Hierarchical clustering applied to the *Globular clusters* dataset (Fig. 2.1 (a)) (Euclidean distance, complete linkage). (a) $K=2$, (b) $K=3$ and (c) $K=4$.

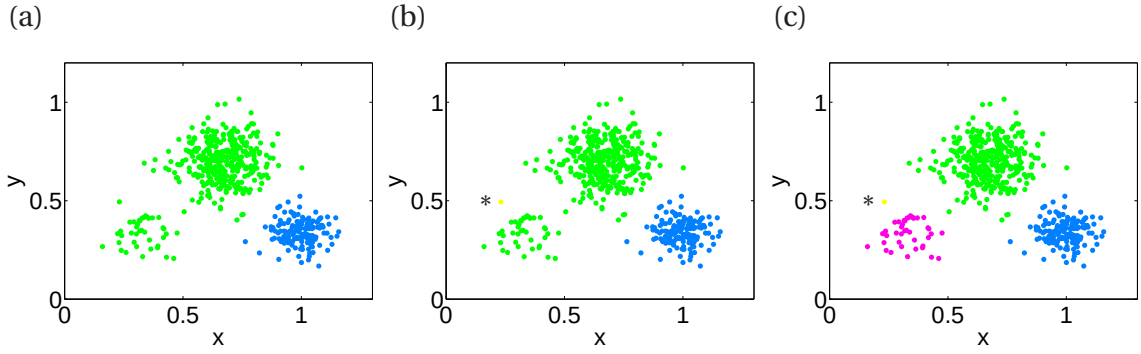


Figure 2.7: Hierarchical clustering applied to the *Globular clusters* dataset (Fig. 2.1 (a)) (Euclidean distance, single linkage). (a) $K=2$, (b) $K=3$ and (c) $K=4$. In (b) and (c) right next to the black asterisk there is a single yellow data point with coordinates (0.2,0.5) making up a cluster.

Upon application of hierarchical clustering to the *Globular clusters* dataset (Fig. 2.1 (a)), we identify the three clusters when using average and complete linkage ($K = 3$), Figs. 2.5 (b) and 2.6 (b), but not when using single linkage, Fig. 2.7 (b).

The hierarchical clustering algorithm fails to identify the three globular clusters when using single linkage ($K = 3$), Fig. 2.7 (b), caused by the presence of an intermediate data point between the largest cluster in the middle and the left bottom cluster, merging both cluster and leaving isolated one single data point further away from its neighbours than the intermediate data point to both clusters.

We can also see that when using $K = 4$, the three linkage methods give different results; hierarchical clustering with complete linkage separates the upper middle cluster into two clusters (Figs. 2.6 (c)), very similar to K-means (Fig. 2.2(c)), while using either average or single linkage results in a single element being identified as a cluster (pink data point in Fig. 2.5 (c) and yellow data point in Fig. 2.7 (c)).

Upon consideration of the test datasets shown in Figs. 2.1 (b) and 2.1 (c) we can observe that, like K-means clustering, hierarchical clustering fails to assign data points to the correct cluster, especially points that were on the border of two clusters (Figs. 2.8 to 2.13).

In particular, like K-means clustering, hierarchical clustering using the average linkage criterion was unable to identify the original concentric rings and elongated clusters (Figs. 2.8 (c) and 2.11 (c), respectively). Hierarchical clustering based on the longest linkage criterion (or complete linkage) performed poorly when identifying the three original clusters for these two datasets (Figs. 2.9 and 2.12).

This was not the case with hierarchical clustering using the single linkage criterion, since this method was more successful at identifying clusters in the *Concentric rings* and the *Elongated clusters* datasets than K-means clustering (Fig. 2.10 versus 2.3 and Fig. 2.13 versus 2.4, respectively). This shows that the shortest linkage criterion can be useful in scenarios when clusters are elongated and irregular, such as these two datasets, although it does

not perform as well when clusters are compact and regular, such as the *Globular clusters* dataset, Fig. 2.7 (b).

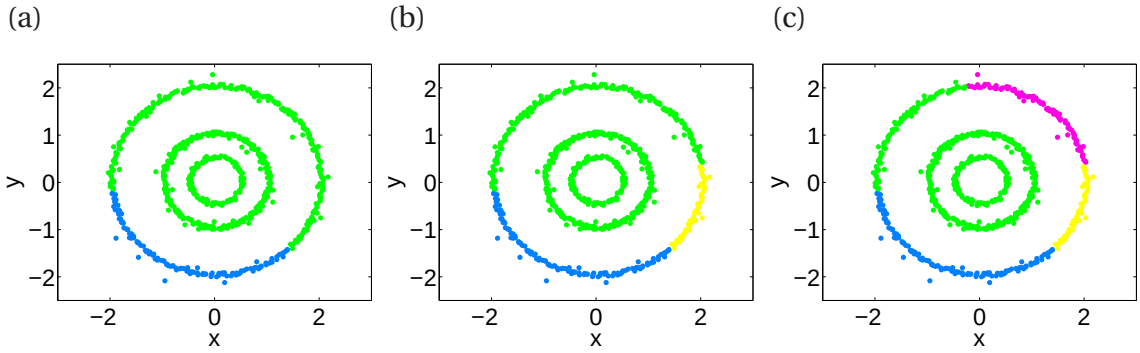


Figure 2.8: Hierarchical clustering applied to the *Concentric rings* dataset (Fig. 2.1 (b)) (Euclidean distance, average linkage). (a) $K=2$, (b) $K=3$ and (c) $K=4$.

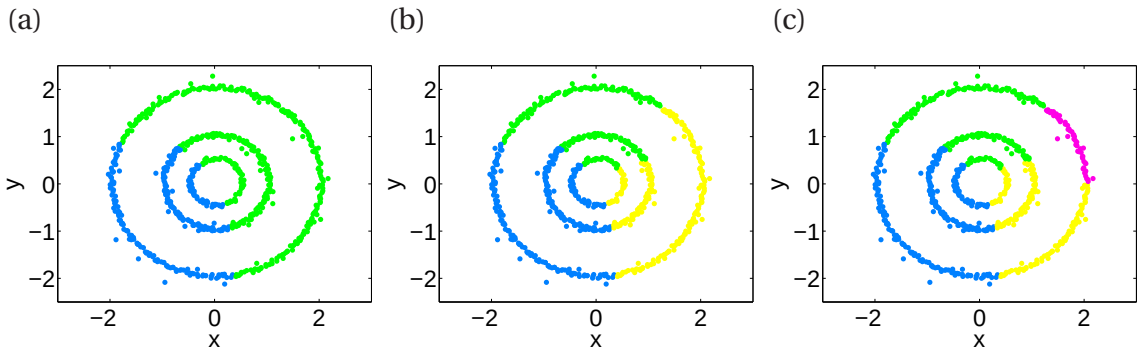


Figure 2.9: Hierarchical clustering applied to the *Concentric rings* dataset (Fig. 2.1 (b)) (Euclidean distance, complete linkage). (a) $K=2$, (b) $K=3$ and (c) $K=4$.

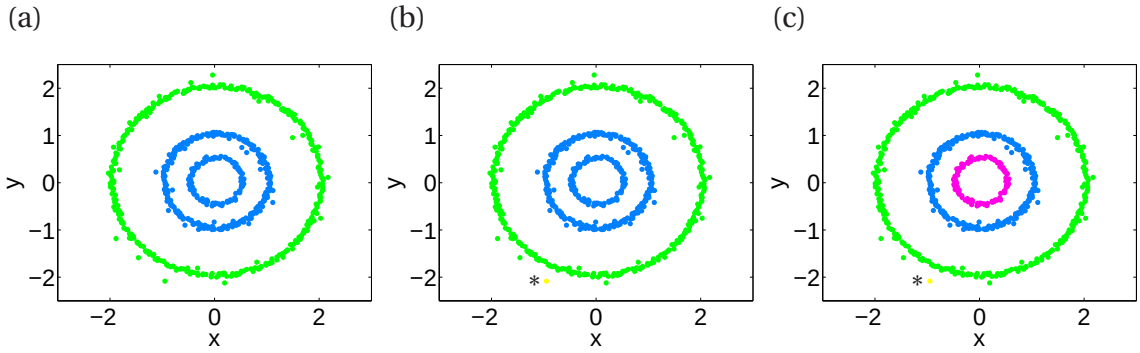


Figure 2.10: Hierarchical clustering applied to the *Concentric rings* dataset (Fig. 2.1 (b)) (Euclidean distance, single linkage). (a) $K=2$, (b) $K=3$ and (c) $K=4$. In (b) and (c) right next to the black asterisk there is a single yellow data point with coordinates (-0.9,-2.0) making up a cluster.

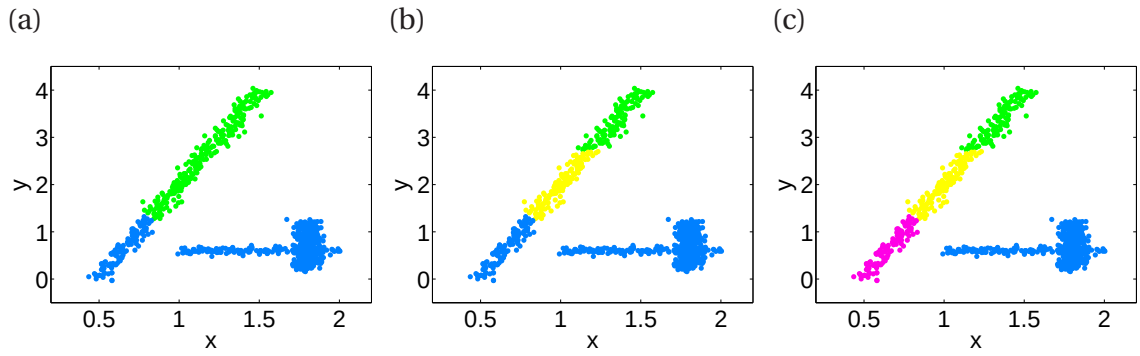


Figure 2.11: Hierarchical clustering applied to the *Elongated clusters* dataset (Fig. 2.1 (c)) (Euclidean distance, average linkage). (a) $K=2$, (b) $K=3$ and (c) $K=4$.

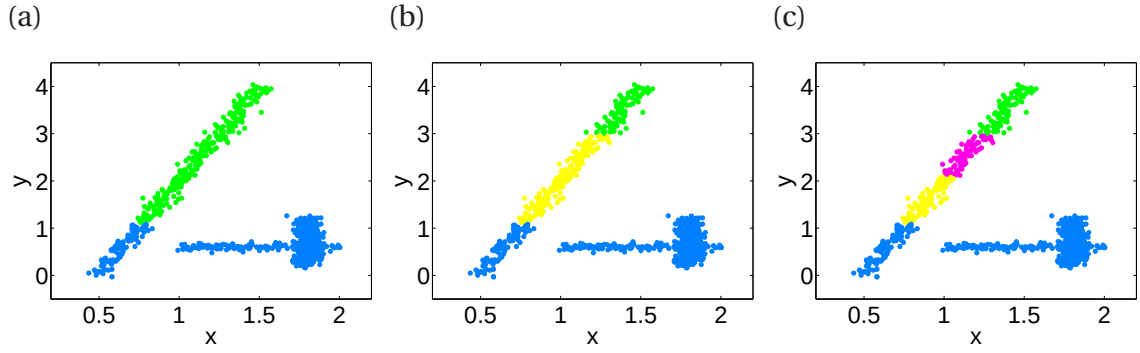


Figure 2.12: Hierarchical clustering applied to the *Elongated clusters* dataset (Fig. 2.1 (c)) (Euclidean distance, complete linkage). (a) $K=2$, (b) $K=3$ and (c) $K=4$.

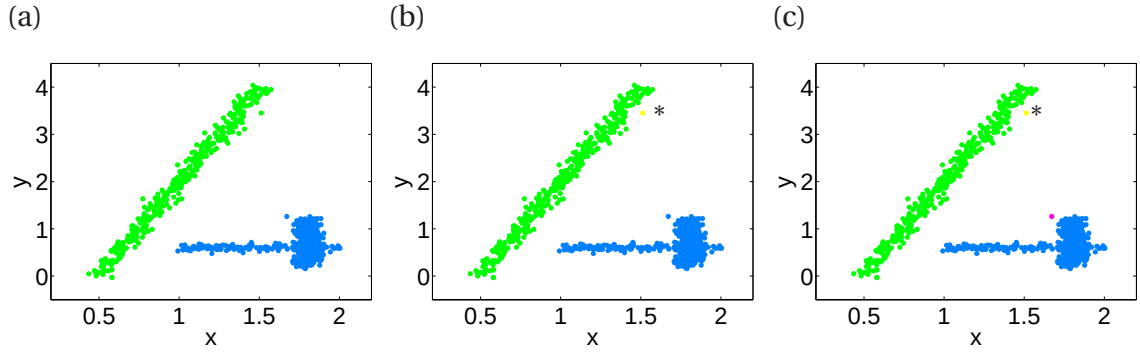


Figure 2.13: Hierarchical clustering applied to the *Elongated clusters* dataset (Fig. 2.1 (c)) (Euclidean distance, single linkage). (a) $K=2$, (b) $K=3$ and (c) $K=4$. In (b) and (c) next to the black asterisk there is a single yellow data point with coordinates (1.5,3.5) making up a cluster.

Moreover single linkage can produce an artifact called the "chaining phenomenon", which occurs when clusters are forced together due to single data points being close to each other regardless of the positions of the other data points in the cluster [De Brevern et al., 2004].

Strengths and weaknesses. Hierarchical clustering is frequently used in biology to capture taxonomies such as kingdom, phylum, genus, species, among others, and it is also often chosen when the number of clusters is not known *a priori*.

HC has played a major role in clustering expression data and in identifying clusters of genes and similarities between patients or samples; it has provided the basis for uncovering the existence of multiple pathways of tissue destruction and repair in patients with rheumatoid arthritis [Van Der Pouw Kraan et al., 2003] and it has also helped define molecular profiles of breast cancer cell lines, which has allowed the cataloging of cell lines and the elucidation of the relationship of such cell lines to specific breast cancer subtypes [Kao et al., 2009].

The hierarchy of the clustering results can be more informative than a flat partitioning if the data present such structure. However, the final result of hierarchical clustering is highly dependent on the type used (agglomerative or divisive) and the linkage function chosen (single, complete or average) [Tan et al., 2006].

Agglomerative or bottom-up hierarchical algorithms are most often used and have a better search landscape, make decisions based on local patterns rather than in the global distribution and are more flexible than their top-down counterparts. However, agglomerative algorithms can be less efficient and earlier decisions will be carried over to the rest of the tree [Kauchak and Dasgupta, 2004; Manning et al., 2008; Varshavsky et al., 2008].

The choice of linkage function (single, complete or average) can have a high impact on the results of hierarchical clustering. Their performance depends on the data being clustered. Complete linkage usually gives compact clusters, single linkage returns more separated and elongated clusters while average linkage gives clusters which are neither too elongated nor too round and compact. When clustering gene expression data, complete and average linkage are recommended over single linkage [Simon, 2003; Varshavsky et al., 2008].

In general terms, if no information is known *a priori* it is recommended to use average linkage since it is not affected by outliers as is complete linkage and does not present the

chaining phenomenon of single linkage, which is discussed above with the help of the test datasets from Fig. 2.1. Average linkage produced clusters of similar size, like K-means, while single and complete linkage yielded clusters of uneven size, including singleton clusters (clusters made up of only one data point).

In Figs. 2.3 and 2.4 we showed two test datasets that could not be properly clustered using the K-means algorithm. In Figs. 2.8-2.13 we also demonstrate that the clusters in the same datasets (*Concentric rings* and *Elongated clusters*) cannot be correctly identified using the hierarchical clustering method without additional *a priori* information. This failure is due to hierarchical clustering making local decisions on which data points/clusters to merge at every time step, without taking into consideration the rest of the data, and such merges can drastically influence the rest of the hierarchy. This motivated us to search for additional clustering methods that could cluster datasets similar to these two.

2.1.1.3 Spectral clustering

Spectral clustering is a flat (non-hierarchical) and hard partitioning method which clusters points using eigenvectors (or the spectrum) of matrices derived from the data distances. This allows a first dimensionality reduction to subsequently perform clustering in the reduced set of dimensions [Kaufman and Rousseeuw, 1990; Ding, 2004]. A very common use for this technique is found in image segmentation.

As a graph based partitioning problem, spectral clustering represents data points as nodes and models the local neighbourhood relationships between the data assigning edges within nodes [Luxburg, 2007]. Edges can be weighted if every pair of nodes is connected by an edge (fully connected graph), or not, and only nodes within a given radius of each other or among its k-nearest neighbours (kNN) will be linked (ε-neighbourhood graph and k-nearest neighbour graph, respectively).

Donath and Hoffman [1973] first suggested partitioning graphs based on eigenvectors of their adjacency matrix. Since then several algorithms that implement and modify this idea

have been published, including those based on the normalised graph Laplacian [Ng et al., 2002] as well as the unnormalised graph Laplacian matrix, such as the geometrically motivated algorithm of Niyogi [2003].

A summary of the spectral clustering algorithm by Ng et al. [2002] is as following:

Given data points $\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_N \in \mathbb{R}^d$; K, the number of clusters to construct and n the number of eigenvectors to use, respectively, construct the K clusters following these steps:

1. Construct the similarity matrix $\mathbf{S} \in \mathbb{R}^{N \times N}$:

$$S_{i,j} = \exp\left[-\frac{\left|\left|\mathbf{x}_i - \mathbf{x}_j\right|\right|^2}{\beta}\right] \tag{2.3}$$

   where $\beta > 0$ is an internal parameter which controls the width of the local neighbourhoods and $||\,||$ corresponds to the Euclidean distance.

2. Define the degree matrix $\mathbf{D}$ with diagonal elements $D_{i,i} = \sum_{j=1}^{N} S_{i,j}, i = 1,2,\ldots N$.

3. Compute the matrix $\mathbf{L} \in \mathbb{R}^{N \times N}$ as:

$$\mathbf{L} = \mathbf{D}^{-1/2}(\mathbf{D} - \mathbf{S})\mathbf{D}^{-1/2}. \tag{2.4}$$

4. Compute the first n eigenvectors of $\mathbf{L}$ (those with the largest eigenvalues) and place them in the matrix $\mathbf{V} \in \mathbb{R}^{N \times n}$ whose columns correspond to these eigenvectors.

5. Normalise the rows of $\mathbf{V}$ to have unit length.

6. Let $\mathbf{h}_i \in \mathbb{R}^n$, $i = 1,2,\ldots N$ be the vector corresponding to the i^{th} row of $\mathbf{V}$. These vectors correspond to a scaled mapping of the original data points into the first n eigenvectors of $\mathbf{L}$.

7. Use K-means clustering to partition the points $\mathbf{h}_i, i = 1,2,\ldots N \in \mathbb{R}^n$ into K clusters $\hat{C}_1, \hat{C}_2, \ldots \hat{C}_K$.

The output of this algorithm is K clusters, where $C_k = \{i \mid \mathbf{h}_i \in \hat{C}_k\}$, $k = 1,2,\ldots K$.

An intuitive explanation of why this method works is as follows. First the similarity matrix **S** is constructed, where $S_{i,j} = 1$ for $\mathbf{x}_i = \mathbf{x}_j$ and $S_{i,j} \sim 0$ for very dissimilar $\mathbf{x}_i$ and $\mathbf{x}_j$. Using this similarity matrix we can embed the original data into a lower dimensional Euclidean space preserving the local neighbourhoods. In this lower dimensional space K-means clustering can be performed and return the clusters of the original data points [van Leeuwen et al., 2007; Bah, 2008].

The question then arises as to how to map the original graph with the adjacency matrix **S** into a smaller dimensional Euclidean space. For simplicity, let us consider mapping the graph into a line $\mathbf{y} = (y_1, y_2, \ldots, y_N)$, $y_i \in \mathbb{R}, i = 1, 2, \ldots N$ so that $\mathbf{x}_i \to y_i$ for all the elements in A, and so that the points that were near in the multidimensional dataset remain as close together as possible [Bah, 2008]. This can be achieved by minimising the objective function with regards to y_i and y_j

$$\sum_{i=1}^{N}\sum_{j=1}^{N}\left(y_i - y_j\right)^2 S_{i,j}, \tag{2.5}$$

where y_i and y_j are data points in the mapped line. When two data points are very close to each other, $S_{i,j}$ is nearly 1, penalising $\left(y_i - y_j\right)^2$, and thus forcing y_i to be close to y_j, and when the data points are far away from each other, $S_{i,j}$ is nearly zero, so it will not be penalised if they are mapped far away from each other [Niyogi, 2003; van Leeuwen et al., 2007; Bah, 2008].

The sum in (2.5) can also be written as [van Leeuwen et al., 2007; Bah, 2008]

$$\begin{aligned}
\sum_{i=1}^{N}\sum_{j=1}^{N}\left(y_i - y_j\right)^2 S_{i,j} &= \sum_{i=1}^{N}\sum_{j=1}^{N}\left(y_i^2 - 2y_i y_j + y_j^2\right) S_{i,j} && (2.6)\\
&= \sum_{i=1}^{N} y_i^2 D_{i,i} + \sum_{j=1}^{N} y_j^2 D_{j,j} - 2\sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j S_{i,j} && (2.7)\\
&= 2\mathbf{y}^T\mathbf{L}\mathbf{y} && (2.8)
\end{aligned}$$

where **D** is a symmetric matrix with diagonal components $D_{i,i} = \sum_{j=1}^{N} S_{i,j}$ and $\mathbf{L} = \mathbf{D} - \mathbf{S}$.

Since **S** and **D** are symmetric, **L** is also symmetric, and since

$$\sum_{i=1}^{N}\sum_{j=1}^{N}\left(y_i - y_j\right)^2 S_{i,j} \geq 0,$$

we have that $\mathbf{y}^{\mathrm{T}}\mathbf{L}\mathbf{y} \geq 0$ for any y, *i.e.* **L** is positive-semidefinite [van Leeuwen et al., 2007].

Since

$$\mathbf{L}\mathbf{v}_1 = (\mathbf{D}-\mathbf{S})\mathbf{v}_1 = \mathbf{D}\mathbf{v}_1 - \mathbf{S}\mathbf{v}_1 \tag{2.9}$$

$$= \begin{bmatrix} \sum_{j=1}^{N} S_{1,j} & & \\ & \ddots & \\ & & \sum_{j=1}^{N} S_{N,j} \end{bmatrix} \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} - \begin{bmatrix} S_{1,1} & \dots & S_{1,N} \\ \vdots & \ddots & \vdots \\ S_{N,1} & \dots & S_{N,N} \end{bmatrix} \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} \tag{2.10}$$

$$= \begin{pmatrix} \sum_{j=1}^{N} S_{1,j} \\ \vdots \\ \sum_{j=1}^{N} S_{N,j} \end{pmatrix} - \begin{pmatrix} \sum_{j=1}^{N} S_{1,j} \\ \vdots \\ \sum_{j=1}^{N} S_{N,j} \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix} = 0 \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}, \tag{2.11}$$

the smallest eigenvalue of **L** is $\lambda_1 = 0$, with a corresponding eigenvector $\mathbf{v}_1 = \mathbf{1} = (1,1,\dots 1)^{\mathrm{T}}$ [Bah, 2008]. Furthermore, since **L** is positive semi-definite iff λ_i is greater than or equal to zero for all i, it has N non-negative, real-valued eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_N$ [van Leeuwen et al., 2007].

Expressing **y** in terms of the eigenvectors of **L**, we have $\mathbf{y} = a_1\mathbf{v}_1 + a_2\mathbf{v}_2 + \ldots a_N\mathbf{v}_N$, and since the eigenvectors are orthogonal (and assuming they have also been normalised to 1) the term in Eq. (2.8) becomes

$$\begin{aligned} 2\mathbf{y}^{\mathrm{T}}\mathbf{L}\mathbf{y} &= 2(a_1\mathbf{v}_1 + a_2\mathbf{v}_2 + \ldots a_N\mathbf{v}_N)^{\mathrm{T}}\,\mathbf{L}(a_1\mathbf{v}_1 + a_2\mathbf{v}_2 + \ldots a_N\mathbf{v}_N) \\ &= 2(a_1\mathbf{v}_1 + a_2\mathbf{v}_2 + \ldots a_N\mathbf{v}_N)^{\mathrm{T}}\,[a_1\mathbf{L}\mathbf{v}_1 + a_2\mathbf{L}\mathbf{v}_2 + \ldots a_N\mathbf{L}\mathbf{v}_N] \\ &= 2(a_1\mathbf{v}_1 + a_2\mathbf{v}_2 + \ldots a_N\mathbf{v}_N)^{\mathrm{T}}\,[a_1\lambda_1\mathbf{v}_1 + a_2\lambda_2\mathbf{v}_2 + \ldots a_N\lambda_N\mathbf{v}_N] \\ &= 2\left(\lambda_1 a_1^2 + \lambda_2 a_2^2 + \ldots \lambda_N a_N^2\right) = 2\left(\lambda_2 a_2^2 + \lambda_3 a_3^2 + \ldots \lambda_N a_N^2\right) \geq 0. \end{aligned} \tag{2.12}$$

From Eqs. (2.8) and (2.12) the objective function in Eq. (2.5) achieves its minimum 0 when $a_2 = a_3 = \cdots = a_N = 0$, i.e. $\mathbf{y} = \mathbf{v}_1$. Thus, $\mathbf{v}_1 = \mathbf{1}$ is the minimiser. However, this eigenvector

does not provide us with useful information, since it corresponds to a mapping of every data point to a single point [van Leeuwen et al., 2007; Bah, 2008]. Adding the orthogonality constraint

$$\mathbf{y}^{\mathrm{T}}\mathbf{D1} = 0 \tag{2.13}$$

to avoid this trivial mapping, the following smallest eigenvector $\mathbf{v}_2$ will be the minimiser of Eq. (2.8) subject to Eq. (2.13) [Bah, 2008].

The argument above is valid for embedding the graph into a line (1 dimension). Following a similar line of reasoning for the generalised problem of embedding the graph into an n-dimensional Euclidean space [Belkin and Niyogi, 2003], it can be shown that using the n smallest nontrivial eigenvectors $\mathbf{v}_2, \mathbf{v}_3, \ldots \mathbf{v}_{n+1}$ of $\mathbf{L}$ leads us to obtain a meaningful mapping into n-dimensions where the shortest distances in the original data space are preserved [Luxburg, 2007; Bah, 2008].

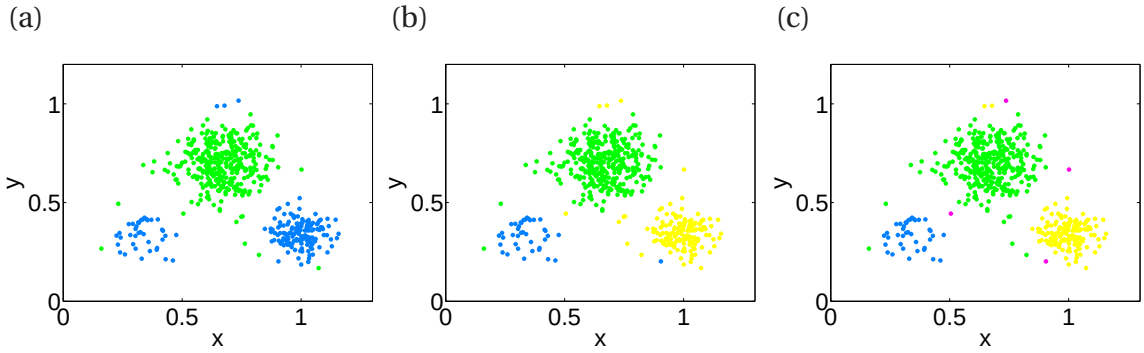


Figure 2.14: Spectral clustering applied to the *Globular clusters* dataset (Fig. 2.1 (b)). (a) $K = 2$, (b) $K = 3$ and (c) $K = 4$.

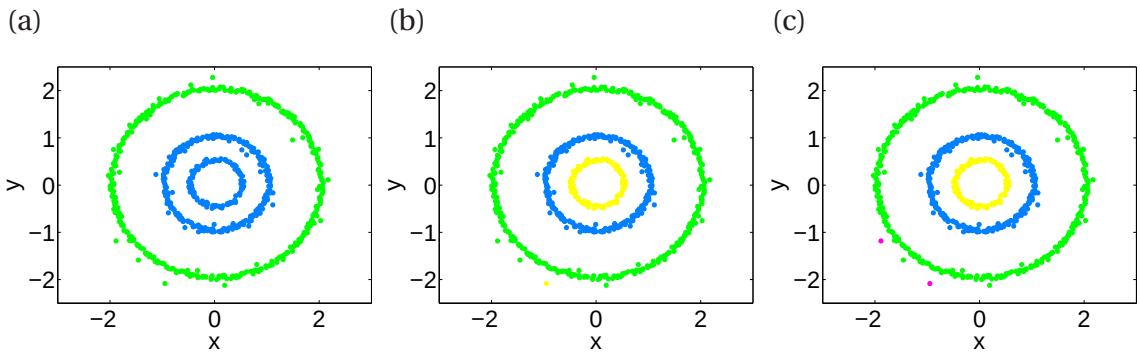


Figure 2.15: Spectral clustering applied to the *Concentric rings* dataset (Fig. 2.1 (b)). (a) $K = 2$, (b) $K = 3$ and (c) $K = 4$.

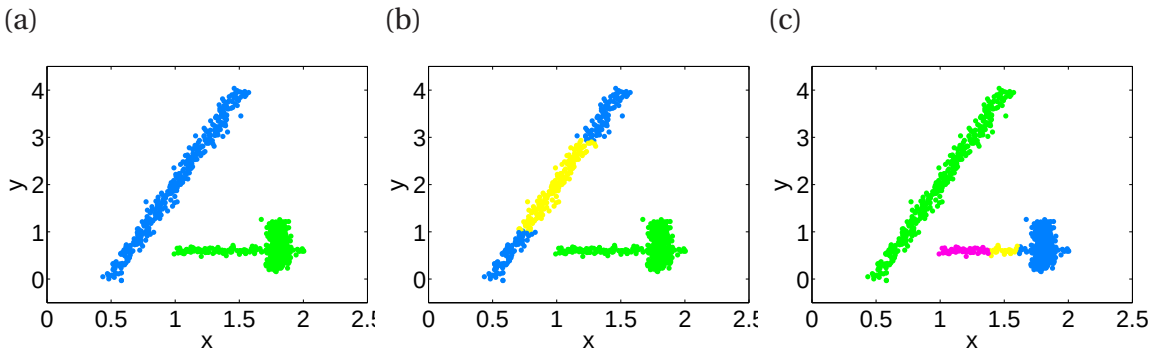


Figure 2.16: Spectral clustering applied to the *Elongated clusters* dataset (Fig. 2.1 (c)). (a) $K = 2$, (b) $K = 3$ and (c) $K = 4$.

Figs. 2.14 (b), 2.15 (b) and 2.16 (a) show that spectral clustering can separate the globular clusters in Fig. 2.1 (a), rings in the dataset shown in Fig. 2.1 (b) and the elongated clusters in Fig. 2.1 (c), respectively. The projection of such complex datasets in the eigenspace of the graph Laplacian is easily separable by traditional clustering methods, such as K-means [Ding, 2004]. However, this technique is crisp, making each data point a member of just one cluster, while in some biological problems, data points could belong to more than one cluster, such as genetic expression where genes typically participate in more than one pathway or have several functions [Dembélé and Kastner, 2003].

Strengths and weaknesses. One of the main advantages of spectral clustering is that it allows the identification of non-linearly separable clusters. However, it suffers from some fundamental limitations, mainly that the first eigenvectors of the graph adjacency matrix cannot identify clusters of different scales, density and size [Nadler and Galun, 2007]. Some real datasets where spectral clustering does not perform as well as other common clustering techniques are shown in Section 4.1.2.

Spectral clustering has been applied to different areas of systems biology; it has contributed to the discovery of modification profiles of peptidomes (measured in mass shifts in Dalton) [Menschaert et al., 2009] and to find cell populations in lymphoma diagnosis using flow cytometry data [Zare et al., 2010].

2.1.1.4 Self-organising maps

A self-organising map (SOM) is a kind of artificial neural network that allows data to be mapped into a low dimensional grid. The data are represented by neurons (processing units) with associated weight vectors (also called prototype or code vectors) aligned on a pre-defined grid. The most commonly used grids are the 2-D rectangular and hexagonal lattices, shown in Figs. 2.17 (a) and 2.17 (b), respectively. SOMs are used for many purposes, such as data visualisation, classification, and clustering [Filippone et al., 2008].

Given a dataset $\mathbf{A} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_N\}$ where $\mathbf{x}_i \in \mathbb{R}^d$, $i = 1, 2, \ldots N$, and the weight vectors $\hat{\mathbf{B}} = \{\mathbf{w}_1, \mathbf{w}_2, \ldots \mathbf{w}_{\hat{N}}\}$ where $\mathbf{w}_j \in \mathbb{R}^d$, $j = 1, 2, \ldots \hat{N}$ and where generally $\hat{N} \ll N$, SOMs are built following the steps below:

1. **Initialisation:** Initialise the weight vectors (for example, randomly).

2. **Sampling:** Randomly choose an input data point ($\mathbf{x}_r \in \mathbf{A}$).

3. **Matching:** Determine the best matching unit (BMU) or winning neuron ($\mathbf{w}_g \in \hat{\mathbf{B}}$), i.e. the neuron whose weight vector is closest (in Euclidean distance) to the input, $\mathbf{x}_r$.

4. **Updating:** Update all weight vectors $\mathbf{w}_i$, $i = 1, 2, \ldots \hat{N}$, of the BMU and its neighbours, using a given update function, such as

$$\mathbf{w}_i(t+1) = \mathbf{w}_i(t) + \theta\left(\mathbf{w}_g, \mathbf{w}_i, t\right)\alpha(t)(\mathbf{x}_r - \mathbf{w}_i(t)), \tag{2.14}$$

   where $\alpha(t)$ is called the learning coefficient, and is a monotonically decreasing function of t while $\theta\left(\mathbf{w}_g, \mathbf{w}_i, t\right)$ is the neighbourhood function, which depends on the grid distance from the neuron $\mathbf{w}_i$ and the BMU $\mathbf{w}_g$, and also decreases with time (typically a Gaussian function), such that for small t values, the changes in the weight vectors will be large, while for larger values of t, the changes will be smaller at each iteration [Filippone et al., 2008].

5. **Continuation:** Increase the time parameter ($t \rightarrow t+1$). If $t < t_{\max}$, go back to step number 2, otherwise stop.

6. **Clustering:** Once the SOM has been built, the weight vectors can be clustered using any clustering method, or can be used as initial mean vectors for clustering techniques which require them, such as K-means [Chi and Yang, 2008].

The parameter $t_{\max}$ is usually a very large number, $t_{\max} \gg N$, so that each data point will be considered as input several times on average [Filippone et al., 2008].

SOMs iteratively adapt (move) the weight vectors towards the original data points. At each iteration, the neuron that is closest (according to a distance metric) to a randomly chosen data point is modified (by changing its weight vector), and the adaptation is propagated to the neighbours of this neuron.

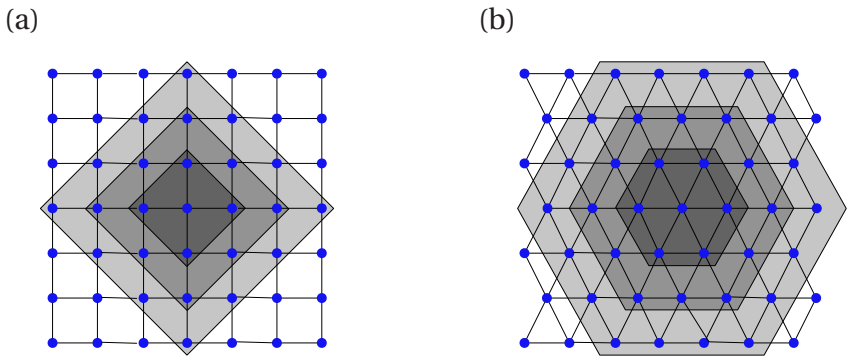


Figure 2.17: (a) Rectangular grid, 4 neighbours. (b) Hexagonal grid, 6 neighbours. The superimposed shades of grey, from lighter to darker, represent the neighbourhoods of radius 3, 2 and 1 respectively.

During training, the map reorganises itself using a neighbourhood function, thereby preserving the local topology, as shown schematically in Fig. 2.18.

After the training (adaptation) of the map, the weight vectors can be clustered using traditional clustering techniques, such as K-means, to assign the data points to the cluster to which the BMU belongs.

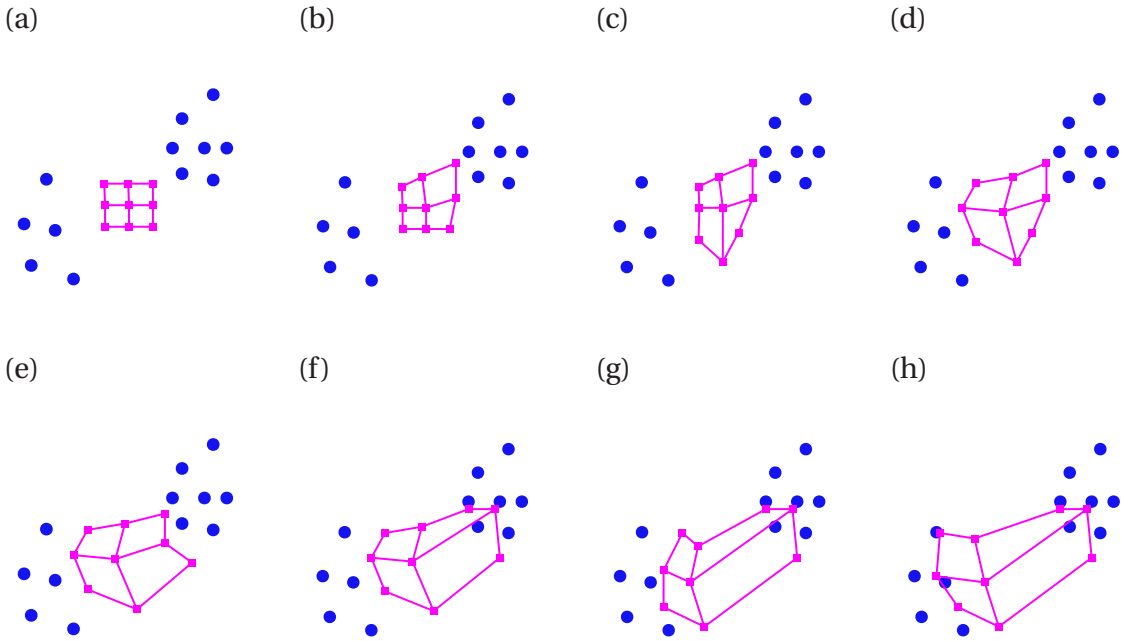


Figure 2.18: Schematic representation of the training of a SOM using a rectangular grid of 3 by 3 nodes in 8 steps, sequentially presented in (a)-(h). These steps precede the clustering of the weight vectors.

Figs. 2.19, 2.20 and 2.21 show the performance of SOM clustering on the *Globular clusters*, *Concentric rings* and *Elongated clusters* datasets, respectively, for a Gaussian neighbourhood function

$$\theta(\mathbf{w}_g, \mathbf{w}_i)(t) = \exp\left(\frac{-d_{ig}^2}{2\sigma_t^2}\right), \tag{2.15}$$

where σ_t is the neighbourhood radius at time t, chosen for this example to be a constant value of 2, and d_{ig} is the distance between nodes i and g in the map. The learning function used is linear: $\alpha(t) = \alpha_0\left(1 - \frac{t}{t_{\max}}\right)$, where $\alpha_0 = 1$ is the initial learning rate and $t_{\max}$ corresponds to the training length or number of time steps ($t_{\max} = 300$).

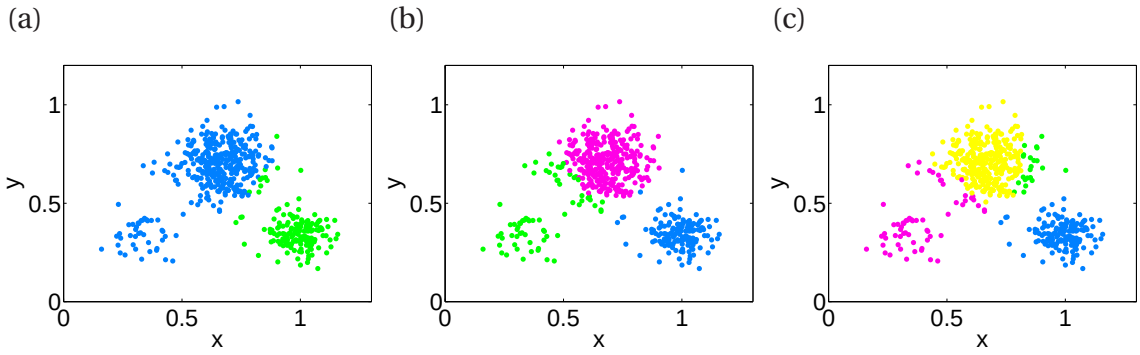


Figure 2.19: SOM clustering applied to the *Globular clusters* dataset (Fig. 2.1 (a)) (Euclidean distance, hexagonal lattice with 15×15 neurons, 300 time steps, K-means, random initialisation, linear learning rate, Gaussian neighbourhood). (a) $K = 2$, (b) $K = 3$ and (c) $K = 4$. Colour code as in Fig. 2.2.

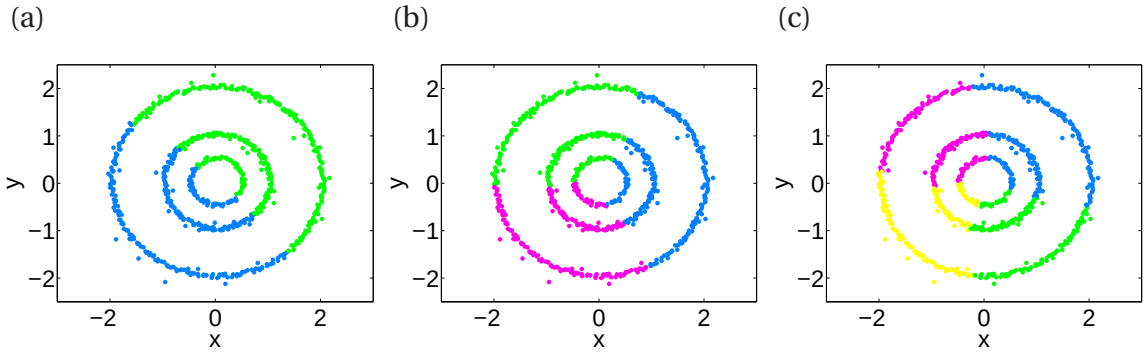


Figure 2.20: SOM clustering applied to the *Concentric rings* dataset (Fig. 2.1 (b)) (Euclidean distance, hexagonal lattice with 15×15 neurons, 300 time steps, K-means, random initialisation, linear learning rate, Gaussian neighbourhood). (a) $K = 2$, (b) $K = 3$ and (c) $K = 4$. Colour code as in Fig. 2.2.

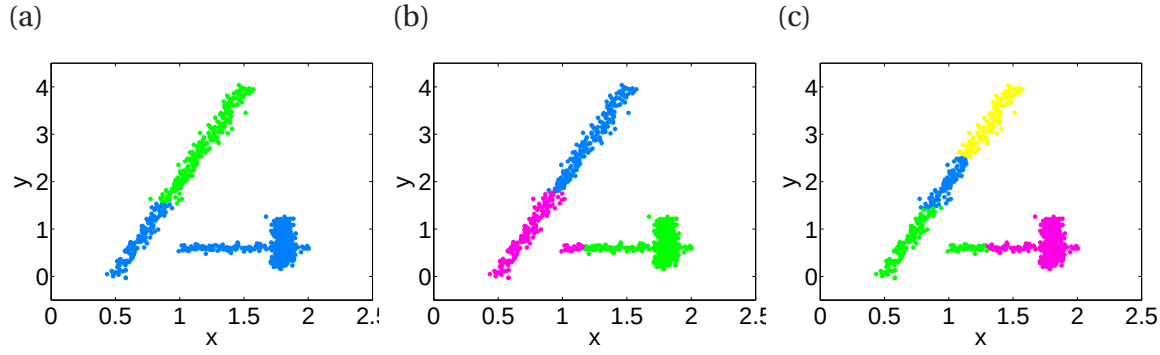


Figure 2.21: SOM clustering applied to the *Elongated clusters* dataset (Fig. 2.1 (c)) (Euclidean distance, hexagonal lattice with 15×15 neurons, 300 time steps, K-means, random initialisation, linear learning rate, Gaussian neighbourhood). (a) $K = 2$, (b) $K = 3$ and (c) $K = 4$. Colour code as in Fig. 2.2.

Similarly to K-means, SOMs showed to be able to identify the globular clusters (Fig. 2.19 (b)) but unable to identify the concentric rings (Figs. 2.20 (b)) and the elongated clusters (2.21 (b)), and instead divided the rings in slices and the diagonal cluster into two. SOMs failed when clustering non-linearly separable datasets for similar reasons as K-means, which is in fact the method used in the final step of the algorithm, but also because it is very difficult to capture the convoluted shapes of clusters in complex datasets using pre-defined grids.

Strengths and weaknesses. SOMs produce a 2-D discretised representation of the data, by imposing the structure and iteratively adjusting the map. Neighbouring nodes are then assigned to the same cluster via standard clustering methods such as K-means.

The use of SOM clustering has helped to identify novel therapeutic targets for the inhibition of the replication of the human immunodeficiency (HIV-1) virus, which is now crucial since the viral resistance develops in a large proportion patients undergoing highly active antiretroviral therapy (HAART) [Liang et al., 2005].

SOM clustering is suitable for large datasets and it is a useful visualisation tool but it presents an important limitation known as the 'boundary effect', which affects the nodes on the edge of the map and is responsible for retaining influence of the random initialisation of weights resulting in an inappropriate representation of the topology, and hence poor clustering results [Kiang et al., 1997; Wang et al., 2011]. Additionally, SOMs require the user to specify the grid topology as well as the number of clusters [Zhao et al., 2004], and as we showed above, SOMs were not able to identify the test datasets with clusters that were not globular.

Other important drawbacks associated with SOM clustering are: its accuracy depends on the number of iterations, the maps are not guaranteed to preserve their topology and it can over-represent regions of lower density while at the same time can under-represent regions of higher densities [Su et al., 2002].

2.1.2 Fuzzy clustering algorithms

2.1.2.1 Fuzzy C-means clustering

Fuzzy C-means (FCM) is a soft (fuzzy) clustering method that aims to minimise a weighted within-cluster sum of squares. As a soft method, it allows one data point to belong to more than one cluster, in contrast to crisp clustering methods such as K-means and spectral clustering. This technique was developed by Dunn in 1973 and improved by Bezdek et al. in 1983 and it is commonly used in pattern recognition.

Given the dataset $\mathbf{A} = \{\mathbf{x}_1, \mathbf{x}_2, \dots \mathbf{x}_N\}$ where $\mathbf{x}_i \in \mathbb{R}^d$, $i = 1, 2, \dots N$ are the data points to be clustered, and K the number of clusters, the goal of FCM is to find for each data point $\mathbf{x}_i$ a degree of membership $u_{i,k}$ in every cluster, $k = 1, 2, \dots K$.

The algorithm for FCM clustering, given a parameter $q > 1$ and the number of clusters, K, is as follows:

1. Assign, randomly, membership degrees to each data point for the K clusters ensuring they sum to 1, i.e., $\sum_{k=1}^{K} u_{i,k} = 1, i = 1, 2, \dots N$.
2. Compute the centroid $\widetilde{\mathbf{c}}_k$ of each cluster ($k = 1, 2, \dots K$) using the formula below.

$$\widetilde{\mathbf{c}}_k = \frac{\sum_{i=1}^{N} u_{i,k}^q \mathbf{x}_i}{\sum_{i=1}^{N} u_{i,k}^q}. \tag{2.16}$$

3. Compute the membership degree for each data point in each cluster using the following equation (see Appendix A.1 for how to derive this equation).

$$u_{i,k} = \frac{1}{\sum_{l=1}^{K} \left(\frac{||\mathbf{x}_i - \widetilde{\mathbf{c}}_k||}{||\mathbf{x}_i - \widetilde{\mathbf{c}}_l||} \right)^{\frac{2}{q-1}}}. \tag{2.17}$$

4. Repeat steps 2 and 3 until convergence to a stationary point[3] for the centroids $\widetilde{\mathbf{c}}_k$ and the degrees of membership $u_{i,k}$, $i = 1,2,\dots N$, $k = 1,2,\dots K$.

This method is based on the minimisation of an objective function J, where the membership degrees are raised to the power of a real number, q, greater than 1, widely known as the "fuzziness parameter" or "weighting exponent" [Bezdek et al., 1983]:

$$\mathrm{J} = \sum_{i=1}^{N} \mathrm{J}_i = \sum_{i=1}^{N} \sum_{k=1}^{K} u_{i,k}^{q} \|\mathbf{x}_i - \widetilde{\mathbf{c}}_k\|^2, \tag{2.18}$$

subject to the restriction known as the "probabilistic constraint of FCM" [Krishnapuram and Keller, 1993] requiring that the sum of the membership degrees is 1:

$$\sum_{k=1}^{K} u_{i,k} = 1, \quad i = 1,2,\dots N. \tag{2.19}$$

The solution of this minimisation problem is obtained using the method of Lagrange multipliers. The details are shown in the Appendix, Section A.1. After applying this technique we obtain the expressions for $\widetilde{\mathbf{c}}_k$ and $u_{i,k}$ in Eqs. (2.16) and (2.17), respectively.

For q close to 1, the cluster centre nearest to the point is given a higher weight than the other clusters, and the results are very similar to those of K-means clustering. In contrast, for very large values of q, the clusters tend to become indistinguishable from one another. In the limit $q \to \infty$, each data point has the same membership value in every cluster $u_{i,k} = \frac{1}{K}$, $i = 1,2,\dots N$, $k = 1,2,\dots K$, and all cluster centroids coincide with the centroid of the whole dataset $\widetilde{\mathbf{c}}_k = \frac{1}{N} \sum_{i=1}^{N} \mathbf{x}_i$.

FCM allows the identification of data points that are in the intersections of clusters and gives more insight into the cluster structure than hard techniques. It is particularly useful when working with genetic profiling data, since genes can be associated with more than one cluster of genes at the same time.

[3] The FCM algorithm always converges to a local minimum or at worst to a saddle point [Hathaway and Bezdek, 1988].

Fig. 2.22 shows the clustering result for the *Globular clusters* dataset, with $K = 2$, 3 and 4, where we see a good definition of the clusters. We used $q = 2$, which is the most common value for this parameter, and which indeed gives satisfactory results for this dataset.

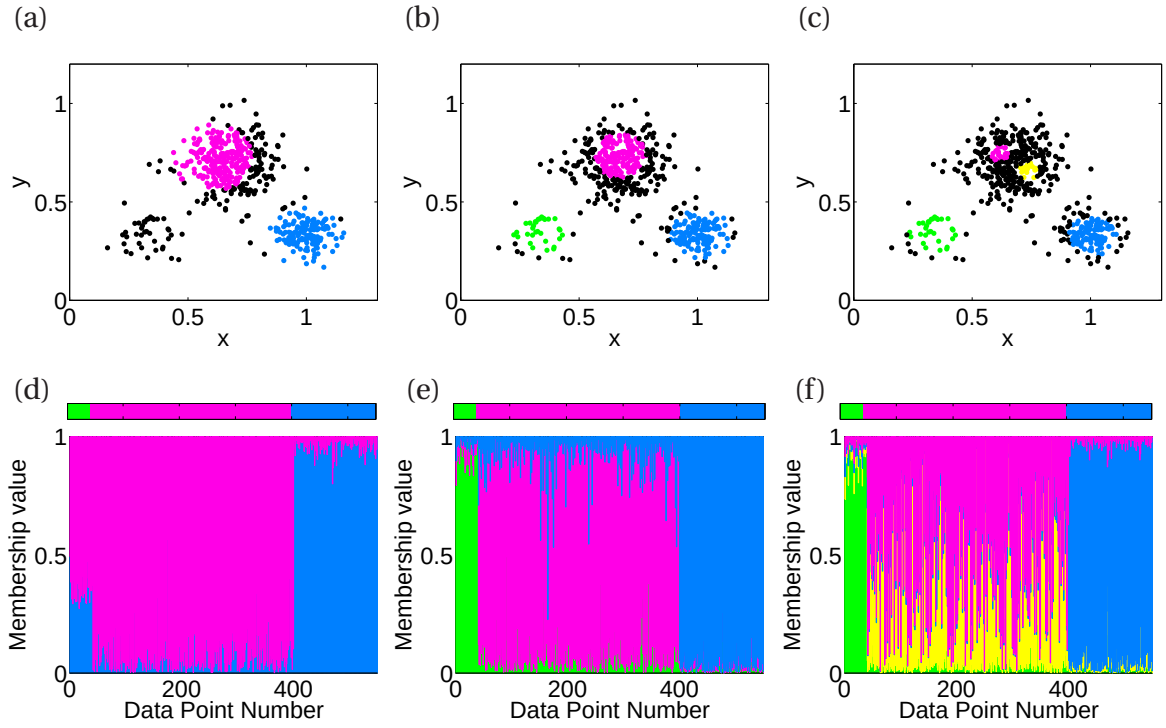


Figure 2.22: FCM applied to the *Globular clusters* dataset (Fig. 2.1 (a)), $q = 2$. (a) HCT-plot, $z = 0.9$, $K = 2$, (b) HCT-plot, $z = 0.9$, $K = 3$, (c) HCT-plot, $z = 0.9$, $K = 4$, (d) membership values, $K = 2$, (e) membership values, $K = 3$ and (f) membership values, $K = 4$.

Visualising the output of soft methods is more challenging than for the previously considered hard methods. An option is to plot the membership values using stacked colourcoded vertical bars as shown in Fig. 2.22 (d)–(f), which we call MV plots. Alternatively, we can visualise the result of the algorithm through an HCT-plot ("Hard Clusters by Threshold") which we define as follows: a data point is assigned to a cluster only if its membership value for that cluster is higher than a chosen threshold z, assuming that there is at most one cluster for every data point that fulfills this criteria. All the other data points are unassigned, and consequently plotted in black. Such a plot is shown in Fig. 2.22 (a) for $z = 0.9$.

An alternative to an HCT-plot is a contingency table (also known as confusion matrix),

which shows the number of data points from each class assigned to different clusters, as shown later in Section 2.2.4.2, Table 2.2. Neither contingency tables nor HCT-plots show the complete result of applying a given soft clustering method to a dataset, since they contain less information than the complete result (all the membership values), and depend on a threshold.

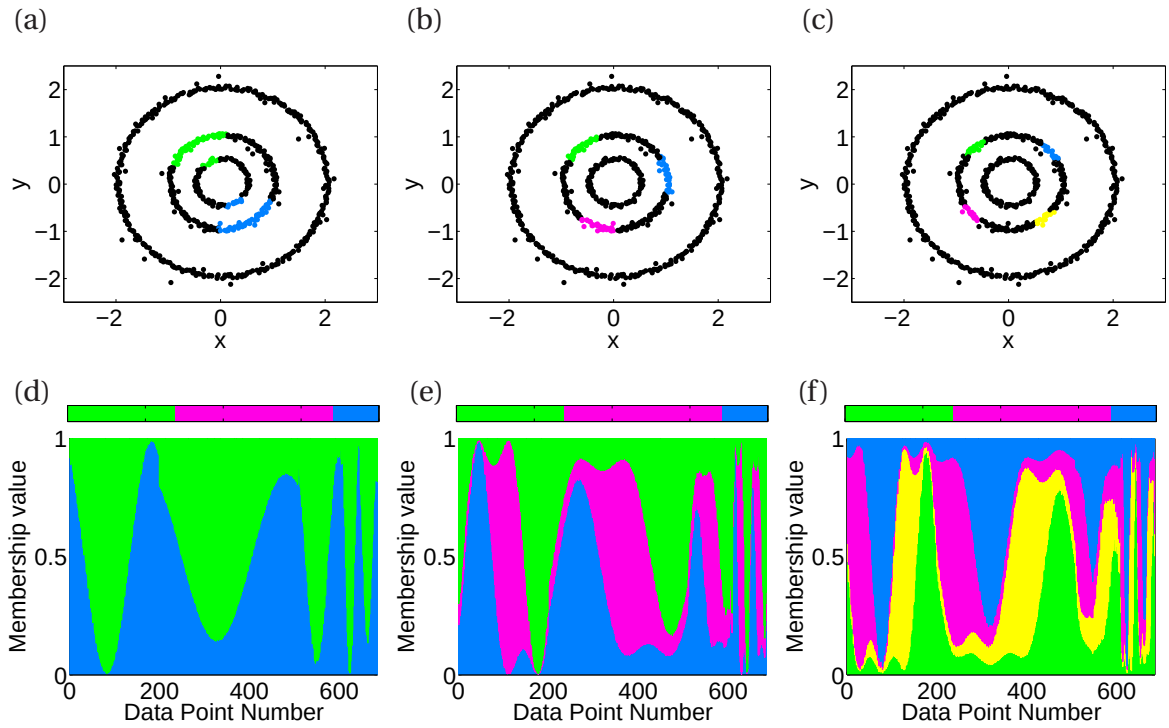


Figure 2.23: FCM applied to the *Concentric rings* dataset (Fig. 2.1 (b)), $q = 2$. (a) HCT-plot, $z = 0.9$, $K = 2$, (b) HCT-plot, $z = 0.9$, $K = 3$, (c) HCT-plot, $z = 0.9$, $K = 4$, (d) membership values, $K = 2$, (e) membership values, $K = 3$, and (f) membership values, $K = 4$.

A clear example of a dataset where FCM fails is the *Concentric rings* dataset, where the clusters identified belong to the same ring (Fig. 2.23). Using smaller, less stringent thresholds, we would see that the clusters form a pie-chart structure, like K-means and SOMs in Figs. 2.3 and 2.20, allocating data points from different rings to the same clusters. Another example where FCM fails is the *Elongated clusters* dataset (results shown in Fig. 2.24). We later repeat these results in Figs. 4.13, 4.14 and 4.15 to compare FCM with the novel clustering method we develop.

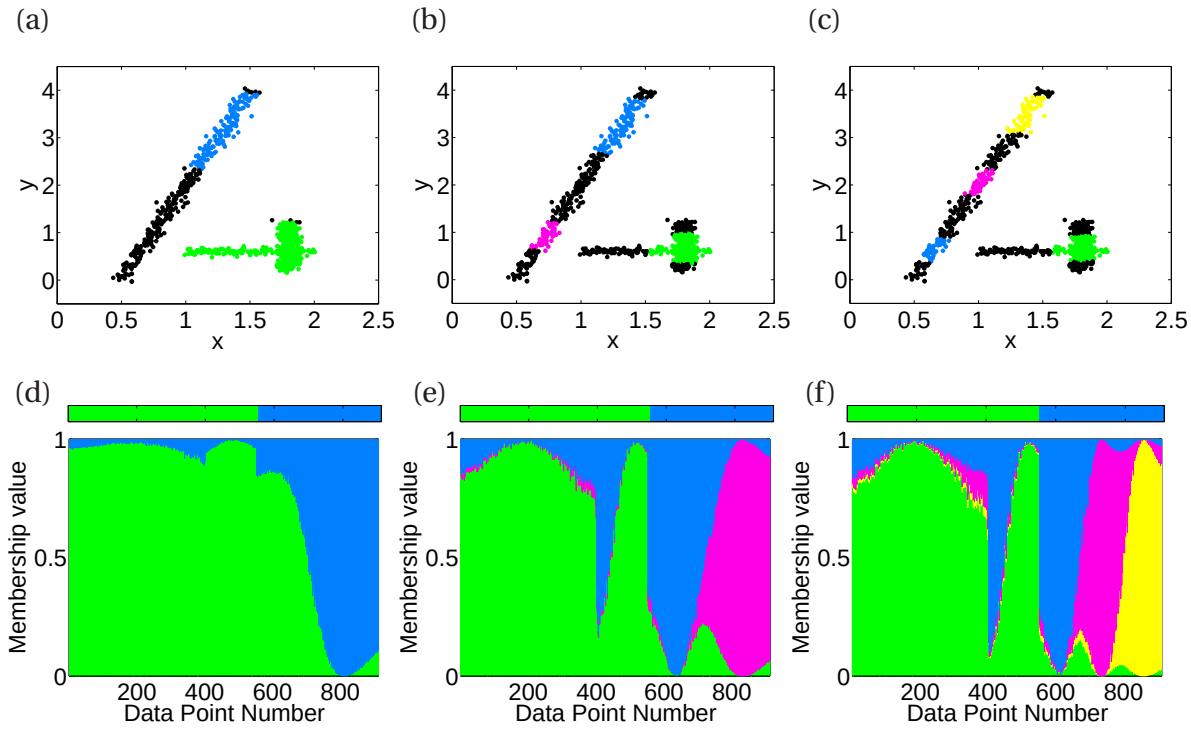


Figure 2.24: FCM applied to the *Elongated clusters* dataset (Fig. 2.1 (b)), $q = 2$. (a) HCT-plot, $z = 0.9$, $K = 2$, (b) HCT-plot, $z = 0.9$, $K = 3$, (c) HCT-plot, $z = 0.9$, $K = 4$, (d) membership values, $K = 2$, (e) membership values, $K = 3$ and (f) membership values, $K = 4$.

Strengths and weaknesses. FCM has been successful in a number of applications; one in particular is in medical image segmentation, where it is used as an aid in clinical diagnosis of brain tumour MRI image scans [Gopal and Karnan, 2010].

As we saw above, FCM's shortcomings are similar to those of K-means: both work best for globular, compact, well-separated clusters that are of roughly the same density and size but cannot identify concentric or elongated clusters. Additionally, there are no clear guidelines as to how the values of the parameter q should be chosen, and the number of clusters has to be given as an input [Tan et al., 2006].

Since the membership values assigned by FCM are inversely proportional to the relative distance of the point to the cluster centroids [Pal et al., 1997], a data point that is very far away from both clusters would be assigned an intermediate membership value (near 0.5) to both clusters, which would also be the case if that point lay in the middle of the line joining the centroids of both clusters. Hence, FCM is unable to identify outliers.

2.1.2.2 Possibilistic C-means clustering

The inability of identifying outliers using FCM is addressed by the Possibilistic C-means (PCM) clustering method. This technique was developed by Krishnapuram and Keller [1993] and is a modified version of FCM. PCM drops the probabilistic constraint of FCM (Eq. (2.19)), i.e. it does not require that the membership values of a data point in all clusters add up to 1. This way, a noisy data point very distant (in Euclidean terms) from all the clusters will be assigned a low membership value in all clusters, giving a good indication that the point does not belong to the natural clusters in the data, and is likely an outlier.

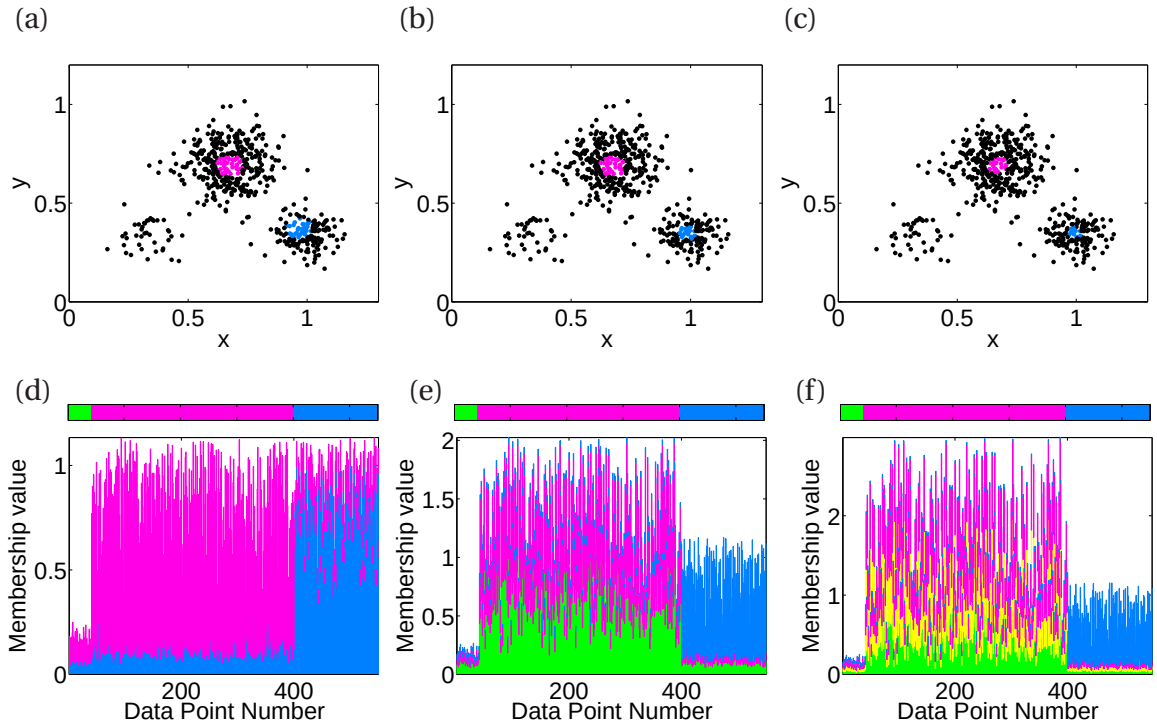


Figure 2.25: PCM applied to the *Globular clusters* dataset (Fig. 2.1 (a)), $q = 2$. (a) HCT-plot, $z = 0.9$, $K = 2$, (b) HCT-plot, $z = 0.9$, $K = 3$, (c) HCT-plot, $z = 0.9$, $K = 4$, (d) membership values, $K = 2$, (e) membership values, $K = 3$ and (f) membership values, $K = 4$.

However, PCM is highly sensitive to the initialisation of the centroids and it can produce coincident clusters [Barni et al., 1996], as can be seen in Fig. 2.25 (b)-(c), where PCM does not separate the data points into three disjoint clusters, but rather gives high membership values for data points in the large middle cluster – underperforming, for this example, com-

pared to all the previously mentioned clustering methods (namely K-means, hierarchical clustering, spectral clustering and FCM).

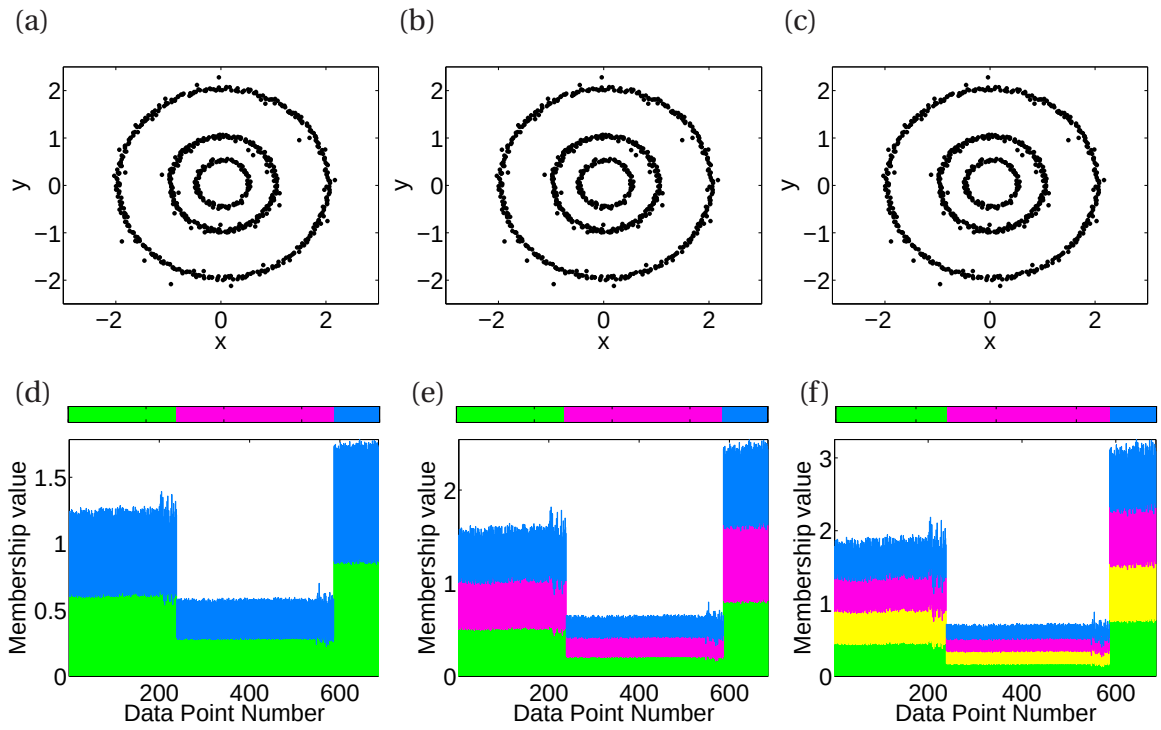


Figure 2.26: PCM applied to the *Concentric rings* dataset (Fig. 2.1 (b)), $q = 2$. (a) HCT-plot, $z = 0.9$, $K = 2$, (b) HCT-plot, $z = 0.9$, $K = 3$, (c) HCT-plot, $z = 0.9$, $K = 4$, (d) membership values, $K = 2$, (e) membership values, $K = 3$ and (f) membership values, $K = 4$.

For the *Concentric rings* dataset both Figs. 2.26 and 2.27 show that PCM assigns every data point to every cluster with similar membership values, and none of the data point has a membership value higher than the threshold ($z = 0.9$), hence the black colours in the corresponding HCT plots (a)-(c). It is clear that there are three distinct clusters, even when using $K = 2$ and $K = 4$ (Fig. 2.26 (d)-(f)), from the heights of the membership value bars (notice that for PCM, unlike FCM, these bars do not necessarily add up to 1), but the algorithm cannot correctly assign the data points to their respective clusters.

Analogously, the MV plots in Figs. 2.27 (d)-(f) present several peaks. The two peaks in the left side of the MV plots represent, respectively, the vertical and the horizontal groups of points, while the smaller peak corresponds to the diagonal cluster.

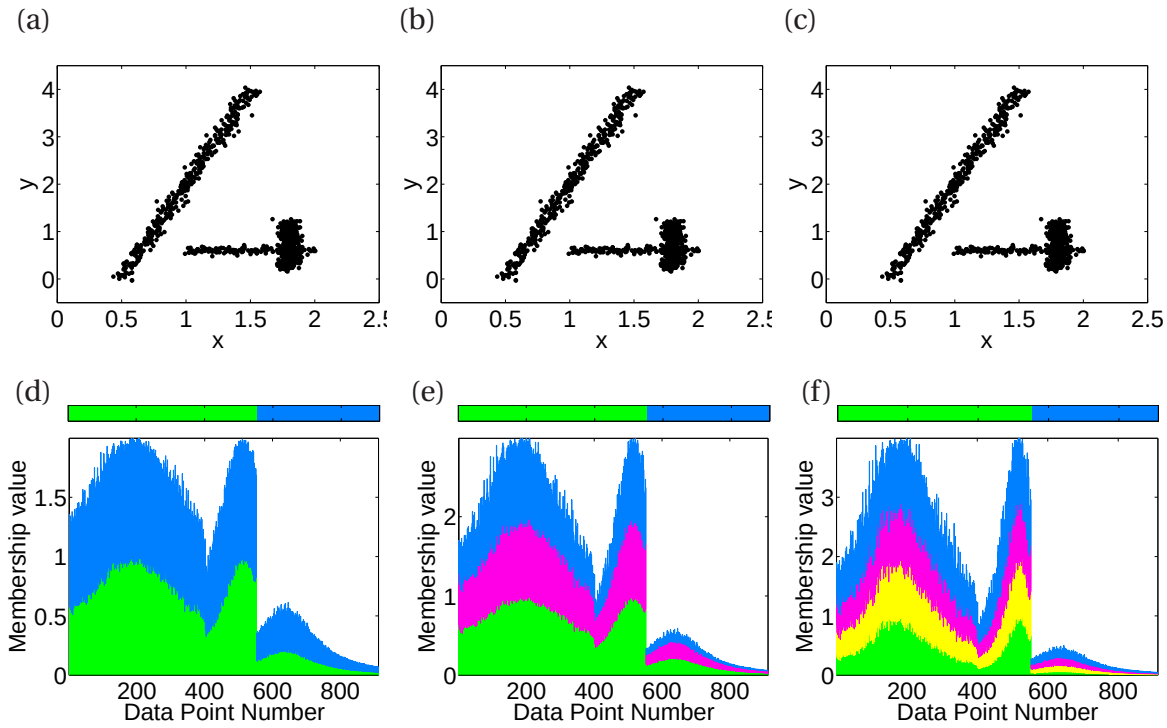


Figure 2.27: PCM applied to the *Elongated clusters* dataset (Fig. 2.1 (b)), $q = 2$. (a) HCT-plot, $z = 0.9$, $K = 2$, (b) HCT-plot, $z = 0.9$, $K = 3$, (c) HCT-plot, $z = 0.9$, $K = 4$, (d) membership values, $K = 2$, (e) membership values, $K = 3$ and (f) membership values, $K = 4$.

Strengths and weaknesses. PCM solves an important shortcoming of FCM by being less susceptible to outliers, but in turn presents even more substantial weaknesses, such as being highly sensitive to initialisation and yielding coincident clusters [Kim et al., 2005].

Using synthetic test datasets we have shown that PCM underperforms in comparison to all the previous algorithms, by not identifying the globular clusters and not defining cluster assignments in concentric rings and elongated clusters.

As well as failing to identify non-linearly separable clusters, PCM failed dramatically in the synthetic test datasets due to the poor initialisation of the centroids. A good solution to improve them is to carry out PCM after identifying the cluster centres using another clustering algorithm, but this can be computationally intensive.

PCM has been successfully applied to identifying proteins which had been misannotated in the Gene Ontology [Pal et al., 2005]. A number of algorithms exist that improve upon PCM by targeting one or more of its defects [Saad and Alimi, 2009].

2.1.3 Summary of clustering methods

In Table 2.1 we summarise the clustering methods described above, their characteristics and the parameters and options that they require.

The first data clustering methodology we described was the K-means algorithm. Introduced by MacQueen in 1967, it is the prototypical example of a non-overlapping, hard (crisp) clustering approach [Gan et al., 2007]. The applicability of the K-means algorithm, however, is limited by the requirement that clusters should be well-separated and "convex-shaped" in order to be properly identified (such as those in Fig. 2.1 (a)), which is not the case in most biological data. We considered two fundamentally distinct approaches that have been proposed in the past to address these two restrictions (fuzzy methods in Section 2.1.2.1 and spectral methods in Section 2.1.1.3).

	Hard or soft	Flat or hierarchical	Classification category	Parameters or options
K-means	hard	flat	compactness	K
HC	hard	hierarchical	connectedness	linkage
Spectral	hard	flat	connectedness	β, K, n
SOM	hard	flat	compactness	$K, \alpha(t), \sigma_t, t_{\max}, \theta$, grid
FCM	soft	flat	compactness	q, K
PCM	soft	flat	compactness	q, K

Table 2.1: Summary of categories and parameters of reviewed clustering methods. The classification category of compactness includes clustering techniques which minimise the intra-cluster variation, while connectedness comprises clustering methods that ensure that neighbours are clustered together [Handl et al., 2005].

The FCM algorithm was proposed as an alternative soft clustering approach that generates soft partitions for a given dataset. In the case of FCM, the clusters to be identified do not have to be well-separated, as the method assigns cluster membership probabilities to ambiguous elements of the dataset that cannot be readily assigned to one specific cluster.

However, the method does not exploit the intrinsic geometry of non-convex clusters, and, as we show in Section 2.1.2.1, its performance is drastically reduced when applied to some datasets, for example those in Figs. 2.1 (b) and 2.1 (c). This behaviour can also be observed in the case of the standard K-means algorithm [Ng et al., 2002]. These algorithms have been very successful in a number of diverse areas, such as image segmentation [Trivedi and Bezdek, 1986], analysis of genetic networks [Stuart et al., 2003], protein class prediction [Zhang et al., 1995], epidemiology [French et al., 2008], but here (in Section 2.1.2.1) we also explore datasets for which their performance is poor.

More recently, approaches based on spectral graph theory have been devised to circumvent the computational problems arising with datasets that have a manifold structure geometry. By this we mean a collection of points with a structure that resembles a Euclidean space (of a certain dimension) in the vicinity of each point, and such a structure can be locally mapped (with its neighbourhood) into $\mathbb{R}^{\eta}$ where η is the dimension of the manifold. However, these algorithms are generally hard clustering methods, which do not allow data points to belong to more than one cluster. This limits their applicability in clustering genetic expression data, where alternative or superimposed regulatory patterns are to be identified [Dembélé and Kastner, 2003].

From testing the clustering methods over the three chosen synthetic test datasets we learnt that although they are among the most well used clustering methods, they can fail if the datasets are non-linearly separable or are of dissimilar density/size. This can be overcome by using a method that does not rely on centroids as the representation of the clusters and that uses information on the local neighbourhood of data points. A clustering technique that uses concepts of diffusion distances, whose alternative interpretation can be random walks over graphs, could address the issues observed in other methods in this chapter.

There is a vast array of alternative clustering methods to the ones presented above, yet it is not within the scope of this thesis to review them exhaustively. We chose not to include clustering methods less widely used that make implicit assumptions on the underlying dis-

tribution of the data such as Gaussian mixtures models (GMM), which are limited to finding clusters of a special shape and would therefore fail to identify complex clusters such as the ones in Fig. 2.1 [Zhang et al., 2007].

Additionally, we did not discuss a number of density-based clustering algorithms (DBSCAN, DBRS, DENCLUE, etc.) that can identify clusters of arbitrary shapes but present major disadvantages when clustering complex systems biology data; such as being unable to identify clusters of different dispersion, having problems when clustering binary, categorical and mixed-type datasets, being sensitive to input parameters as well as showing limitations when clustering high-dimensional data and crucially, in their vast majority, being hard clustering methods [Jiang et al., 2003; Houle, 2008; Jain et al., 2008].

2.2 Cluster validation

In order to interpret and compare clustering results generated by different algorithms, and to explore a range of parameter space, we need to be able to assess each partition in a *quantitative* way. To do this we first need to select an appropriate similarity function between the data points.

2.2.1 Similarity

Similarity is a distance metric, indicating how similar objects are to each other [Schaeffer, 2007]. The similarity is always non-negative, with small values indicating that observations are far away together (and therefore alike), and large values (often scaled to have a maximum of 1) indicating very similar data points. The similarity between any data point and itself is the maximum, often 1.

There are several ways of defining similarity measures, the Euclidean distance being the most commonly used one [Filippone et al., 2008]. Others include the Manhattan (or L_1)

distance and various correlation distances [Schaeffer, 2007].

The similarity based on the Euclidean distance is computed as

$$\mathrm{Sim}^{\mathrm{Eucl}}_{\mathbf{x}_i,\mathbf{x}_j} = \frac{1}{1+\delta^{\mathrm{Eucl}}_{\mathbf{x}_i,\mathbf{x}_j}}, \tag{2.20}$$

where $\delta^{\mathrm{Eucl}}_{\mathbf{x}_i,\mathbf{x}_j}$ between observations $\mathbf{x}_i$ and $\mathbf{x}_j$ is given by

$$\delta^{\mathrm{Eucl}}_{\mathbf{x}_i,\mathbf{x}_j} = \sqrt{\sum_{l=1}^{d}\left(x_{il}-x_{jl}\right)^2}, \tag{2.21}$$

where x_{il} is the l^{th} element of the i^{th} observation $\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{id})$, x_{jl} is the l^{th} element of the j^{th} observation $\mathbf{x}_j = \left(x_{j1}, x_{j2}, \ldots, x_{jd}\right)$, and d is the dimension of the data points $\mathbf{x}_i$ and $\mathbf{x}_j$.

The process of assessing the results of a clustering algorithm is known as *cluster validation*. The main problem faced when evaluating the goodness of the partition when fitting the underlying data is to decide the optimal number of clusters to be employed, such as K in the case of K-means clustering. This is usually not an issue when analysing two-dimensional data sets, in which case it is possible to visually verify the validity of the clusters found by the algorithm. In higher dimensional datasets, this visual validation becomes very difficult [Rousseeuw, 1987].

Clustering assessment relies on the choice of an appropriate validity index; depending on what a given clustering algorithm optimises, some validity indices will be better suited than others to assess performance. We will now consider different validity indices for the two main types of clustering algorithms: *hard* and *soft* clustering.

2.2.2 Hard clustering

2.2.2.1 Silhouette width

Silhouette width (SW) is a type of cluster validity measure or a confidence indicator on the membership of the observations in each cluster based on the comparison of their tightness and separation. It is often used to help determine how many clusters are present in a given dataset.

Let us define the data points as the vectors $\mathbf{x}_i \in \mathbb{R}^d$, $i = 1,2,\dots N$. The silhouette widths, SW_i, are calculated for each observation $\mathbf{x}_i$, $i = 1\dots N$ as follows:

$$\mathrm{SW}_i = \frac{(b_i - a_i)}{\max\{a_i, b_i\}}, \tag{2.22}$$

where a_i corresponds to the within distance: the average distance between the data point $\mathbf{x}_i$ and the other elements in the cluster to which $\mathbf{x}_i$ is assigned, and b_i corresponds to the minimum between distance: minimum distance between the data point $\mathbf{x}_i$ and all other data points that do not belong to $\mathbf{x}_i$'s cluster. From Eq. (2.22) we see that $-1 \leq \mathrm{SW}_i \leq 1$. For a data point $\mathbf{x}_i$ exhibiting a within similarity a_i much smaller than the minimum between distance b_i, its silhouette width is close to 1, and it suggests that $\mathbf{x}_i$ is very likely to belong to its assigned cluster. If a_i and b_i are very similar, then the data point could belong to either cluster, the one to which it has been assigned or the one with the minimum between similarity. On the contrary, if the within similarity of $\mathbf{x}_i$ is much larger than its minimum between distances, SW_i is close to -1 and in this case the data point is likely to have been misassigned [Jain et al., 2008].

The average silhouette width ($\overline{\mathrm{SW}}$) for all the data points gives a single quantitative value of the quality of the clustering result, and can be compared between different K values, other parameters or clustering results:

$$\overline{\mathrm{SW}} = \frac{1}{N}\sum_{i=1}^{N} \mathrm{SW}_i. \tag{2.23}$$

According to Kaufman and Rousseeuw [1990], an average silhouette ($\overline{\text{SW}}$) width greater than 0.5 indicates a reasonable partition of the data (and greater than 0.8 a good or very good data clustering), and $\overline{\text{SW}} < 0.2$ would imply that the dataset being explored does not exhibit cluster structure [Martínez and Martínez, 2004].

Silhouette widths receive their name from the shape of their horizontal bar plots (Fig. 2.28), where the SW_i values are grouped according to their cluster assignment and each cluster is sorted such that higher SW_i values are plotted higher in the y-axis, forming silhouettes like the ones in Fig. 2.28.

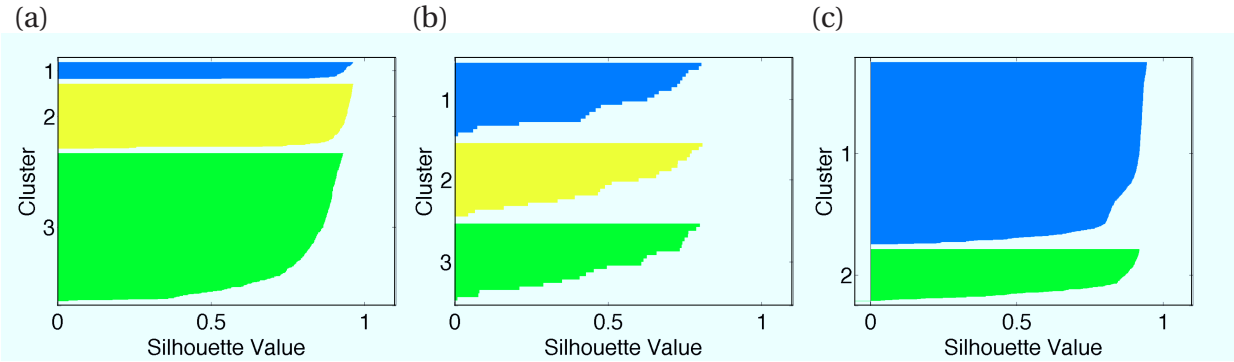


Figure 2.28: Silhouette widths of K-means on the test datasets from Fig. 2.1. (a) *Globular clusters* dataset, $K = 3$, results and colour code as in Fig. 2.2 (b). $\overline{\text{SW}} = 0.838$. (a) *Concentric rings* dataset, $K = 3$, results and colour code as in Fig. 2.3 (b). $\overline{\text{SW}} = 0.523$. (c) *Elongated clusters* dataset, $K = 2$, results and colour code as in Fig. 2.4 (a). $\overline{\text{SW}} = 0.842$.

From Fig. 2.28 we can deduce that K-means gives reasonable-to-good partitionings of the three test datasets. However, we saw in Section 2.1.1.1, that neither the *Concentric rings* nor the *Elongated clusters* datasets were correctly clustered. This is a result of using Euclidean distances in both the K-means method and the $\overline{\text{SW}}$ quality measure over datasets with complex (non-linearly separable) cluster geometry. In order to overcome this we will introduce in Section 3.5.1 a new silhouette width type of measure which uses diffusion distances between the data points.

2.2.2.2 Rand index (RI)

Another important index that measures the degree of similarity between two partitionings is the Rand index [Rand, 1971], defined as,

$$\mathrm{RI}(\mathbf{C}^{(1)},\mathbf{C}^{(2)}) = \frac{a+d}{a+b+c+d} \tag{2.24}$$

where a is the number of pairs of points that belong to the same cluster in partitionings $\mathbf{C}^{(1)}$ and $\mathbf{C}^{(2)}$, b denotes the points that belong to the same cluster in partitioning $\mathbf{C}^{(1)}$ but not in $\mathbf{C}^{(2)}$, while c corresponds to the pairs of points that belong to different clusters in partitioning $\mathbf{C}^{(1)}$ but in the same cluster in $\mathbf{C}^{(2)}$ and finally, d is defined as the number of pairs of points that are neither co-clustered in $\mathbf{C}^{(1)}$ nor in $\mathbf{C}^{(2)}$.

Morey and Agresti [1984] developed an alternative Rand method that allows for a *randomness* treatment, namely, they suggested a correction for the chance term, extensively discussed by Hubert and Arabie [1985]. We use RI to compare partitions in Section 5.3.1.

2.2.3 Fuzzy clustering

2.2.3.1 Fuzzy silhouette width

The fuzzy silhouette width (FSW) is an adaptation of hard silhouette widths for soft clustering results. It generalises the SW method by introducing a weighted average of the silhouette widths which gives more importance to the crisper points than to those that present a higher degree of fuzziness [Campello and Hruschka, 2006].

The FSW of a partitioning is computed according to the following formula:

$$\mathrm{FSW} = \frac{\sum_{i=1}^{N} \left(u_{i,a} - u_{i,b}\right)^{\theta} \mathrm{SW}_i}{\sum_{i=1}^{N} \left(u_{i,a} - u_{i,b}\right)^{\theta}}, \tag{2.25}$$

where $u_{i,a}$ and $u_{i,b}$ are the first and second highest membership values, respectively, of data point $\mathbf{x}_i$ in all clusters (a being the cluster closer to $\mathbf{x}_i$) and SW_i is the hard silhouette width defined by Eq. (2.22).

The parameter $\theta \geq 0$, in analogy with the "fuzziness parameter" q of Fuzzy C-means clustering, Eq. (2.18), modifies the weights with which the silhouette widths contribute to the overall clustering measure. A commonly used value for θ is 1 [Campello and Hruschka, 2006]. In Section 3.5.1 we will compare FSW with DSW and a new silhouette width type of measure.

The quality measures explained above are internal indices because they use intrinsic information of the dataset to assess the quality of the partitioning [Handl et al., 2005]. In the following section we describe external indices that require external information such as the true labels.

2.2.4 Clustering evaluation when true clusters are known

Clustering evaluation measures that are derived by comparing a partitioning to a set of reference clusters, such as classification error and the area under the receiver operating characteristic (ROC) curve, are also known as external measures [Handl et al., 2005; Tuikkala et al., 2008].

We know the true clustering of data if we have generated the synthetic data ourselves, such as the *Globular clusters*, *Concentric rings* and *Elongated clusters* datasets in Fig. 2.1, or if the data are sampled from known distinct groups, such as the *Iris* dataset in Section 4.1.1.2, or even if it is dictated by an expert, such as in the classification of malignant versus benign tumour cells in *The Wisconsin Diagnostic Breast Cancer* (*WDBC*) dataset in Section 4.1.1.5. In such cases the following quality measures can be used for evaluating the performance of a clustering algorithm.

2.2.4.1 Classification error

For hard clustering methods, classification error provides an intuitive performance indicator. It is simply the fraction of data points incorrectly classified:

$$\text{Classification error}\,(\%) = \frac{\text{number of data points incorrectly clustered}}{N} \times 100 \tag{2.26}$$

where N denotes the total number of data points. In order to obtain the number of data points that have been correctly clustered we first have to pair the identified clusters to the reference clusters in a way that minimises the total number of incorrect cluster assignments. A matrix of costs (**MC**), with the following entries, is first computed:

$$\mathbf{MC}_{k,l} = \text{number of data points from cluster } \widetilde{C}_k \text{ which do not belong to } C_l \tag{2.27}$$

where the partitioning $\mathbf{C} = \{C_k | k = 1,2,\ldots K\}$ is the one being assessed and $\widetilde{\mathbf{C}} = \{\widetilde{C}_k | k = 1,2,\ldots \widetilde{K}\}$ is the reference clustering of the dataset. The objective is then to find a minimum cost assignment where the pairing of cluster k to cluster l incurs a cost of $\text{MC}_{k,l}$, or in the clustering case, a matching which minimises the number of misclassifications.

The Hungarian method (also known as the 'Munkres' assignment algorithm, Kuhn [1955]; Topchy et al. [2003]) efficiently solves this minimum cost assignment problem. The details of the algorithm are beyond the scope of this thesis, but it is worth mentioning that it can also be described in terms of bipartite graphs, where the costs correspond to edge lengths [Fukuda and Matsui, 1989].

This comparison index is applicable only to partitionings with the same number of clusters as in the reference partitioning (i.e. $K = \widetilde{K}$, making the matrix **MC** square: just one cluster can be matched to an original label at one time).

We will use classification errors in Section 4.1.2 to compare the performance of the different clustering methods discussed, both soft and hard, over a range of benchmark datasets.

Another summary table often used when comparing classification results is the confusion matrix or contingency table, which differs from **MC** in that instead of the number of correctly classified data points it contains its complement, i.e. the number of data points which were misclassified (see Table 2.2).

2.2.4.2 ROC curves

If the true grouping of the data is known, a useful way to evaluate the performance of a clustering technique is a receiver operating characteristic (ROC) curve which, in data classification, plots the true positive rate (TPR) versus the false positive rate (FPR) [Fawcett, 2006]. Each data point in the curve represents a pair of values (FPR, TPR). A perfect clustering method would give a curve that passes through the upper left corner of the ROC graph, corresponding to a true positive rate of 100% and a false positive rate of 0%, whereas with a random classifier, true positive rates coincide with false positive rates, resulting in the diagonal TPR=FPR.

ROC curves were originally designed to compare only two classes, and one approach to deal with multiple classes is called the class reference formulation [Fawcett, 2004], which compares one cluster against the rest of the data at a time by considering the data that do not belong to the cluster of interest C_k as one large cluster C_k^c.

		Clusters identified	
		C_k	C_k^c
True clusters	$\widetilde{C}_{h(k)}$	True positives	False negatives
	$\widetilde{C}_{h(k)}^c$	False positives	True negatives

Table 2.2: Confusion matrix

where C_k^c and $\widetilde{C}_{h(k)}^c$ represent the complement of C_k and $\widetilde{C}_{h(k)}$, respectively. True positives are the data points in cluster C_k which are also in cluster $\widetilde{C}_{h(k)}$; true negatives are the objects that do not belong to cluster C_k or $\widetilde{C}_{h(k)}$; false positives are the data points assigned to

cluster C_k which should not have been, since they do not belong to cluster $\widetilde{C}_{h(k)}$; and finally false negatives are the objects that belong to cluster $\widetilde{C}_{h(k)}$ but have been misclassified.

In order to determine the TPR and FPR of clusters obtained using a hard clustering method, consider two partitionings, $\mathbf{C} = \{C_k, k = 1,2,\dots K\}$ and $\widetilde{\mathbf{C}} = \{\widetilde{C}_k, k = 1,2,\dots \widetilde{K}\}$, of a dataset $\mathbf{x}_1, \mathbf{x}_2, \dots \mathbf{x}_N \in \mathbb{R}^d$. Let us assume partitioning $\widetilde{\mathbf{C}}$ is the true clustering of the dataset, while $\mathbf{C}$ is the partitioning to be assessed. For each cluster $C_k \in \mathbf{C}$ we identify its most similar cluster in the true partitioning, $\widetilde{\mathbf{C}}$, label it by the index $h(k)$ and build the confusion matrix shown in Table 2.2.

The TPR, also known as the hit rate [Fawcett, 2006], is the ratio of data points correctly clustered to the total number of points in the true cluster of reference (True positives + False negatives), whereas the FPR is the ratio of false positives to the total number of points which do not belong to the true cluster of reference:

$$\text{TPR} = \frac{\text{True positives}}{\text{True positives} + \text{False negatives}} \tag{2.28}$$

$$\text{FPR} = \frac{\text{False positives}}{\text{False positives} + \text{True negatives}} \tag{2.29}$$

The output of a hard clustering method can be characterised by a single value of TPR and FPR. On the other hand, if the algorithm to be assessed is not hard, we can obtain one (TPR, FPR) pair for every threshold z we use to make the results crisp such that the data points will be assigned to a cluster if their membership values are larger than the given threshold [Fawcett, 2006].

The accuracy of the classification is measured through ROC curves using the area under the curve (AUC). A perfect clustering result has AUC = 1, while the AUC of a random classification or one that gives no useful information regarding how to separate the classes is close to 0.5.

ROC curves will be used to compare soft clustering methods in Sections 4.2.5, 4.1.1.1, 4.1.1.2 and 4.1.1.3 and confusion matrices will be used in 5.3.1.

2.3 Discussion

In this chapter we presented a review of different clustering methods. We discussed flat/hierarchical, hard/fuzzy methods that are widely used in bioinformatics and systems biology.

In order to compare the different clustering techniques, we introduced three test datasets and described drawbacks of K-means and FCM clustering. These disadvantages are especially obvious when we do not know *a priori* how many clusters are in the data, what their shapes are, and how far away the clusters are from each other. The hierarchical clustering method, for which the number of clusters is not prespecified, also failed to identify clusters in the *Concentric rings* dataset, but gave a good result when clustering elongated clusters of data. On the other hand, the spectral clustering technique was successful in clustering the three test datasets, but being a hard clustering technique, did not allow data points to belong to more than one cluster.

We note that the random initialisation of centroids in the K-means and FCM algorithms means that they are nondeterministic, which is why it is recommended to run the algorithm a number of times to avoid suboptimal solutions.

We can see that the effectiveness of a clustering algorithm depends strongly on the characteristics of each dataset. The optimum clustering method choice depends heavily on the structure of the particular data. Hence, different clustering methods are better suited to specific classes of clustering problems.

We have also reviewed a number of frequently used clustering quality measures, both for hard and soft types of clustering techniques. An extensive array of quality measures and criteria can be used to compare results from different clustering algorithms or different parameter values explored by a given method.

In the next chapter we present a new general soft clustering method that overcomes difficulties that currently exist in soft clustering algorithms like FCM and PCM, such as their inability to identify clusters that are not separable in a linear fashion.

Chapter 3

DifFUZZY: A Novel Fuzzy Clustering Algorithm

In this chapter we present and discuss DifFUZZY, a new soft clustering algorithm that is built on concepts from diffusion on graphs and fuzzy clustering methods, and an *ad-hoc* quality measure to assess clustering results and to automatically determine the optimal value of parameters. This chapter covers material from Cominetti et al. [2010].

3.1 The algorithm

We define the input to the algorithm to be the list of N d-dimensional data points

$$\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N \in \mathbb{R}^d. \tag{3.1}$$

In addition, an external parameter M and three optional internal parameters, γ_1, γ_2 and γ_3 are required. M, an integer that represents the minimum number of data points per identified cluster, ensures no clusters of very small size are created as a few data points normally constitute a set of outliers. The three optional parameters, γ_1, γ_2 and γ_3, have optimal values 0.3, 0.1 and 1, respectively, which we found to give consistently good results for all of the datasets analysed. An intuitive explanation of the method and the significance of these parameters is provided in Section 3.1.4.

The output from DifFUZZY is the number of clusters found, K, and the membership vector for each data point, denoted by $u_{i,k}$ ($0 \leq u_{i,k} \leq 1$), which corresponds to the probability

that $\mathbf{x}_i$ is a member of cluster $\mathbf{C}_k$, and the membership degrees of any data point sum to 1, that is

$$\sum_{k=1}^{K} u_{i,k} = 1, \quad i = 1,2,\ldots,N. \tag{3.2}$$

DifFUZZY has been implemented in Matlab and C++ and can be downloaded from the following web page: `http://www.maths.ox.ac.uk/cmb/diffuzzy`. Details about its implementation are presented in the Appendix, Section B.4. The algorithm can be divided into three main steps, which are explained below in Sections 3.1.1-3.1.3.

3.1.1 Identification of the core of clusters

In order to identify the cluster cores we first determine the number of clusters, K, by optimising an internal parameter $\sigma \in (0,\infty)$.

We define an auxiliary function $F(\sigma) : (0,\infty) \to \mathbb{N}$ as follows. We construct the so-called σ-neighbourhood graph [van Leeuwen et al., 2007] where each node represents one data point from the dataset (3.1), i.e. the σ-neighbourhood graph has N nodes, as represented in Fig. 3.1. The i^{th} and j^{th} node will be connected by an edge if $\left|\left|\mathbf{x}_i - \mathbf{x}_j\right|\right| < \sigma$, where $||\cdot||$ represents the Euclidean norm and σ is the neighbourhood radious. $F(\sigma)$ is equal to the number of components of the σ-neighbourhood graph consisting of at least M vertices. In other words, $F(\sigma)$ represents the number of components of the σ-neighbourhood graph of order (size) larger than M. The value of σ^* corresponds to the maximizer of the function $F(\sigma)$ for a given dataset

Fig. 3.2 shows a schematic representation of $F(\sigma)$, obtained during the first step of DifFUZZY using 12 data points and M = 4. For very small σ values (Fig. 3.1 (a)), there are as many components as nodes, and thus there are no components of size at least 4. For intermediate σ values, as in Fig. 3.1 (b), there are two components with four or more vertices. Finally, for large σ values, all the vertices are connected, therefore there is only one component of size at least 4 (Fig. 3.1 (c)).

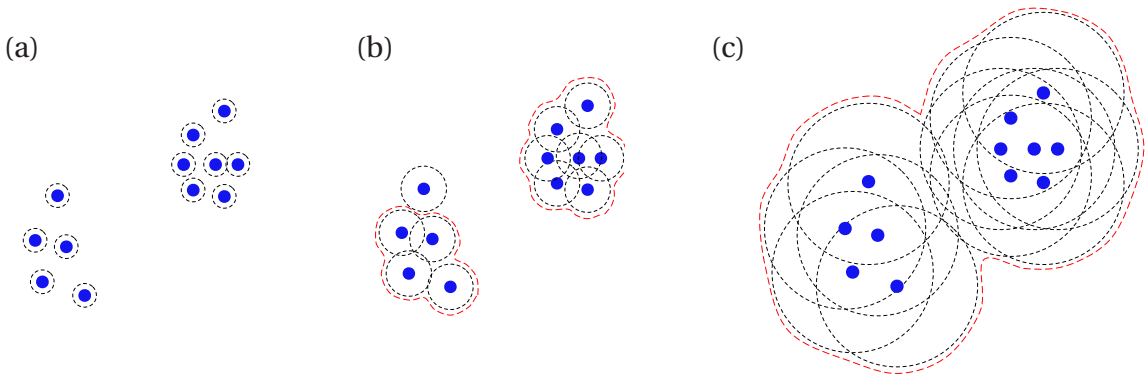


Figure 3.1: Schematic representation of the first step of DifFUZZY using 12 data points and M = 4. Neighbourhoods with σ values: (a) very small, (b) intermediate and (c) large. Dotted circles represent σ-neighbourhoods around each data point (dark dots). Neighbourhood clusters surrounded by a red dotted line correspond to the components of the σ-neighbourhood graph which contain at least M vertices. σ-neighbourhood graphs, also known as ϵ-neighbourhood graphs, are similarity graphs in which a pair of nodes are connected if their distance is less than a radious (σ or ϵ, respectively) and can be weighted, as it is the case for DifFUZZY's first step, but also unweighted. See text for more details.

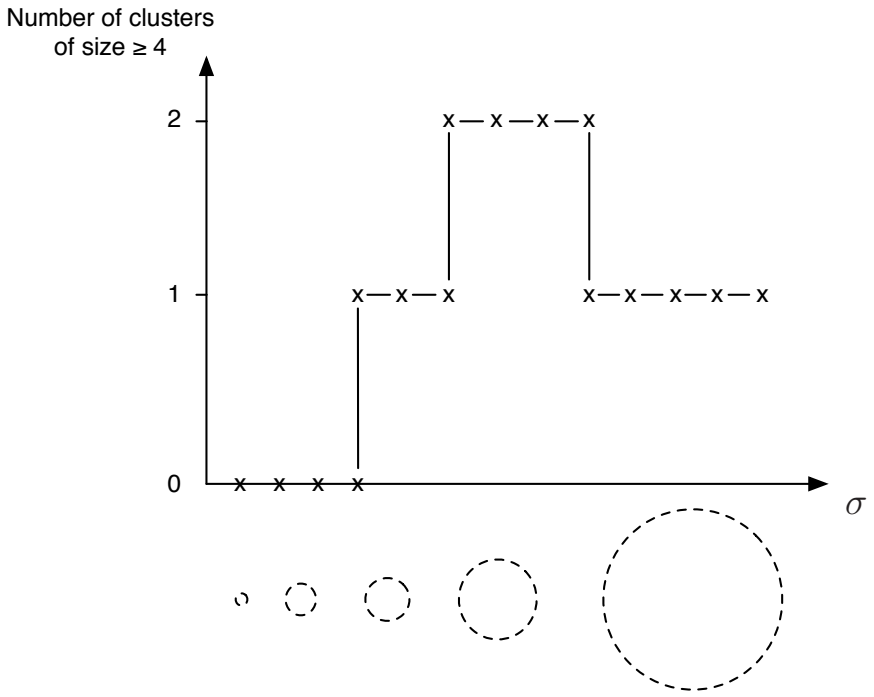


Figure 3.2: Plot of $F(\sigma)$ for increasing values of σ. Each x symbolises a single point of the plot, which have been connected by lines for ease visualisation. Values of σ along the horizontal axis are represented by dashed circles with radius σ. The function $F(\sigma)$, defined as the number of clusters of size ≥ 4, is plotted on the vertical axis.

Fig. 3.3 (b) shows an example of the plot of $F(\sigma)$, which was obtained using the dataset presented in Fig. 3.3 (a). We can see that $F(\sigma)$ begins from zero, and then increases to its maximum value –3–, before settling back down to a value 1. The final value will always be one, because the σ-neighbourhood graph is fully connected for sufficiently large σ values, i.e. it has only one component.

DifFUZZY computes the number of clusters K as the maximum value of $F(\sigma)$, i.e.

$$K = \max_{\sigma \in (0,\infty)} F(\sigma).$$

For the example in Fig. 3.3 (b), we have $K = 3$, which agrees with the number of true globular clusters in the original dataset of Fig. 3.3 (a).

In Fig. 3.3 (b) we see that there is an interval of values of σ for which $F(\sigma)$ reaches its maximum value K. At the next step, DifFUZZY computes σ^*, which is defined as the smallest σ for which $F(\sigma)$ is equal to K. Then the σ^*-neighbourhood graph is constructed. The components of this graph which contain at least M vertices will form the "cores" of the clusters to be identified. Each data point, $\mathbf{x}_i$, which lies in the k^{th} core, is assigned the membership values $u_{i,k} = 1$ and $u_{i,l} = 0$ for $l = 1, 2, \ldots K, l \neq k$, meaning that this point fully belongs to the k^{th} cluster. Every such point will be called a hard point in what follows. The remaining points are called soft points. Now what is left is to assign a membership value to each soft point. This will be done in two steps. First, we compute three auxiliary matrices in Section 3.1.2, and then we assign the membership values to soft points in Section 3.1.3.

3.1.2 Computation of auxiliary matrices W, D and P

Before the membership values of soft points are calculated, DifFUZZY computes three auxiliary matrices: **W**, **D** and **P**, where **W** is a weight matrix, containing the weight of the edge between nodes, **D** is the degree matrix, diagonal matrix with information about the degree, or number of edges that reach each node, and finally **P** the transition matrix whose entries represent the probability from moving from one node to another in one time step.

To achieve this we first define a family of matrices $\widehat{\mathbf{W}}(\beta)$ with entries

$$\widehat{W}_{i,j}(\beta) = \begin{cases} 1 & i \text{ and } j \text{ are hard points in the same cluster core} \\ \exp\left(-\frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{\beta}\right) & \text{otherwise,} \end{cases} \tag{3.3}$$

where β is a positive real number that we now determine. Now let the function $L(\beta) : (0,\infty) \to (0,\infty)$ be the sum

$$L(\beta) = \sum_{i=1}^{N}\sum_{j=1}^{N} \widehat{W}_{i,j}(\beta). \tag{3.4}$$

The log-log plot of function $L(\beta)$ is shown in Fig. 3.3 (c) for the dataset given in Fig. 3.3 (a). We can see that it has two well defined limits;

$$\lim_{\beta\to 0} L(\beta) = N + \sum_{k=1}^{K} n_k(n_k - 1), \tag{3.5}$$

and

$$\lim_{\beta\to\infty} L(\beta) = N^2, \tag{3.6}$$

where n_k is the number of points in the k^{th} cluster core. As explained in Section 3.1.4, we are interested in finding a β value that corresponds to an intermediate value of $L(\beta)$. DifFUZZY does this by finding β^* which satisfies the relation

$$L(\beta^*) = (1-\gamma_1)\left(N + \sum_{k=1}^{K} n_k(n_k-1)\right) + \gamma_1 N^2, \tag{3.7}$$

where $\gamma_1 \in (0,1)$ is an internal parameter of the method. Its default value is 0.3.

Then the auxiliary matrices are defined as follows. Let

$$\mathbf{W} = \widehat{\mathbf{W}}(\beta^*). \tag{3.8}$$

D is defined as a diagonal matrix with diagonal elements

$$D_{i,i} = \sum_{j=1}^{N} W_{i,j}\,, \quad i = 1,2,\ldots,N, \tag{3.9}$$

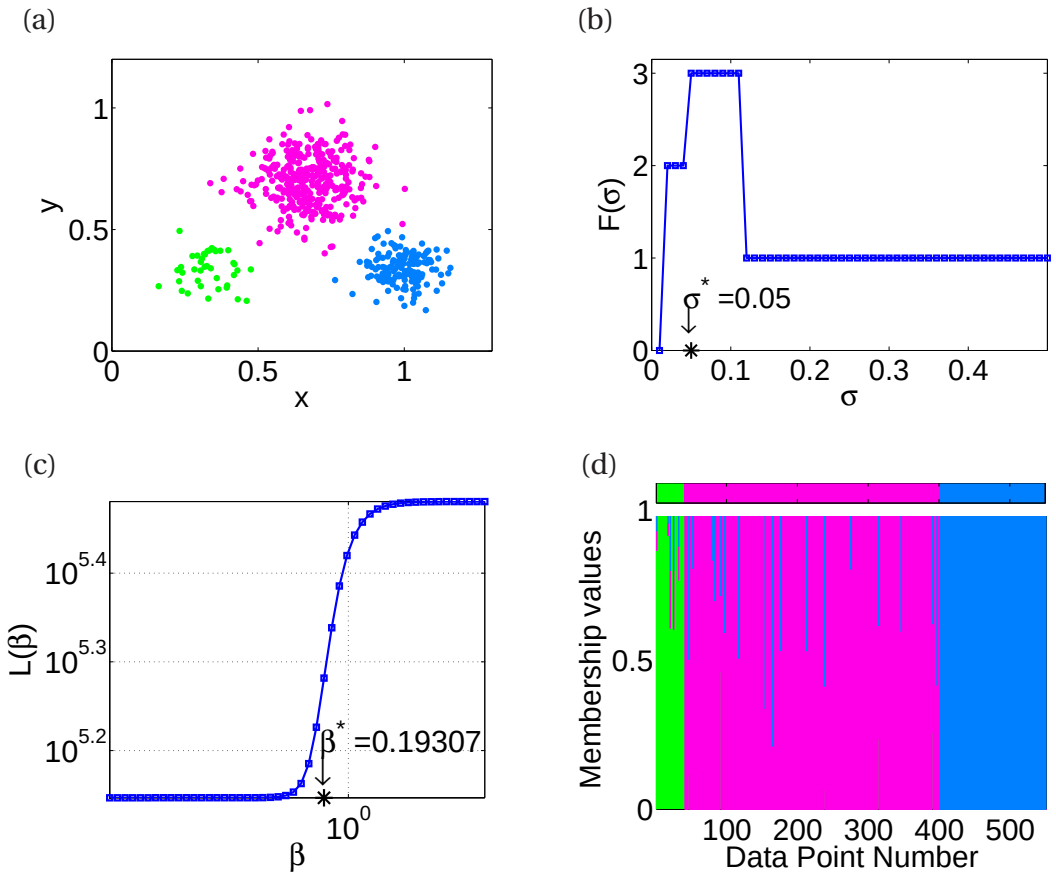


Figure 3.3: (a) *Globular clusters* dataset. Colour code: green, red and blue data points correspond to the clusters. (b) Plot of $F(\sigma)$ for this dataset when M = 35. $F(\sigma)$ is maximal at $\sigma^* = 0.05$ giving three clusters ($K = 3$). (c) Plot of $L(\beta)$ for this dataset, given by Eq. (3.4) plotted on a logarithmic scale. $\beta^* = 0.068665$ was obtained using Eq. (3.7). (d) DifFUZZY membership values for this dataset. Each data point is represented by a stacked bar of total height 1 (from Eq. (3.2)). The different colours represent membership degrees in the true clusters. The colour bar above shows the true membership values of the data points (known from the design or gathering of the dataset). This representation will be used in Figs. 4.14 and 4.15.

where $W_{i,j}$ are the entries of matrix **W**. Finally, **P** is defined as

$$\mathbf{P} = \mathbf{I} + [\mathbf{W} - \mathbf{D}] \frac{\gamma_2}{\max\limits_{i=1,2,\ldots N} D_{i,i}}, \tag{3.10}$$

where $\mathbf{I} \in \mathbb{R}^{N \times N}$ is the identity matrix and γ_2 is an internal parameter of DifFUZZY, with default value of 0.1.

The definition of the matrices **W**, **D** and **P** can be intuitively understood in terms of diffusion processes on graphs, as explained in Section 3.1.4.

3.1.3 The membership values of soft data points

Let $\mathbf{x}_s$ be a soft data point, i.e. a point that does not belong to a cluster core. To assign its membership value $u_{s,k}$ in cluster $k \in \{1,2,\dots K\}$, we first find the hard point in the k^{th} core which is closest (in Euclidean distance) to $\mathbf{x}_s$. This point will be denoted by $\mathbf{x}_{k(s)}$ in what follows. Based on the matrix $\mathbf{W}$ defined by Eq. (3.8), we construct a new matrix $\overline{\mathbf{W}}$ which is equal to the original matrix $\mathbf{W}$, with row s replaced by row $k(s)$ (index of point closer to $\mathbf{x}_s$) and column s (index of $\mathbf{x}_s$) replaced by column $k(s)$. Using $\overline{\mathbf{W}}$ instead of $\mathbf{W}$, matrices $\overline{\mathbf{D}}$ and $\overline{\mathbf{P}}$ are computed from (3.9) and (3.10), respectively. We also compute an auxiliary integer parameter α by

$$\alpha = \left\lfloor \frac{\gamma_3}{|\log \lambda_2|} \right\rfloor, \tag{3.11}$$

where λ_2 corresponds to the second largest eigenvalue of $\mathbf{P}$, $\lfloor\cdot\rfloor$ denotes the floor function and γ_3 is the third internal parameter of the method.

Next, we compute the diffusion distance between the soft point $\mathbf{x}_s$ and the k^{th} cluster by

$$\text{dist}(\mathbf{x}_s, k) = \left|\left| \mathbf{P}^{\alpha}\mathbf{e} - \overline{\mathbf{P}}^{\alpha}\mathbf{e} \right|\right|, \tag{3.12}$$

where $\mathbf{e}(j) = 1$ if $j = s$, and $\mathbf{e}(j) = 0$ otherwise. Finally, the membership value of the soft point $\mathbf{x}_s$ in the k^{th} cluster, $u_{s,k}$, is determined by the following formula:

$$u_{s,k} = \frac{\text{dist}(\mathbf{x}_s, k)^{-1}}{\sum_{l=1}^{K} \text{dist}(\mathbf{x}_s, l)^{-1}}. \tag{3.13}$$

This procedure is performed for every soft data point $\mathbf{x}_s$ and every cluster $k \in \{1,2,\dots,K\}$.

3.1.4 Geometric and graph interpretation of DifFUZZY

In this section, we provide an intuitive geometric explanation of the ideas behind the DifFUZZY algorithm: namely that two data points ($\mathbf{x}_i$ and $\mathbf{x}_j$) belong to the same cluster if there is a high probability that a random walk originating at the node representing $\mathbf{x}_i$

reaches the $\mathbf{x}_j$'s node before reaching a node of a data point of a different cluster [Harel and Koren, 2001], or that a random walk rarely jumps between clusters and stays most of the time within a cluster [Luxburg, 2007]. In the Appendix B.1 we show how diffusion can be understood in terms of a random walk over a network.

The matrix $\mathbf{P}$ can be thought of as a transition matrix whose rows all sum to 1, and whose entry $\mathrm{P}_{i,j}$ corresponds to the probability of jumping from the node (data point) i to the node j in one time step. The j^{th} component of the vector $\mathbf{P}^{\alpha}\mathbf{e}$, which is used in (3.12), is the probability of a random walk to end up in the j^{th} node, $j = 1, 2, \ldots, N$, after α time steps (hence taking the integer in Eq. (3.11)), provided that it starts in the s^{th} node.

In this geometric interpretation we can give an intuitive meaning to the auxiliary parameters γ_1, γ_2 and γ_3. The parameter $\gamma_1 \in (0,1)$ is related to the time scale of this random walk. $\gamma_1 \sim 1$ corresponds to the case where all the nodes are highly connected, and therefore the diffusion will occur almost instantaneously, whereas for values of $\gamma_1 \sim 0$, there will be close to no diffusion between any pair of cluster cores. Therefore, we are interested in an intermediate point, where there is enough time to diffuse, but where equilibrium has not yet been reached.

The parameter $\gamma_2 \in (0,1)$ ensures that none of the entries of the transition matrix $\mathbf{P}$ are negative, which is required, since they represent transition probabilities and it can be interpreted as the length of the time step of the random walk on the graph. For very small values of γ_2 we have $P \sim I$, for which the probabilities of transition between different data points is close to zero, therefore there will not be any diffusion during one time step.

The parameter $\gamma_3 \in (0,\infty)$ is the number of time steps the random walk is going to be run or propagated, capturing information of higher order neighborhood structure [Coifman and Lafon, 2006]. Small values of γ_3 give us few time steps, whereas large values of γ_3 give us a large number of time steps. In the first situation not much diffusion has taken place, whereas in the latter case, when the random walk is propagated a very large number of time steps, the diffusion process is near to reaching equilibrium.

The matrix $\overline{\mathbf{P}}$ is used to represent a different diffusion process, an equivalent one to the first random walk, but over a new graph, where the data point $\mathbf{x}_s$ has been moved to the position of the data point $\mathbf{x}_n$. This matrix then corresponds to the transition matrix for this auxiliary graph. A cartoon-type representation of this graph interpretation of DifFUZZY's diffusion distance is shown in Fig. B.5 in Appendix B.2.

3.2 DifFUZZY's performance over test datasets

In this section we apply DifFUZZY to the three test datasets introduced in the previous chapter, in Fig. 2.1.

A detailed comparison of DifFUZZY with alternative clustering methods presented in Chapter 2 over several benchmark datasets, including the ones in Figs. 3.4–3.6, will be shown in the next chapter.

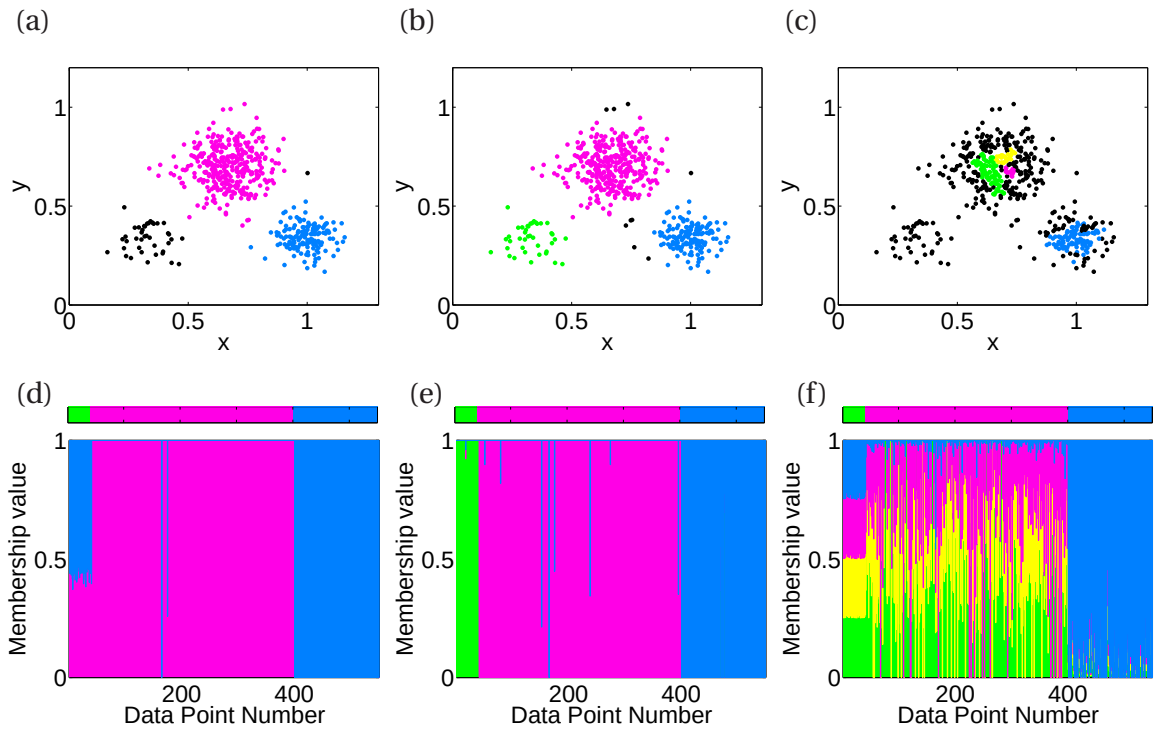


Figure 3.4: DifFUZZY applied to the *Globular clusters* dataset (Fig. 2.1 (a)). (a) HCT-plot, $z = 0.9$, M = 150, (b) HCT-plot, $z = 0.9$, M = 39, (c) HCT-plot, $z = 0.9$, M = 17, (d) membership values, M = 150, (e) membership values, M = 39 and (f) membership values, M = 17.

For the *Globular clusters* dataset (Fig. 3.4) we see that for M = 39 DifFUZZY identifies with precision the three clusters. For large values of M (>150), the green cluster in Fig. 3.4 (b) becomes a group of soft data points with similar membership values, nearly 0.6 on the blue cluster and 0.4 on the pink, larger cluster. If the parameter M is sufficiently small (<17), then the pink cluster is separated into three cluster cores, the blue cluster remains well separated, and the bottom left cluster is again left as a set of soft data points with membership values roughly uniformly distributed among the cluster cores.

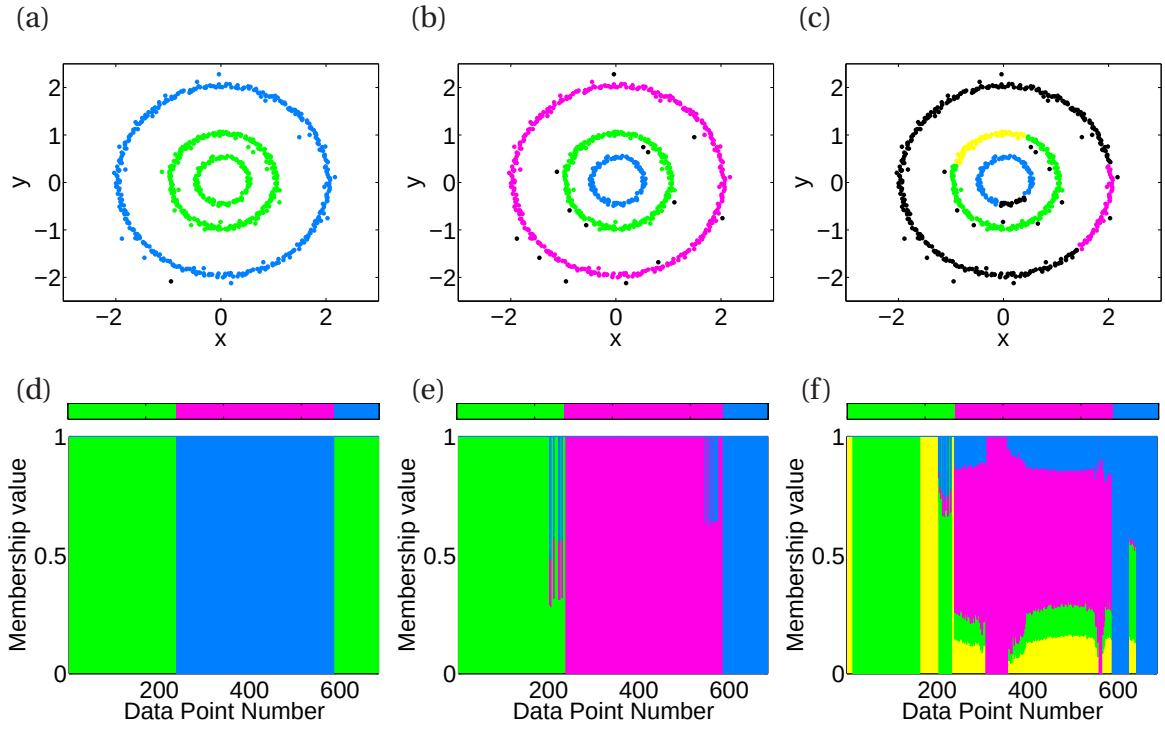


Figure 3.5: DifFUZZY applied to the *Concentric rings* dataset (Fig. 2.1 (b)). (a) HCT-plot, $z = 0.9$, M = 300, (b) HCT-plot, $z = 0.9$, M = 90, (c) HCT-plot, $z = 0.9$, M = 50, (d) membership values, M = 300, (e) membership values, M = 90 and (f) membership values, M = 50.

Similarly to the *Globular clusters* dataset, the *Concentric rings* dataset poses no challenge to DifFUZZY, which can identify the three distinct clusters (Fig. 3.5 (b)) for M = 90. For larger values of M (> 300), the two internal rings are merged into one cluster and naturally, for even larger M values, the three clusters are identified as one cluster (figure not shown). For small values of M (< 50), the middle cluster is separated into two clusters and the cluster cores reduce their sizes, yielding more soft data points.

However, the inner-most cluster is identified fairly well, and in the case of the outer-most cluster, the data points show higher membership values in the correct ring (pink cluster).

Fig. 3.6 shows that DifFUZZY identifies the two elongated clusters (M = 300) with few soft data points. For smaller values of M, the right cluster is allocated into two clusters, a horizontal and a roughly vertical line-shaped cluster, and a cluster core is identified in the diagonal, left cluster, where additional cluster cores are found for even smaller M values.

The soft data points in Fig. 3.6 (b)-(c) present intermediate membership values in all identified clusters. This is a consequence of the choice of diffusion parameters for such dense clusters, for which the diffusion processes reaches equilibrium almost instantly and, as a consequence, the membership values of the soft data points are assigned equally to every cluster core, either in the right or left cluster.

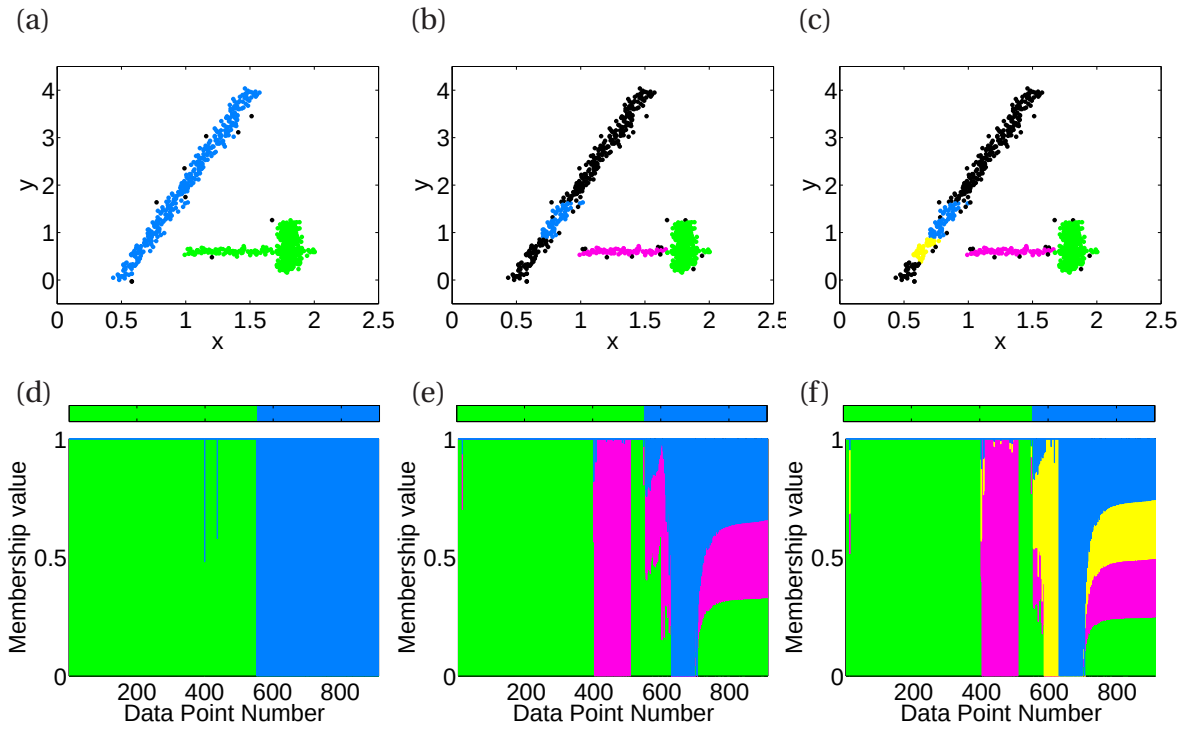


Figure 3.6: DifFUZZY applied to the *Elongated clusters* dataset (Fig. 2.1 (b)). (a) HCT-plot, $z = 0.9$, M = 300, (b) HCT-plot, $z = 0.9$, M = 60, (c) HCT-plot, $z = 0.9$, M = 40, (d) membership values, M = 300, (e) membership values, M = 60 and (f) membership values, M = 40.

If a smaller value for the time scale of the random walk (parameter γ_1) had been chosen, then the soft data points would have presented significantly higher membership values to their true cluster cores, as shown in Fig. B.6 (b)-(c) in Appendix B.3. In Section 3.2.1 we explore the effects of the internal parameters of DifFUZZY using this example (*Elongated clusters* dataset, M = 60), which features a number of soft data points whose membership values vary for different γ values.

The previous three examples indicate how the value of M affects various characteristics of the results such as the number of cluster cores and the number of soft data points. In these three datasets, the original clusters were recovered with the highest accuracy when the number of soft data points was the lowest. This suggests that, in general, M values that give the fewest soft points can lead to the best partitioning, the one closest to the natural clustering structure of the data.

3.2.1 Exploring the effect of DifFUZZY's internal parameters

In Figs. 3.7-3.9 we show results of DifFUZZY applied to the *Elongated clusters* dataset, with M = 60, which yields three cluster cores (Fig. 3.6 (b)). The membership values of the soft data points depend on the values of the internal parameters γ_1, γ_2 and γ_3.

The results obtained using the default parameter values ($\gamma_1 = 0.3$, $\gamma_2 = 0.1$ and $\gamma_3 = 1$) are shown in Figs. 3.7 (b), 3.8 (b) and 3.7 (b). These default values give similar membership values for the soft data points in all clusters, which indicates that the high density of the clusters makes the diffusion process almost instantaneous, which is maintained if the time scale of the random walk is further increased (Fig. 3.7 (c)). In the extreme case, this gives membership values $u_{i,k} = \frac{1}{K} = \frac{1}{3}$, $i = 1,2,\ldots N_{\text{und}}$, $k = 1,2,3$, where N_{und} is the number of soft data points (Fig. 3.8 (a)).

Decreasing the time scale of the random walk (γ_1) or reducing the number of time steps (γ_3) allows the process to reach intermediate levels, and it gives results closer to the real partitioning, where the blue cluster is better identified (Figs. 3.7 (a) and 3.9 (a)).

If the length of the time scale is further increased (γ_2), then the soft data points of the blue cluster are assigned with higher membership values to the other clusters (Fig. 3.8 (c)), getting further away from the real cluster structure of the *Elongated clusters* dataset.

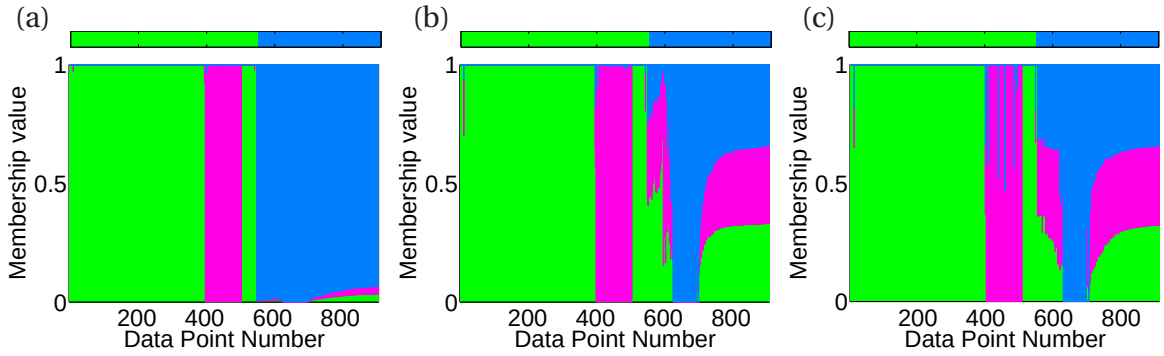


Figure 3.7: DifFUZZY applied to the *Elongated clusters* dataset (Fig. 2.1 (b)), M = 60. Membership values (a) $\gamma_1 = 0.01$, (b) $\gamma_1 = 0.3$, (c) $\gamma_1 = 10$.

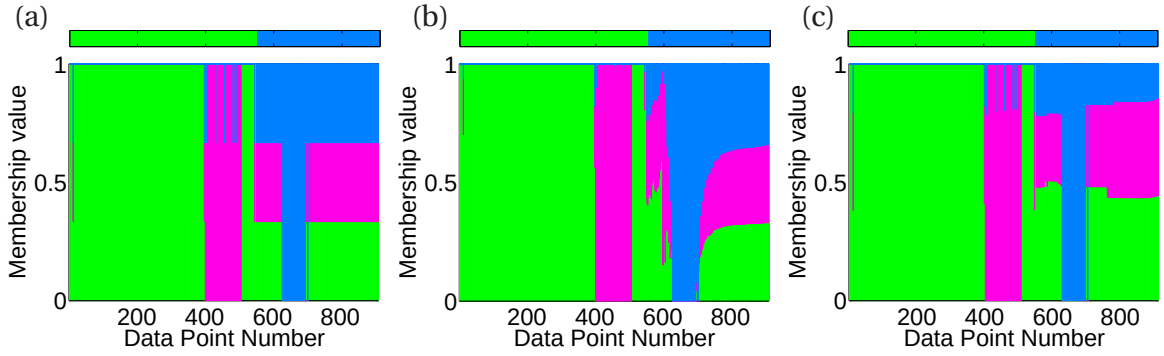


Figure 3.8: DifFUZZY applied to the *Elongated clusters* dataset (Fig. 2.1 (b)), M = 60. Membership values (a) $\gamma_2 = 0.001$, (b) $\gamma_2 = 0.1$, (c) $\gamma_2 = 1$.

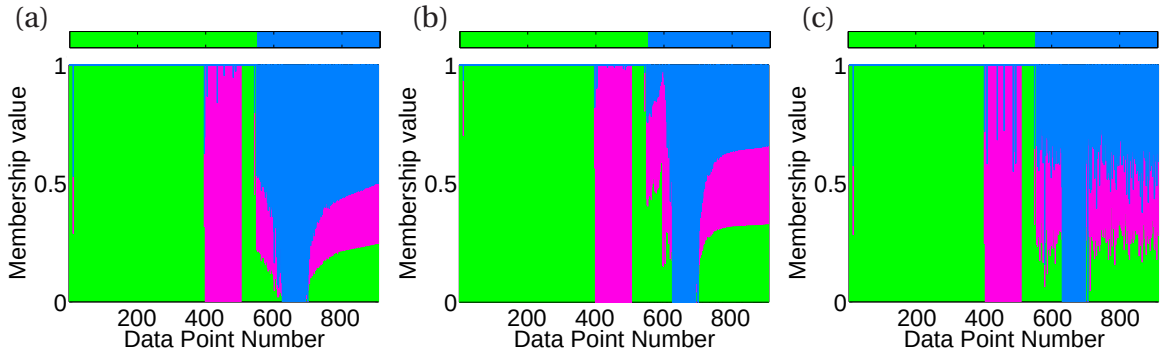


Figure 3.9: DifFUZZY applied to the *Elongated clusters* dataset (Fig. 2.1 (b)), M = 60. Membership values (a) $\gamma_3 = 0.01$, (b) $\gamma_3 = 1$, (c) $\gamma_3 = 100$.

When increasing the number of steps of the random walk (γ_3), the soft data points are also assigned to the three clusters (Fig. 3.9 (c)), but with noisier membership values than when reducing the length of time step (Fig. 3.8 (a)).

We have shown that if we have highly dense clusters and membership values equally distributed between the clusters, then reducing the parameter γ_1 (or alternatively reducing the value of γ_3) can enable us to assign soft data points in a better way.

However, we also notice that by using different values of γ_3, we obtain different resolutions of the clustering of the dataset, so it would be possible to obtain a hierarchy of clusters and a dynamic clustering as the number of times steps in the random walk increases until reaching equilibrium, when all the clusters will form one larger cluster. The results obtained through this approach should be compared with those given by similar random-walk multi-resolution approaches [Lambiotte et al., 2008; Lambiotte, 2010].

3.3 Exploring the $F(\sigma)$ function

Fig. 3.2 depicted a schematic representation of $F(\sigma)$ for a range of increasing σ values. In this particular plot, $F(\sigma)$ is zero for small values of σ, but as the radius of the σ-neighbourhood increases so does $F(\sigma)$. This increment is gradual until it attains a maximum, before settling back down to a value of 1. This is not always the case.

The function $F(\sigma)$ can sometimes display more than one maximum peak, such as in the case shown in Fig. 3.10 (b). We name this dataset the *'3-5 clusters'* dataset, as it was created using three main clusters (blue, yellow/orange and green/purple), two of which can be subdivided into two (yellow/orange and green/purple).

In the cases where $F(\sigma)$ has more than one peak (suggesting the possibility of having two different partitionings, and more crucially, two different sets of cluster cores) the algorithm is modified and local maxima are investigated on a case-by-case basis.

In order to further investigate such double-peaked behaviour, we constructed a 2-D "toy" dataset (depicted in Fig. 3.10) that clearly replicates this special behaviour, which is very rarely found in real datasets. When this situation is present in a dataset, we will refer to it as a "sink" feature in the $F(\sigma)$ plot.

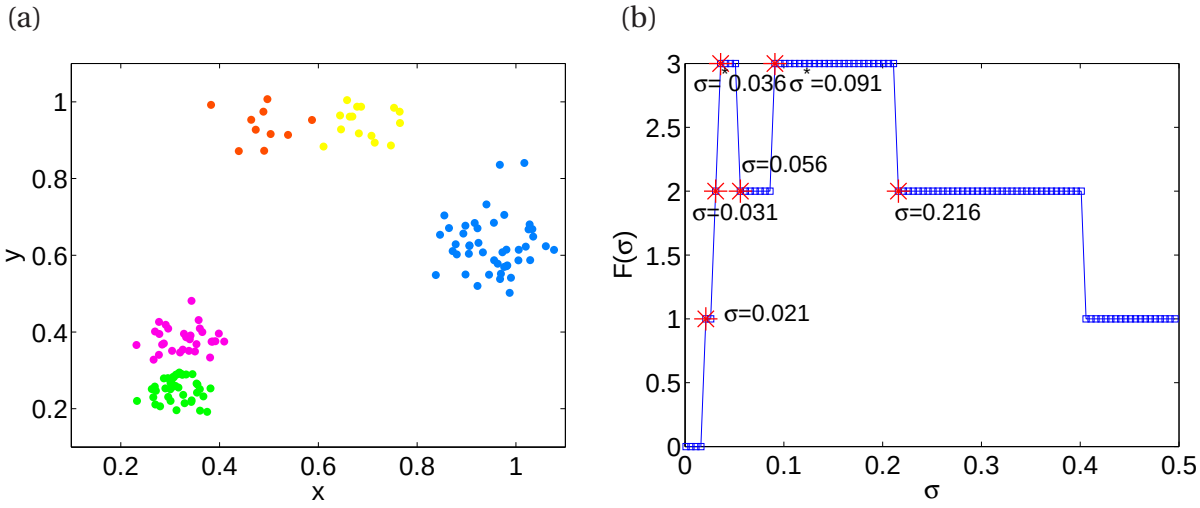


Figure 3.10: (a) *'3-5 clusters'* dataset. (b) $F(\sigma)$, M = 25 for the dataset in (a), showing different values of σ^*.

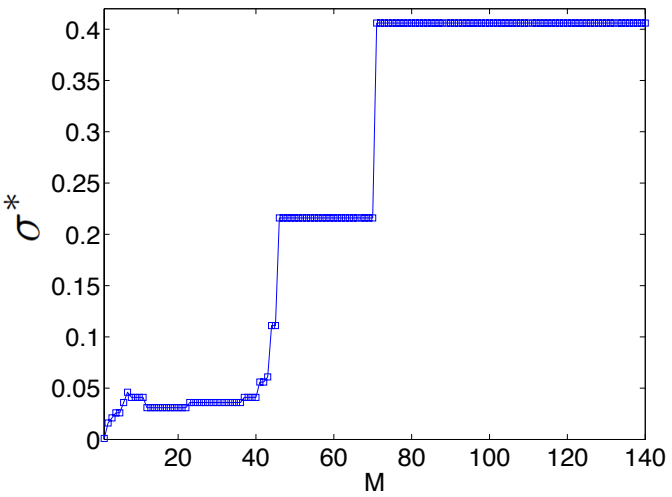


Figure 3.11: σ^* versus M for the dataset in Fig. 3.10 (a).

The shape of the curve we obtain for $F(\sigma)$ is highly dependent on the choice of value for the M parameter, hence, we have generated curves and the value of σ^* for all possible values of M in our example. This corresponds to integer values ranging from 1 to 140, the total number of data points (see Fig. 3.11).

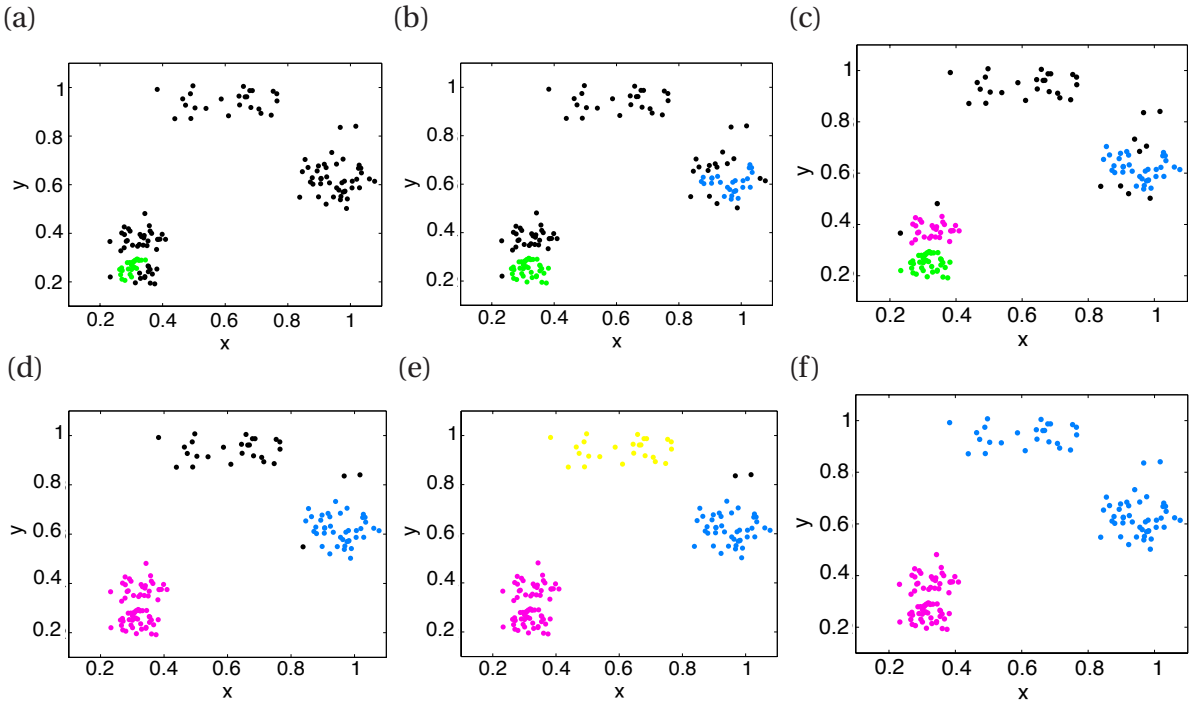


Figure 3.12: DifFUZZY cluster cores for M = 25 on the *'3-5 clusters'* dataset. (a) $\sigma^* = 0.021$, (b) $\sigma^* = 0.031$, (c) $\sigma^* = 0.036$, (d) $\sigma^* = 0.056$, (e) $\sigma^* = 0.091$ and (f) $\sigma^* = 0.216$. In black are the soft data points.

Fig. 3.12 helps to explain the occurrence of the "sink" feature in the $F(\sigma)$ function for this dataset. The 6 subplots show the core clusters for the 6 values of σ identified with an asterisk in Fig. 3.11 (b), which are the smallest values of σ for which there is a change in the value of the function F. There are two values of σ^*, (0.036 (c) and 0.091 (d)) for which 3 cluster cores are identified[1].

[1] The black data points do not make up a cluster, they are the soft soft points.

The green/pink cluster in Fig. 3.12 (a) gives two cluster cores if the value of σ is small, or one large cluster if σ is large enough. For such large values of σ, the yellow top cloud of points is also identified as a cluster core, giving three cluster cores as before the merge of the green and pink clusters.

Now we must determine which of the two values for σ^* (0.036, 0.091) is the best option. Since we give preference for denser cluster cores we choose the first or smallest σ which gives us the maximum of $F(\sigma)$, regardless of what happens after reaching the maximum (if $F(\sigma)$ decreases and then returns to the maximum or if it monotonically decreases until reaching 1).

In this particular example, if the largest of the two σ^* values is used, the three clusters that would be identified include the top cloud of data points, and a fewer number of soft points would be found altogether.

However, if these data points represent noisy points, then identifying the two more compact sub-clusters in the bottom left corner would be more informative than defining it as one large cluster.

Since the double peaked behaviour is highly dataset-dependent, it has been our experience that it occurs only on isolated occasions, and, as we believe that in most circumstances denser cluster cores are preferable, we suggest that, as a first attempt to cluster the data, we should continue using the smallest of the σ^* values, bearing in mind the possibility of missing part of the structure (e.g. Fig. 3.10). We note that DifFUZZY displays the plot $F(\sigma)$, where the user can identify such peculiarities and then look further into the structure of the data.

3.4 About the identification of the cluster cores

In this section we introduce the *Iris* dataset, a classical clustering benchmark dataset described later in detail in Section 4.1.1.2. This dataset was chosen because its cluster cores are superimposed and, if the features of the *Iris* dataset are selectively rescaled, DifFUZZY can give different cluster cores.

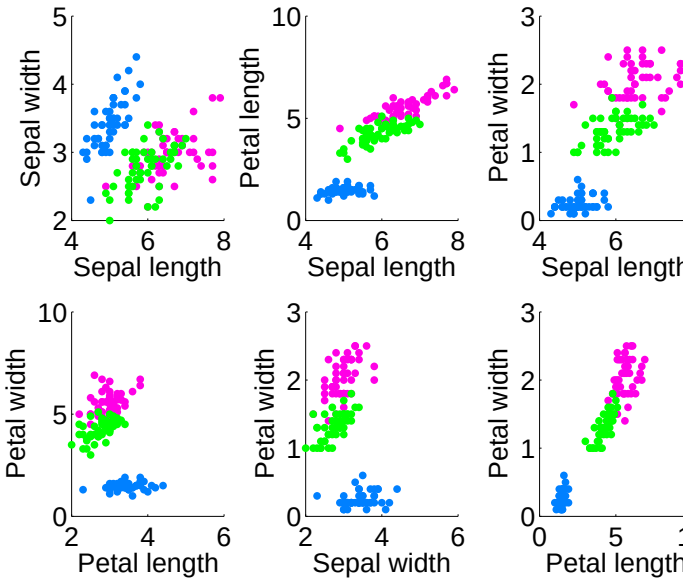


Figure 3.13: Clusters for the *Iris* dataset. Colour code: blue corresponds to the original *Iris setosa* data, green to the original *Iris versicolor* data and red to the *Iris virginica* data.

In Fig. 3.13 we plot the original clusters corresponding to the *Iris* dataset. There we see that the *Iris versicolor* and *Iris virginica* data are superimposed. For M = 15, DifFUZZY performs remarkably well over this dataset, in spite of this superimposition (Fig. 3.14 (a)).

Later, in Table 4.2 we present a summary of the classification error of all the previously presented clustering methods over the *Iris* dataset: DifFUZZY misclassified less than 3% of the data points, while FCM misclassifies 10% of them. The other methods perform much worse than FCM (with the exception of HC, average linkage which performs similar to FCM). However, for larger values of M, some of the cluster cores have data points misassigned (Fig. 3.15).

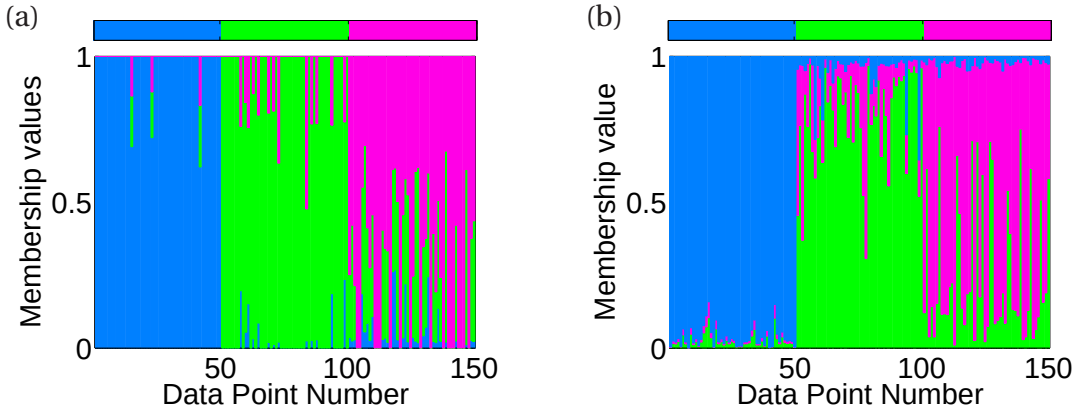


Figure 3.14: (a) DifFUZZY membership values for the *Iris* dataset (M = 15). (b) FCM membership values for the *Iris* dataset (blue: *I. setosa*, green: *I. versicolor* and red: *I. virginica*).

Fig. 3.16 indicates these misassigned cluster core points with an X, and in Fig. 3.15 these data points correspond to the three data points with a complete pink membership value within the green cluster (*Iris versicolor*), therefore giving incorrect clustering results when classifying the soft data points over both clusters. This problem can be addressed by re-scaling each feature (sepal length, sepal width, petal length and petal width) independently, obtaining data which will be better clustered by DifFUZZY.

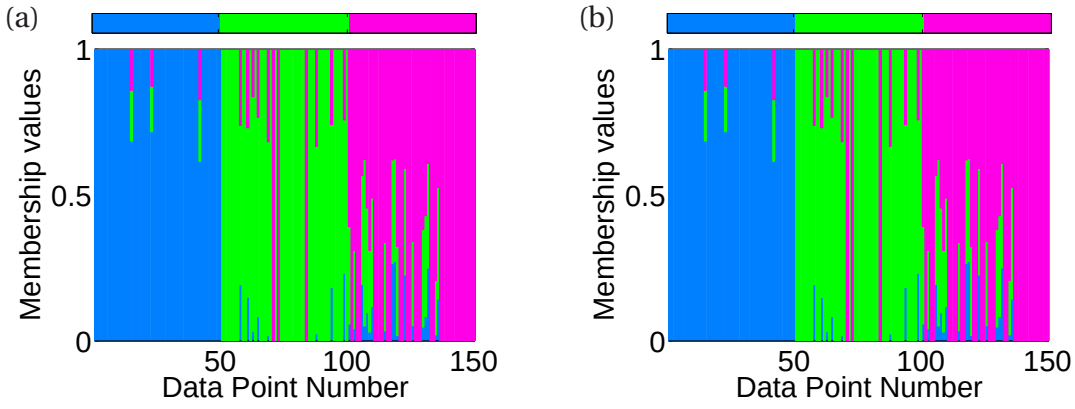


Figure 3.15: (a) DifFUZZY membership values for the *Iris* dataset (M = 20). (b) DifFUZZY membership values for the *Iris* dataset (M = 35).

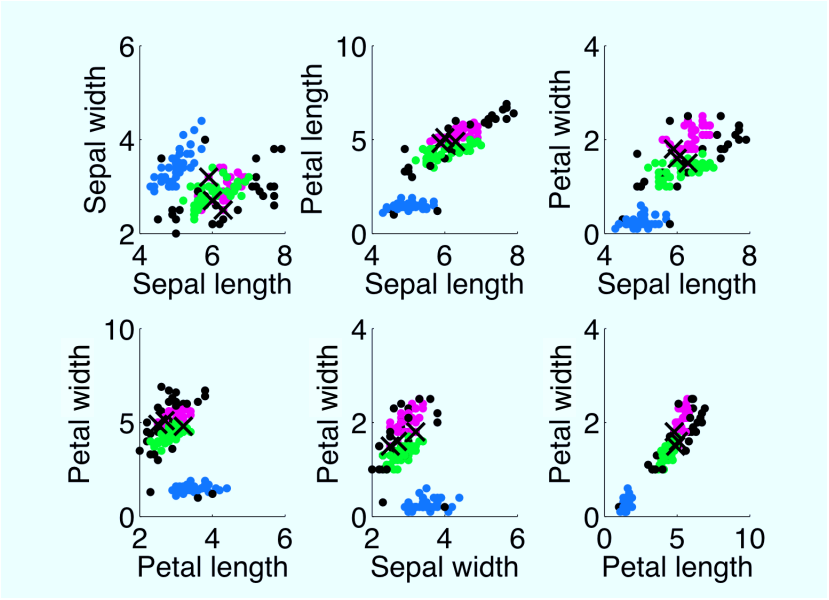


Figure 3.16: DifFUZZY cluster cores of the *Iris* dataset. Marked with X are the wrongly assigned cluster core data points. Colour code: blue represents the cluster core for the *Iris setosa* data, green is the cluster core for the *Iris versicolor* data, pink corresponds to the *Iris virginica* data and black represents soft data points (M = 20).

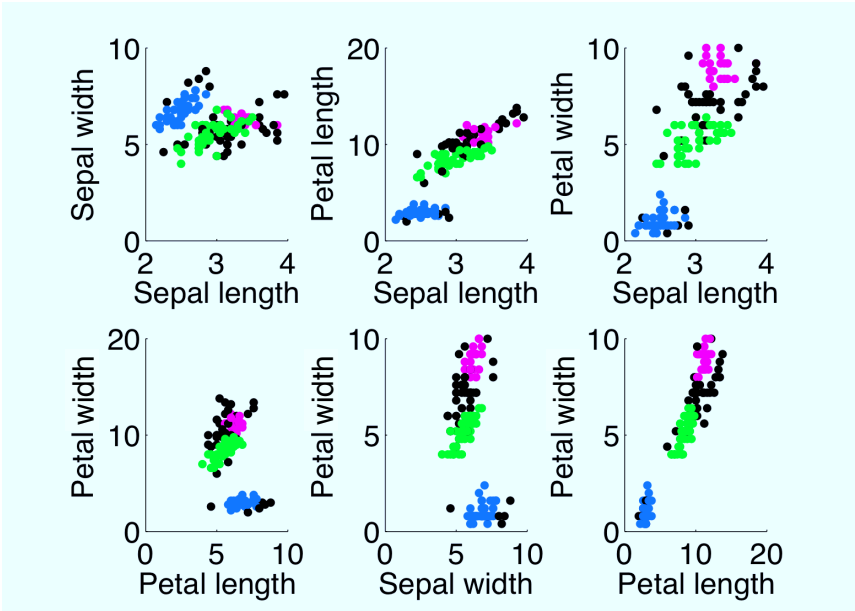


Figure 3.17: DifFUZZY cluster cores of the scaled *Iris* dataset. Feature scaling factors $\{0.5, 2, 2, 4\}$ identified through exploration. There are no wrongly assigned cluster core data points (M = 20).

In Fig. 3.17 we can see that after scaling the *Iris* data there is not a single point misassigned during the identification of the core clusters, for the same value of M used (20 in this particular example). In Fig. 3.18 we show the membership values obtained from applying DifFUZZY and FCM over the scaled *Iris* dataset, and it can be observed that both methods perform very well, identifying the three clusters with a few misassignments, with DifFUZZY having fewer noisy clusters (by this we mean clusters with data points with small membership values in other clusters).

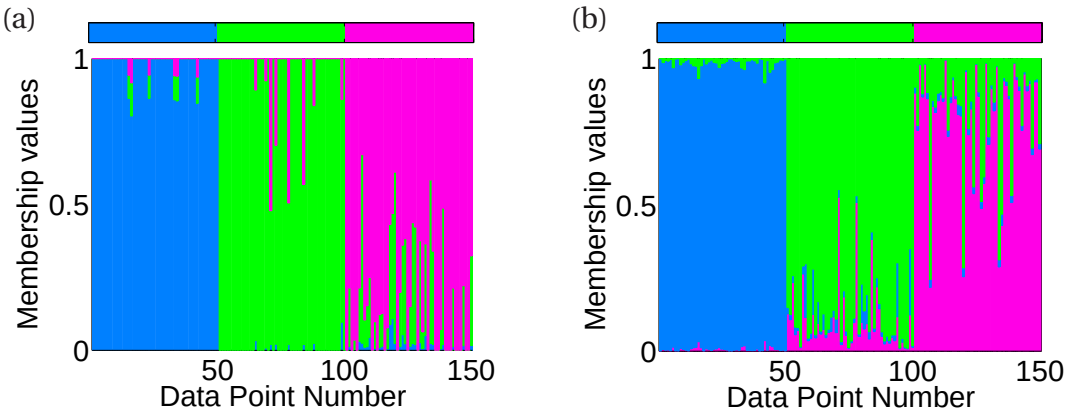


Figure 3.18: (a) DifFUZZY membership values for the Scaled *Iris* dataset. Feature scaling factors $\{0.5,2,2,4\}$ ($M = 20$). (b) FCM membership values for the Scaled *Iris* dataset. Feature scaling factors $\{0.5,2,2,4\}$.

We then considered a computer generated dataset ('*X*' dataset) where we show that a normalisation of the type carried out for the *Iris* dataset may also be required in order to obtain accurate clustering results for some datasets. This example is shown in Fig. 3.19 (a), where in Fig. 3.19 (b), after being scaled, the cluster cores are identified with much more accuracy than without scaling (notice the number of crosses in Fig. 3.19 (a), which represent the misassigned data points in the blue cluster, which in fact corresponds to the right half moon). By both modifying the parameter M and scaling the data we can obtain better results when the problem corresponds to the misassignment of cluster cores.

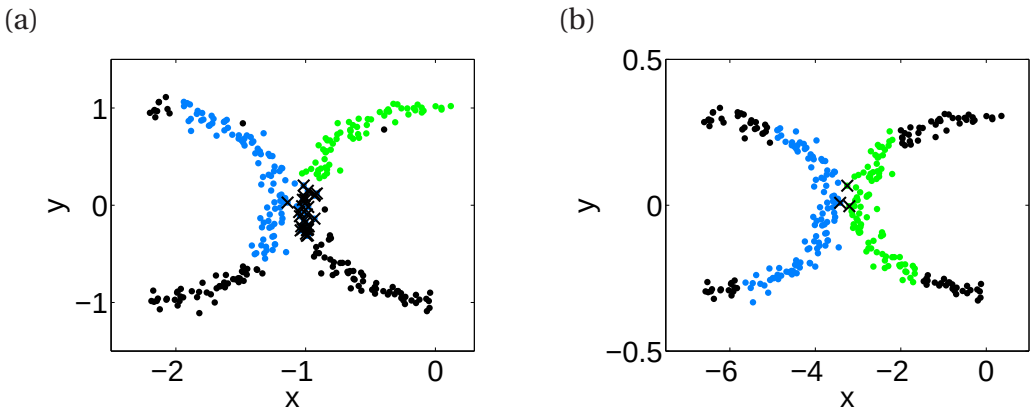


Figure 3.19: '*X*' dataset. (a) Core clusters for the original data (M = 60). (b) DifFUZZY cluster cores for the scaled data, scaling factors $\{3, 0.3\}$ (M = 60). Marked with X are the incorrectly assigned cluster cores.

3.5 Automatic selection of parameter M

Choosing an adequate value for DifFUZZY's parameter M is important for the final outcome of the partitioning of the data. When using large values of M compared to the dataset size, the result will yield one large cluster. Such a result is not very informative, unless the goal is to determine outliers. On the contrary, when using very low values of M, the result will be a very large number of small clusters. Depending on the structure of the data and the question being addressed, either a small or large number of clusters will be more useful and informative. This is why having an automatic method for choosing an appropriate value of M is a clear advantage when the user has little or no information on the data structure.

3.5.1 DifFUZZY silhouette width

Most of the standard techniques used to validate clusters make use of Euclidean distances and determine inter-cluster distances by computing the distance of the points to the centroids of their clusters or to the rest of the data points in that cluster (see Section 2.2). As we have previously shown in examples such as the *Concentric rings* and *Elongated clus-*

ters datasets, the centroids of the clusters are not always a suitable representation of the clusters, and the Euclidean distance between points is not always the best measure of the nearness of a cluster.

As a result, we devised our own quality measure based on the diffusion distance formulated for DifFUZZY, Eq. (3.12), which we will call the DifFUZZY silhouette width (DSW). For the data points in the cluster cores, the DifFUZZY silhouette width is defined as 1, whereas for the soft points it is a function of the diffusive distance to the first and second nearest clusters:

$$\mathrm{dsw}_i = \begin{cases} 1 & \mathbf{x}_i \text{ is a core point} \\ \dfrac{s_{\mathbf{x}_i} - e_{\mathbf{x}_i}}{\max\{s_{\mathbf{x}_i}, e_{\mathbf{x}_i}\}} & \mathbf{x}_i \text{ is an fuzzy point,} \end{cases} \tag{3.14}$$

where $e_{\mathbf{x}_i}$ and $s_{\mathbf{x}_i}$ represent the diffusion distance between the soft data point, $\mathbf{x}_i$, and its first and second closest cluster, respectively.

The average DifFUZZY silhouette width is then computed as the mean of the DifFUZZY silhouette widths of all the data points in the set:

$$\overline{\mathrm{DSW}} = \frac{1}{N}\sum_{i=1}^{N} \mathrm{dsw}_i. \tag{3.15}$$

As with other quality measures, DSW has to be defined for the situation when there is only one large cluster and when there are only singleton sets (clusters with just one element). We choose to use the value of 1, as it could be interpreted as a 'perfect' partitioning, when every data point belongs to the same cluster.

DSW gives a value of 1 for every hard point, and lower values for soft data points. If a soft data point $\mathbf{x}_i$ is highly fuzzy, presenting very similar diffusion distances to its first and second nearest cluster, then it will have a very low DifFUZZY silhouette width, dsw_i. Therefore, we expect to obtain higher values of $\overline{\mathrm{DSW}}$ for datasets with a low percentage of soft points, and lower values for sets of data with a large number of soft points, alerting the user to the fact that the clustering results may be poor.

The parameter M would then be chosen as the one which maximises $\overline{\text{DSW}}$ such that there are at least 2 clusters and at most N clusters, to avoid the trivial solution of every data point forming one large cluster and every data point as a cluster;

$$\text{M}^* = \max_{\text{M}=1,2...N, N>NC>2} DSW_{\text{M}}, \tag{3.16}$$

where M^* corresponds to the automatically optimised value of the M parameter.

A first example that shows the difference between using Euclidean distance versus DifFUZZY's diffusion distance is presented in Fig. 3.20. Starting from the cluster cores misidentified by FCM ($z = 0.9$), we compute the membership values of the soft data points using either the Euclidean or diffusion distances. FCM membership values show an oscillatory pattern[2], assigning high membership to the data points in the *slice*[3], while DifFUZZY assigns very similar membership values to the data points in the outer- or inner-most circle. These data points are as likely to belong to either of the cluster cores, because they will diffuse almost instantly to the vast majority of the points in their rings, and from there diffusion to either of the cluster cores should take the same amount of time. In this case $\overline{\text{DSW}}$ is small, 0.2744, whereas the standard Euclidean $\overline{\text{FSW}}$ is 0.5616, wrongly suggesting this is a reasonable partitioning of this dataset.

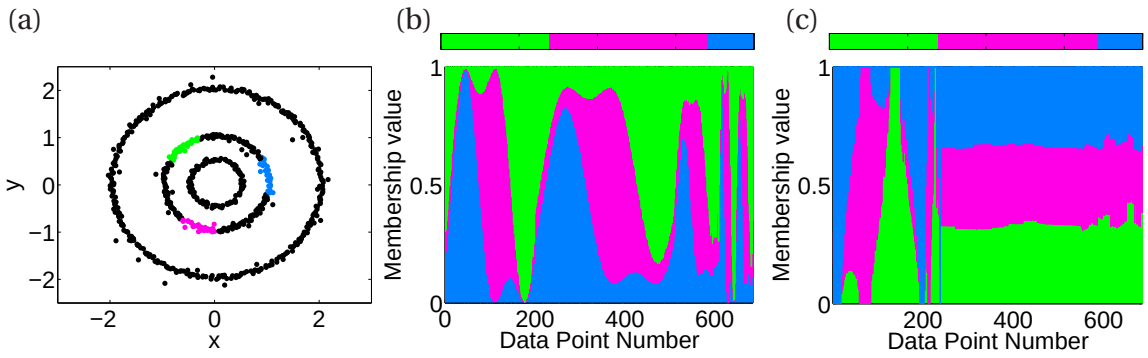
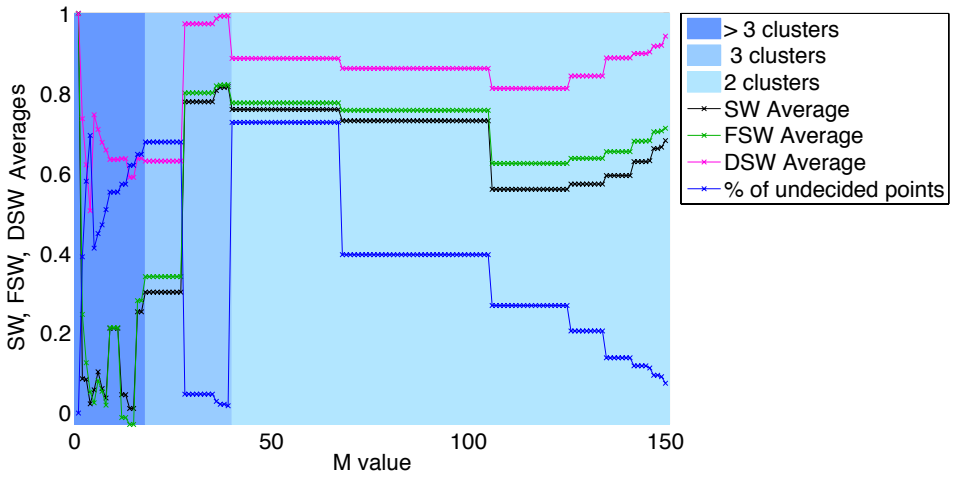


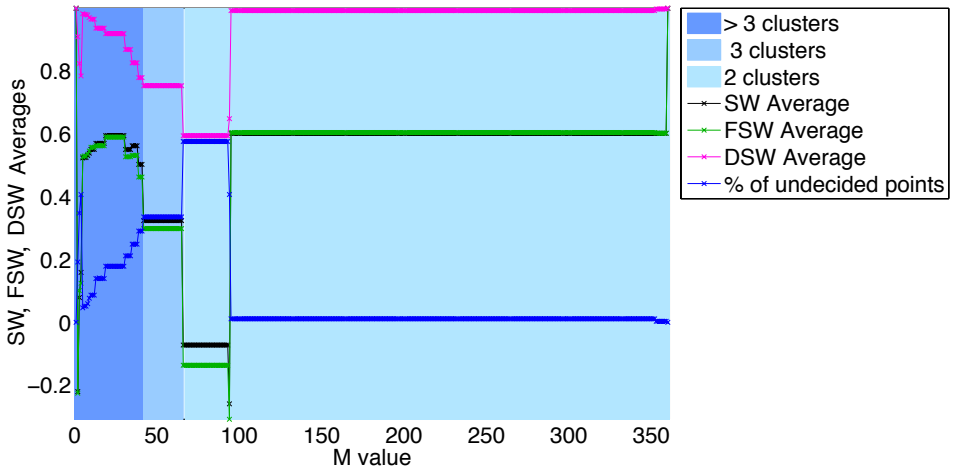
Figure 3.20: FCM applied to the *Concentric rings* dataset, $q = 2$, $K = 3$. (a) HCT-plot, $z = 0.9$, as in Fig. 2.23 (b), (b) membership values, Euclidean distances, as in Fig. 2.23 (e), (c) membership values, diffusion distances.

[2] The oscillatory pattern is a result of the generation of the data points.
[3] By slice we mean data points which have similar azimuthal angles with respect to the centre of the circles.

(a) *Globular clusters* dataset



(b) *Elongated clusters* dataset



(c) *Concentric rings* dataset

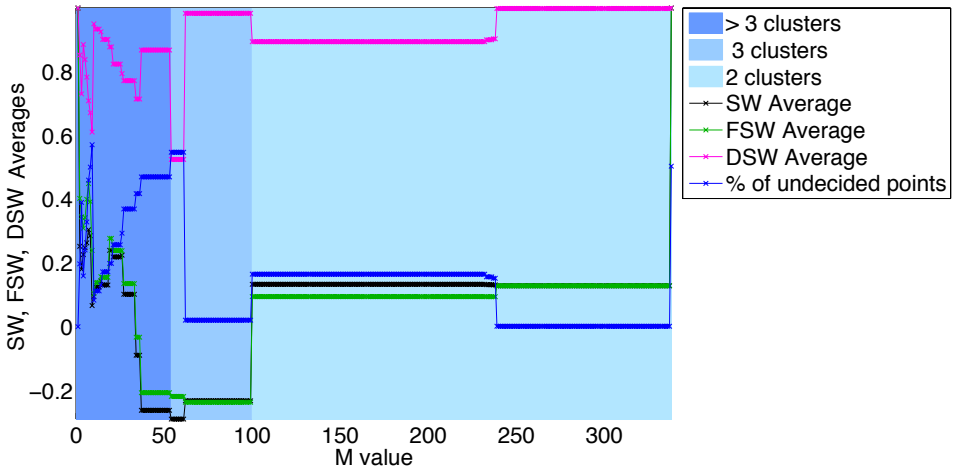


Figure 3.21: For a range of M values we plot the corresponding DSW, FSW and SW quality measure of the results obtained from applying DifFUZZY over the (a) *Globular clusters*, (b) *Elongated clusters* and (c) *Concentric circles* datasets.

3.6 Discussion

The lack of soft methods specifically designed to distinguish clusters of data points that are highly non-linearly separable prompted us to create a novel clustering algorithm, DifFUZZY, that could resolve clusters in convoluted and overlapping data.

A distinct difference between Fuzzy C-means and DifFUZZY is that the latter does not require the cluster centres nor does it calculate them. This translates into a substantial advantage when performing cluster analysis on datasets where the centre is not representative of the cluster, such as concentric rings as seen in Fig. 2.1 (b).

Furthermore, DifFUZZY does not require the fuzziness parameter q as an input, which is especially convenient when studying gene expression data, for which the default parameter $q = 2$ does not yield good results [Dembélé and Kastner, 2003]. Although it is true to say that DifFUZZY requires three user-specified parameters, the default values have consistently proven to give good to optimal results for a broad range of datasets, including those derived from genetic expression experiments.

The first step of DifFUZZY is independent of the next steps and different strategies could be used to assign membership values to the soft points. Alternatively, this novel first step could be used as input parameter by other algorithms to pre-determine the number of clusters in a dataset.

Furthermore, DifFUZZY returns the function $F(\sigma)$ allowing users to identify double peak behaviour (Section 3.3). This further insight of the data lets the user know if the cluster cores identified are unique or not. Many clustering algorithms act like a black box and do not or cannot provide this type of additional information.

DifFUZZY allows the identification of outlier points, and these do not affect the quality of the clustering results, as opposed to the FCM case, where the outliers are also used when calculating the cluster centres. Although there is also the spectral clustering method, which

is used for datasets with non-linear structures, the advantage of DifFUZZY over spectral clustering is the *fuzziness* of the technique, which allows us to assign data points to multiple clusters, enabling the identification of outlier points and superimposed clusters, as well as outperforming on a number of benchmark datasets, as will be shown in Chapter 4.

Using the notion of the diffusion distance in DifFUZZY and standard silhouette widths, we developed an *ad hoc* quality measure specifically targeted for clustering results obtained using our method. The calculation of DifFUZZY Silhouette Widths allowed us to identify the number of clusters in toy datasets and it also suggested that the clustering result obtained using FCM over the *Concentric rings* dataset was not likely to be a true partitioning, which is what FSW wrongly conveyed.

A caveat of DifFUZZY and its associated quality measure is the high dependence on the first step: the correct selection of the cluster cores. In the case of DSW, it assumes that the cluster cores are properly identified, and it gives them a score of 1. In some cases this may not be correct (as is the case of overlapping cluster cores, shown in Section 3.4), and we could modify this by determining the diffusion distance for those data points or by not considering the hard data points.

When the user has prior information on a particular dataset, they can use semi-supervised DifFUZZY, which consists of giving the cluster cores and then using DifFUZZY to compute the membership values of soft points. This could prove useful, for example, in situations when there are additional data which we want to assign to previously identified clusters.

It is possible to identify clusters in data which do not have a natural clustering structure. Here DSW could prove useful to determine how likely the clusters are to represent the intrinsic clustering of the dataset. Although DSW is a suitable quality measure for soft clustering methods, we have to keep in mind what Pal and Bezdek [1997] stated: "no matter how good your index is, there is a dataset out there waiting to trick it (and you)".

Chapter 4

DifFUZZY: Testing and Validation

To better understand the performance of DifFUZZY, it is important to compare it with other clustering algorithms over a wide range of synthetic and real test datasets. In Section 4.1 DifFUZZY is applied to benchmark datasets, widely used to compare clustering techniques, many of which are obtained from biological experiments. Then, in Section 4.2, a number of computer generated test datasets are presented and used to evaluate DifFUZZY, also, illustrating the strengths and weaknesses of the standard FCM method.

4.1 Benchmark datasets

In the next subsections we describe 7 benchmark datasets widely used to compare clustering techniques. In the descriptions of each dataset we include the direct comparisons between FCM[1] and DifFUZZY given that they are both soft techniques and, unlike the other hard methods, can be compared with HCT and ROC curves. The comparison, using classification error, among all the previously described clustering techniques (K-means, FCM, PCM, SOM, spectral and DifFUZZY) applied over these datasets is summarised in Table 4.2.

4.1.1 Description of the benchmark datasets

The benchmark datasets used to evaluate DifFUZZY are obtained from the physical, biological and medical sciences and are well known test datasets in machine learning, clustering and classification: *CRC*, *Ionosphere*, *Iris*, *Leukemia*, *Ruspini*, *WDBC*, *Wine* and *Zoo* datasets.

[1] We do not include PCM in the direct comparison as it consistently gives poorer results than FCM.

4.1.1.1 *CRC* dataset

The *Colorectal Cancer* (*CRC*) dataset [Alon et al., 1999] is a set of gene expression data obtained from DNA microarray experiments and is composed of 62 tissue samples (each described with 2000 expression values), 40 of which have been taken from tumour tissue, and the rest from normal colon tissue. Although in the original dataset each sample is characterised by the expression levels of 2000 selected genes, after preprocessing using standard filtering, thresholding, $\log_{10}$ normalisation and gene selection as done by Tan et al. [2004], the *CRC* dataset was reduced to 370 genes.

It is clear from Fig. 4.1, where we present the plots of the DifFUZZY and FCM membership values for the *CRC* dataset, that DifFUZZY can identify both clusters more accurately than FCM. On average, FCM identifies the clusters, but gives them low membership values, whereas DifFUZZY gives larger membership values, closer to their true labels.

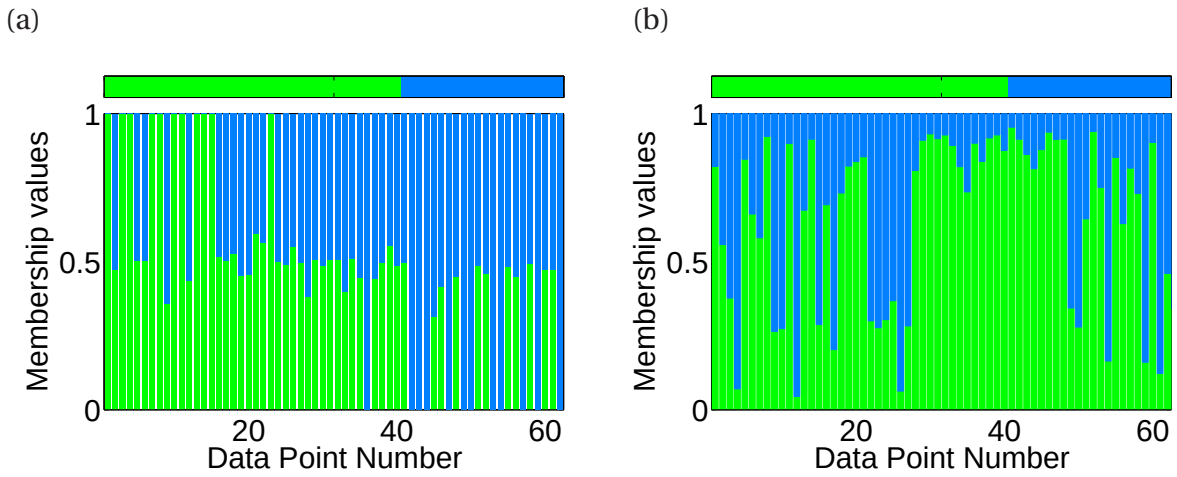


Figure 4.1: (a) DifFUZZY membership values for the *CRC* dataset, M = 11. (b) FCM membership values, $q = 2$, $K = 2$. The colour bars above show the real membership values of the data points.

The corresponding ROC curves for FCM and DifFUZZY are plotted in Fig. 4.2. DifFUZZY clearly outperforms FCM on this dataset since for all thresholds, DifFUZZY's TPR is higher and FPR is lower than those for FCM.

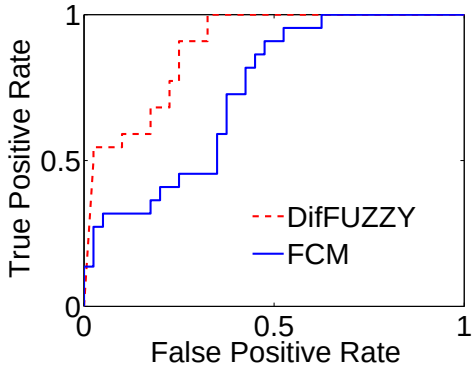


Figure 4.2: DifFUZZY and FCM ROC curves for the *CRC* dataset, M = 11, $q = 2$, $K = 2$.

4.1.1.2 *Iris* dataset

This is a benchmark dataset in pattern recognition[2] . It contains three clusters, corresponding to the three Iris species: *Iris setosa*, *Iris versicolor* and *Iris virginica*. Each cluster has 50 data points comprising the following four dimensions: sepal length, sepal width, petal length and petal width[3]. The cluster *Iris setosa* is linearly separable from the other two, which in turn are not linearly separable from each other [Fisher, 1936].

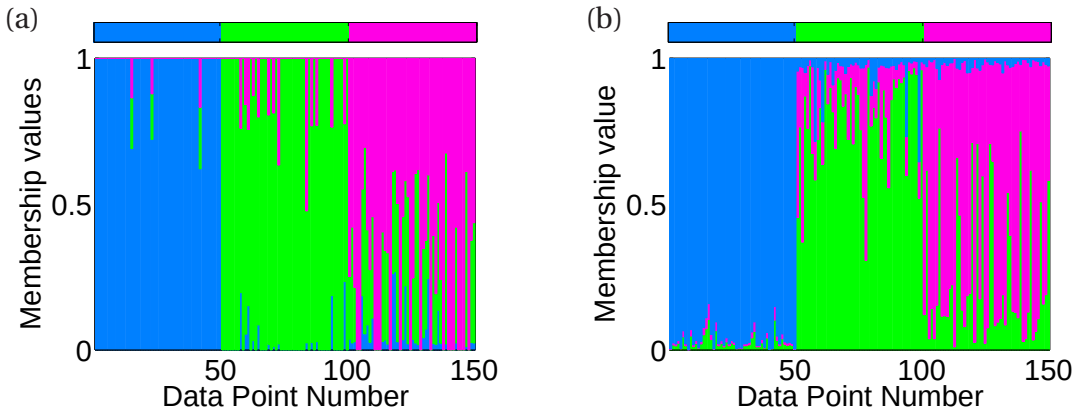


Figure 4.3: (a) DifFUZZY membership values for the *Iris* dataset (M = 15). (b) FCM membership values for the *Iris* dataset (blue: *I. setosa*, green: *I. versicolor* and red: *I. virginica*).

We applied DifFUZZY and FCM to the *Iris* dataset and obtained the results first shown in

[2] The *Iris*, *WDBC*, *Wine* and *Zoo* datasets are all freely available from the University of California Irvine UCI Machine Learning Repository [Asunción and Newman, 2007].

[3] We have used the original dataset, the one whose 90th entries of the four features are {5.5, 2.5, 4, 1.3}[Fisher, 1936], and not the ones published as the *Iris* dataset in Chien [1978] ({5.5, 2.5, 5, 1.3}).

Fig. 3.14 and reproduced here in Fig. 4.3, also shown as ROC curves (see Section 2.2.4.2 for definition) in Fig. 4.4.

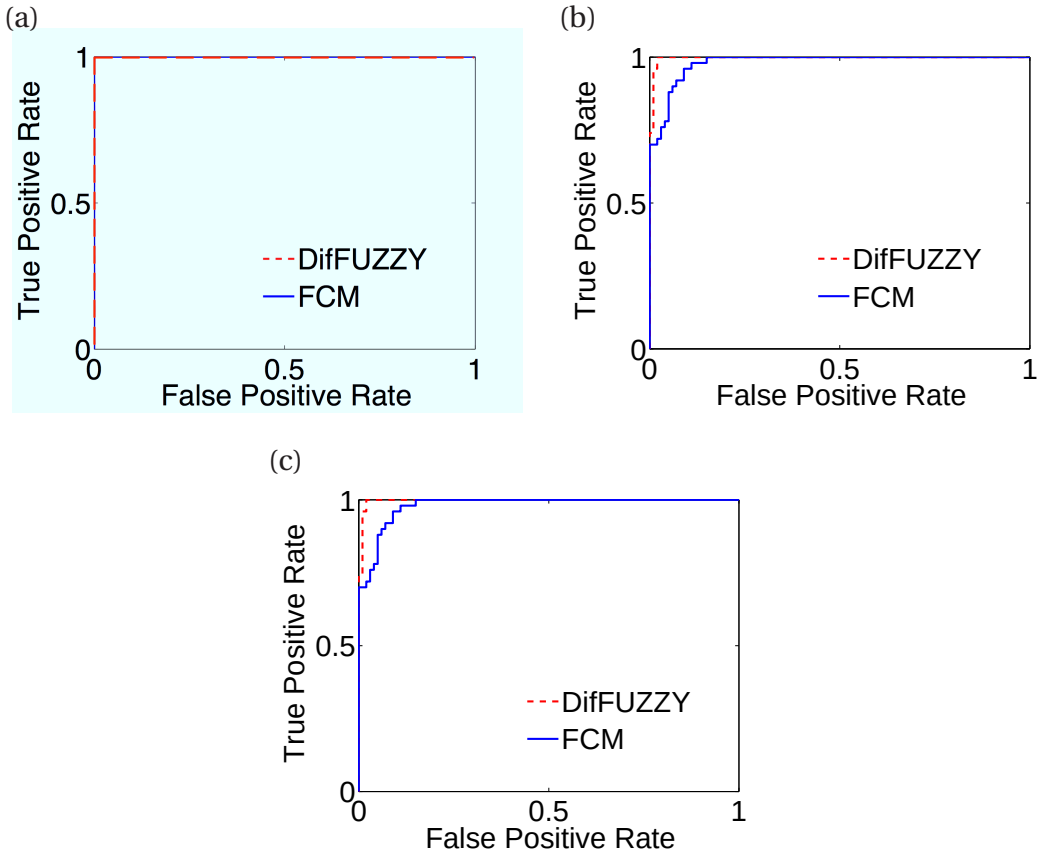


Figure 4.4: DifFUZZY and FCM ROC curves for (a) the *Iris setosa* dataset M = 15, $q = 2$, $K = 2$, (b) the *Iris versicolor* dataset, M = 15, $q = 2$, $K = 2$, (c) the *Iris virginica* dataset, M = 15, $q = 2$, $K = 2$.

Figs. 4.4 (a)-(c) show the ROC curves for the *Iris setosa*, *Iris versicolor* and *Iris virginica* clusters, respectively. For the *Iris setosa* cluster, both DifFUZZY and FCM classify all points correctly (TPR= 1), and do not assign other points to that cluster (FPR= 0), but for the *Iris versicolor* and *Iris virginica* clusters, DifFUZZY performs better than FCM. Additional material regarding the performance of DifFUZZY on the *Iris* dataset for different values of the parameter M was presented and discussed in Section 3.4, such as plots of the membership values and plots of the core clusters. There we showed that datasets with overlapping cluster cores can be better partitioned after scaling each feature by a given weight, or by choosing a smaller value of M.

4.1.1.3 *Leukemia* dataset

The publicly available *Leukemia* dataset [Golub et al., 1999] contains genetic expression data from patients diagnosed with either of two different types of leukemia: acute myeloid leukemia (AML, N=25) or acute lymphoblastic leukemia (ALL, N=47) [Tan et al., 2004]. The dataset, composed of 7129 genes, was obtained from an Affymetrix high-density oligonucleotide microarray.

Before testing our clustering method we pre-processed the data as described by Tan et al. [2004] and obtained 70 genes with the highest squared correlation coefficient sums.

Fig. 4.5 presents the plots of the DifFUZZY and FCM membership values for the *Leukemia* dataset. Here we can see higher membership values for DifFUZZY, which coincide with the original memberships, shown in the colour bars above both plots.

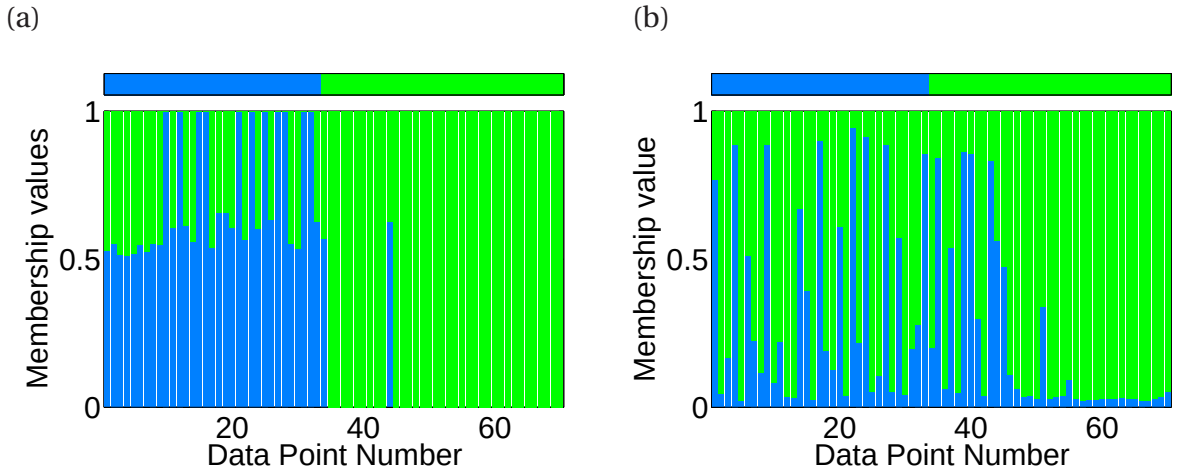


Figure 4.5: DifFUZZY and FCM applied to the *Leukemia* dataset. (a) DifFUZZY membership values, M = 11. (b) FCM membership values, $q = 2$, $K = 2$. The colour bars above show the real membership values of the data points.

According to Fig. 4.6 DifFUZZY slightly outperforms FCM on the *Leukemia* dataset, since for every false positive rate DifFUZZY presents a higher or equal true positive rate. This example also demonstrates how our method can handle high dimensional microarray data for multi-class cancer classification.

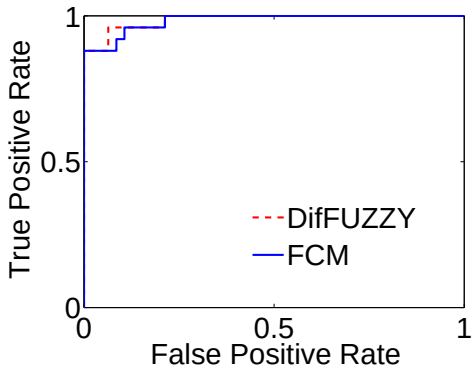


Figure 4.6: DifFUZZY and FCM ROC curves for the *Leukemia* dataset, M = 11, $q = 2$, $K = 2$.

4.1.1.4 *Ruspini* dataset

Another widely known benchmark dataset is named *Ruspini*, which is a set of 75 data points in two dimensions, forming four clusters (Fig. 4.7, Table B.2). This dataset was originally designed to illustrate how boundary data points from compact clusters are misclassified when using averages in the classification process, such as K-means representation of the cluster centroids, and it was suggested that this can be solved by *diluting* the clusters [Ruspini, 1970], or grouping data points according to their local neighbourhoods instead of distance to cluster centroids, which can be attained using diffusion distances.

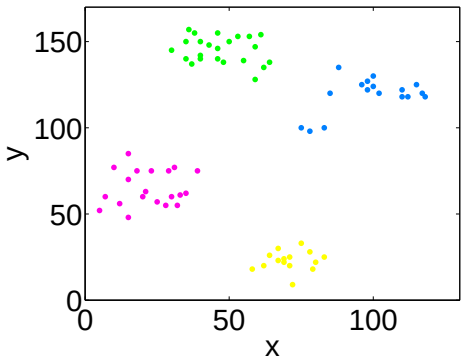


Figure 4.7: *Ruspini* dataset. The different colours represent the reference clusters.

In Fig. 4.7 we can observe that the clusters are well-separated and in principle it should not be a problem for most clustering techniques to resolve the different clusters. The full dataset is included in Appendix B.5.1. In fact, both DifFUZZY and FCM identify the four

clusters, as shown in Fig. 4.8. DifFUZZY identifies all the data points as cluster cores, whereas FCM results include fuzzier results, and a few data points are not fully assigned to their clusters for the threshold $z = 0.9$.

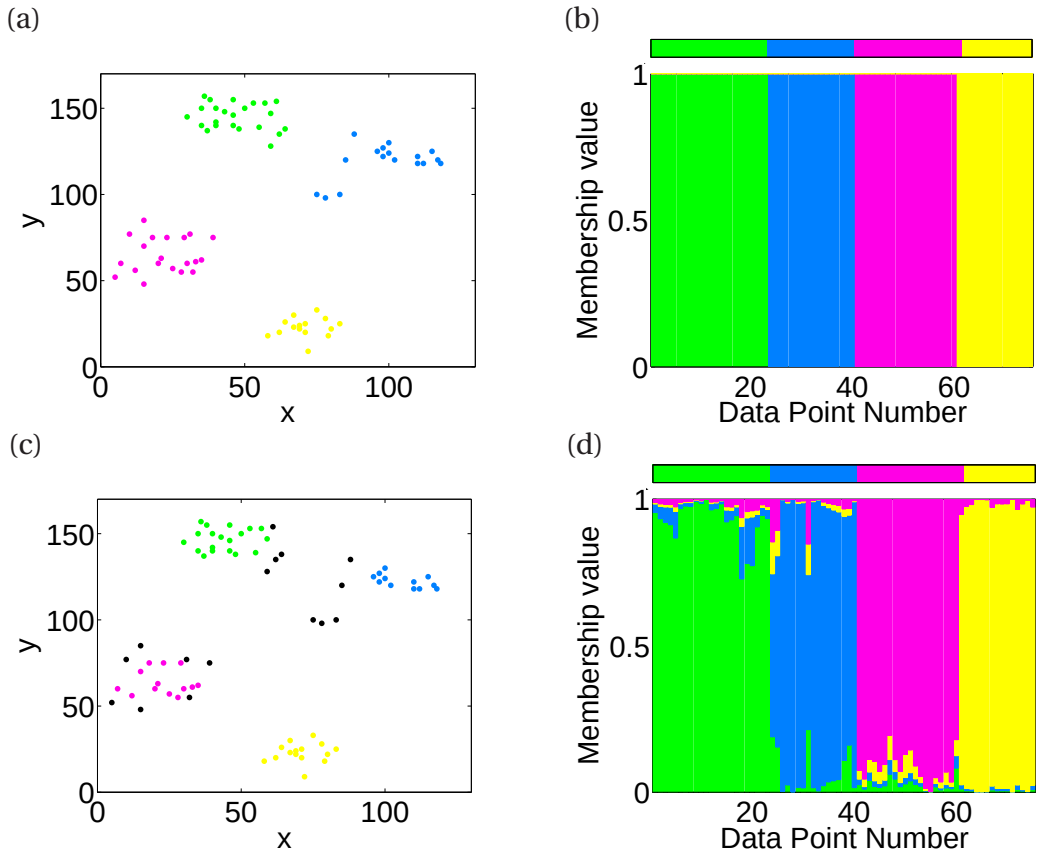


Figure 4.8: DifFUZZY and FCM on the *Ruspini* dataset. (a) DifFUZZY HCT-plot, $z = 0.9$, M = 15. (b) DifFUZZY membership values, M = 15. (c) FCM HCT-plot, $z = 0.9$, $q = 2$, $K = 4$. (d) FCM membership values, $q = 2$, $K = 4$. Colour code: green, red and blue correspond to the membership function of the data points in the three clusters, with the corresponding colour code as in (a). The colour bars above (b) and (d) show the real membership values of the data points.

4.1.1.5 *WDBC* dataset

The Wisconsin Diagnostic Breast Cancer (*WDBC*) dataset[2] is a multidimensional set of 569 data points, each having 32 real values describing the features of cell nuclei from medical images obtained from breast mass non-invasive fine needle aspiration. The nuclei data are classified as malignant or benign, and the real-valued variables digitally computed for

each nucleus include, among others, area, perimeter, smoothness, compactness, concavity, radius, symmetry, fractal dimension, number of concave portions of the contour and standard deviation of gray-scale values [Street et al., 1993].

These data were originally obtained to automate the diagnosis of malignant breast tumours, to avoid the subjectivity of and reliance on the expertise of the pathologist to assess tumour malignancy. Street et al. [1993] found that by comparing nuclear size, shape and texture (through the 32 variables in their dataset) they could accurately identify malignant from benign tumour cells.

Fig. 4.9 shows the MV plots for DifFUZZY and FCM over the *WDBC* dataset, where we can observe how FCM misassigns a larger number of data points from the green cluster into the blue cluster.

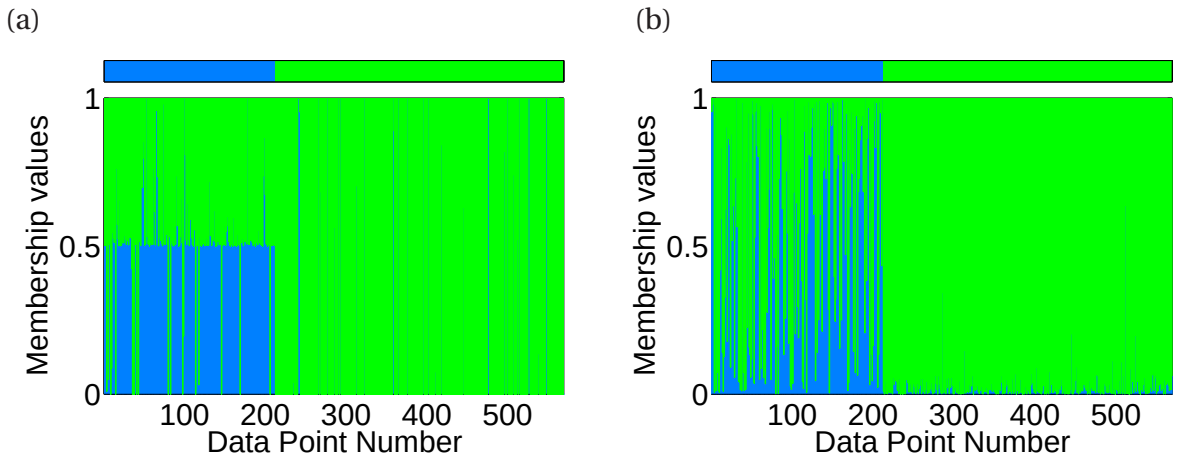


Figure 4.9: (a) DifFUZZY membership values for the *WDBC* dataset, M = 25. (b) FCM membership values, $q = 2$, $K = 2$. The colour bars above show the real membership values of the data points.

4.1.1.6 *Wine* dataset

The *Wine* dataset, consisting of 178 data points described by 18 variables, is another popular dataset[2] to test and illustrate clustering methods. The data are the result of a chemical analysis of wines whose provenance is from three different cultivars (Nebbiolo, Barberas

and Grignolino grapes) all grown in the Italian region of Piedmont [Aeberhard et al., 1992].

The variables are quantities such as the amount/concentration of phenols, flavonoids (both are key elements for the taste of wine) and magnesium in the wine, as well as the colour intensity, hue and the alkalinity of ash[4]. Another variable measures the ratio between absorbances at 280 and 315 nm of diluted wines (OD280/OD315) [Aeberhard et al., 1992; Wehrens, 2008].

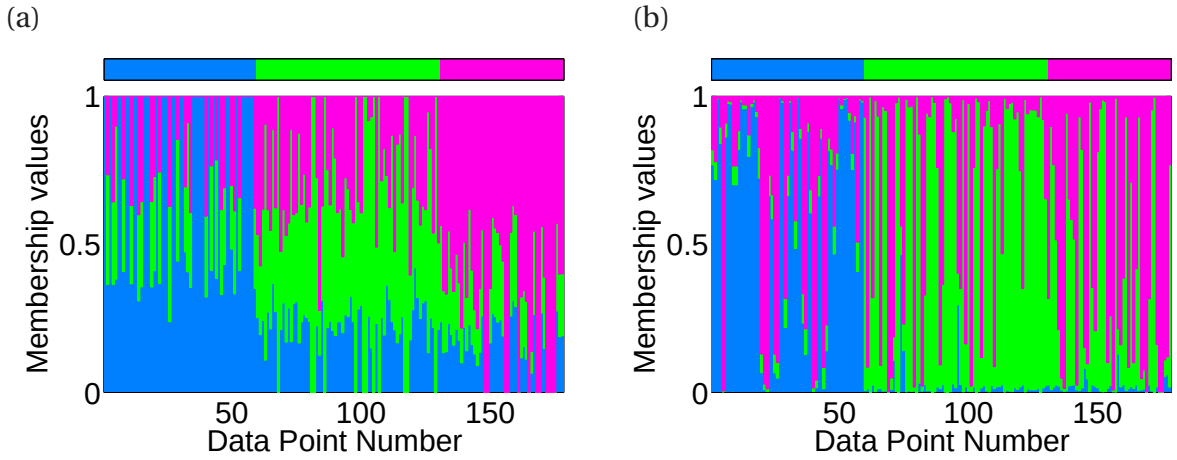


Figure 4.10: (a) DifFUZZY membership values for the *Wine* dataset, M = 14. (b) FCM membership values, $q = 2$, $K = 3$. The colour bars above show the real membership values of the data points, blue: Nebbiolo, green: Barberas and red: Grignolino grape cultivars.

As we observe in Fig. 4.10, neither FCM nor DifFUZZY can identify the three clusters perfectly. However, DifFUZZY's cluster cores coincide with the true memberships and FCM clusters several data points from the first cultivar together with the third one (red bars). In Appendix B.5.2 we include the ROC curves for the three cultivars (Fig. B.7) where it is clear to see the better performance of DifFUZZY versus FCM for each of the three cultivars. In Table 4.2 we present the classification errors of DifFUZZY, FCM and the previously presented clustering algorithms over this dataset. DifFUZZY's classification error (22.72%) is the lowest among all the algorithms.

[4] The ash content of wine is defined as "all the products (inorganic matter) remaining after igniting the residue left from the evaporation of must or wine" [Kosmerl and Bavcar, 2003].

4.1.1.7 *Zoo* dataset

The *Zoo* dataset[2] is a dataset commonly used to test clustering algorithms. It has 101 data points (animals) forming 7 different clusters or categories (mammals, birds, reptiles, fish, amphibians, insects and invertebrates). These data points are described by 16 attributes, 15 of which are categorical or boolean, (hair, feathers, eggs, milk, airborne, aquatic, predator, toothed, backbone, breathes, venomous, fins, tail, domestic, and size similar to a cat), and one is numerical (number of legs). An extract from this dataset is presented in Table 4.1.

Name	Type	Hair	Eggs	Milk	Airborne	Predator	Toothed	Backbone
bass	fish	0	1	0	0	1	1	1
bear	mammal	1	0	1	0	1	1	1
boar	mammal	1	0	1	0	1	1	1
calf	mammal	1	0	1	0	0	1	1
kiwi	bird	0	1	0	0	1	1	1
carp	fish	0	1	0	0	0	1	1
wasp	insect	1	1	0	1	0	1	1
worm	invertebrate	0	1	0	0	0	1	1

Table 4.1: *Zoo* dataset (extract)

In Fig. 4.11 we can see that DifFUZZY successfully clusters the animals into their original categories, identifying all birds, fish, and insects perfectly. On the contrary, FCM can only distinguish the fish cluster, and without high membership values for the elements in this cluster (Fig. 4.11 (b)). For the rest of the animals FCM does a poor job in identifying clusters and separating the different animal classes, whereas DifFUZZY performs remarkably well, even giving us further information about the dataset. From the membership values plotted in Fig. 4.11 (a) we see that DifFUZZY identifies three sub-clusters within the cluster of mammals: the light green cluster, the dark green and a small multicolour cluster. The former corresponds to 4-legged mammals, the next one represents two-legged mammals,

and the latter includes aquatic mammals (the dolphin, porpoise, sea lion and seal), four animals which share similarities with fish, mammals and other animal-types. These sub-clusters of mammals are depicted in Fig. 4.12 (a).

In the case of the invertebrates, represented in Fig. 4.12 (b), we see three clusters of animals clustered by DifFUZZY, the ones with dark blue membership value (invertebrates with legs and aquatic), the red ones (invertebrates without legs and mostly non-aquatic) and the multicoloured ones, which correspond to two special animals: the scorpion and the octopus, which differ from the rest of the animals in this category in not laying eggs (the scorpion) and having a very large size in comparison with the other invertebrates (the octopus). They also share other characteristics with animals in other clusters.

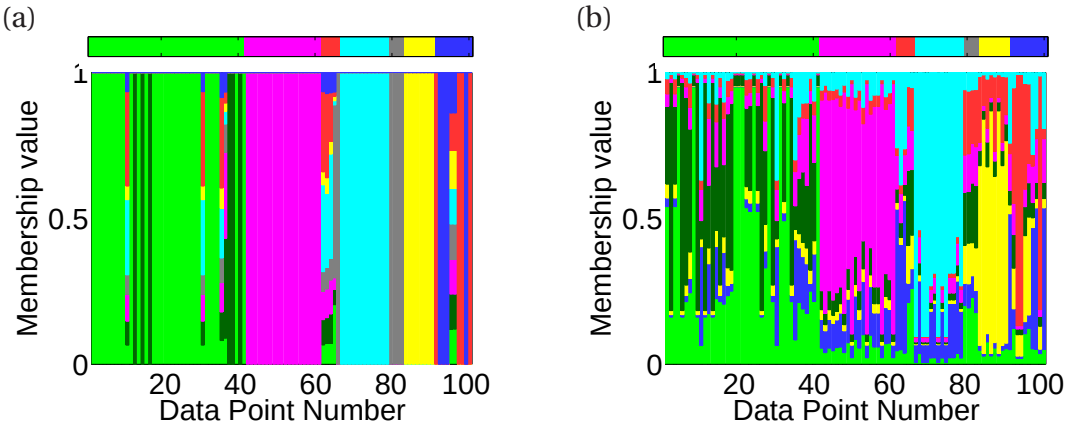


Figure 4.11: DifFUZZY and FCM on the *Zoo* dataset. (a) DifFUZZY membership values, M = 4. (b) FCM membership values, $q = 2$, $K = 7$. Different colours correspond to different clusters (green: mammals, purple: birds, red: reptiles, light blue: fish, gray: amphibians, yellow: insects and dark blue: invertebrates).

Finally, the reptiles (Fig. 4.12 (c)) are considered to have similarities with fish, invertebrates, mammals and birds. In particular, the tortoise is closer to the amphibians and four-legged mammals than other clusters (hence the green and gray colour in its frame in Fig. 4.12 (b)), and the tuatara[5] is assigned to the amphibians, which, given there was not a single cluster of reptiles identified, would be its closest family. Tuataras are indeed among the less evolved reptiles in the cluster, and share many attributes with amphibians.

[5] Reptile native to New Zealand which resembles the lizard but belongs to a different lineage.

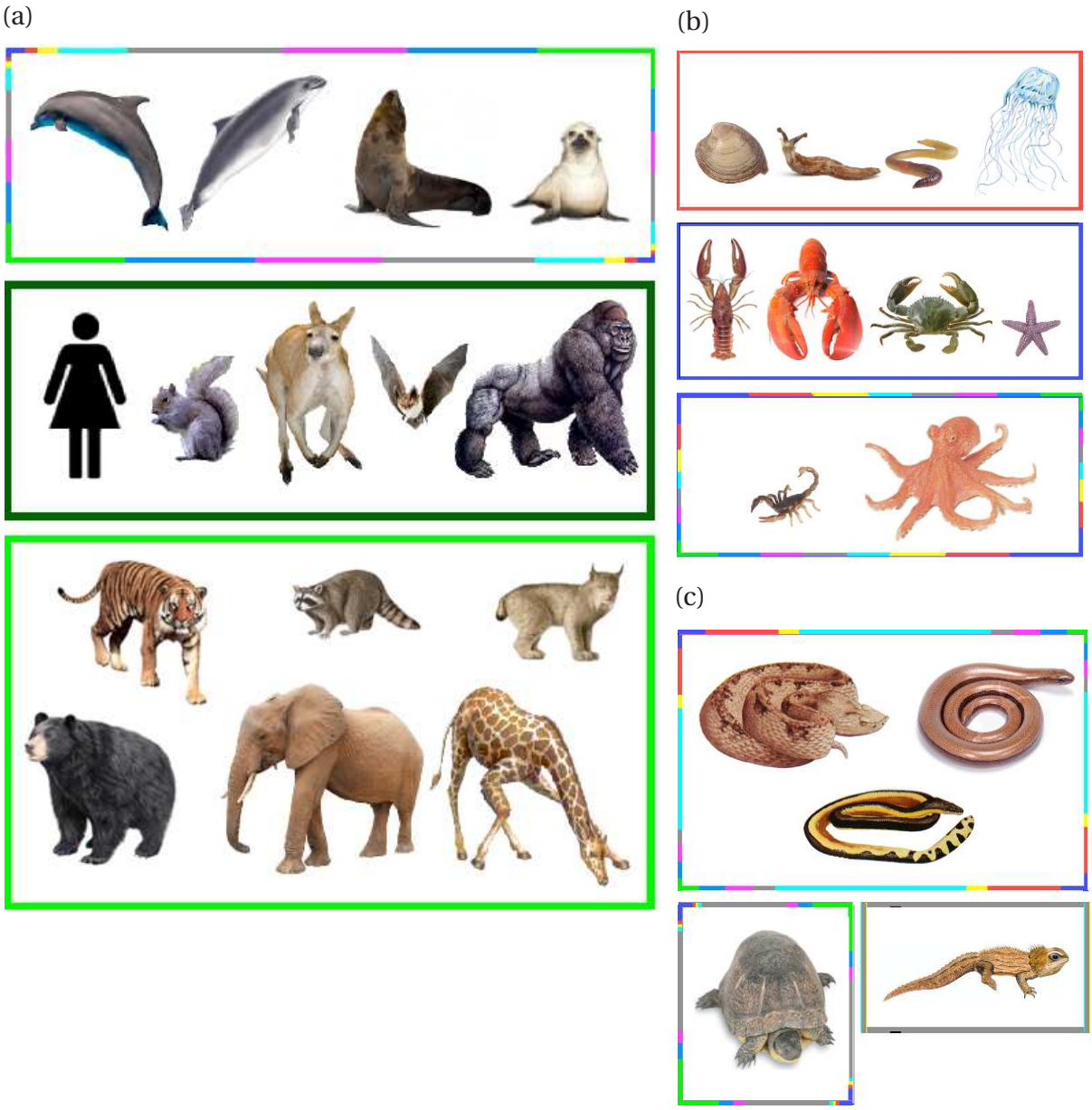


Figure 4.12: DifFUZZY clusters from the *Zoo* dataset. (a) Mammals, (b) invertebrates and (c) reptiles. The colour patterns of the frames represent the membership values in the different clusters, as in Fig. 4.11.

4.1.2 Summary of results over benchmark datasets

In order to compare DifFUZZY (soft) with the other clustering methods presented in Chapter 2.1 (both hard and soft) using the benchmark datasets previously introduced, we chose to use the intuitive classification error measure, defined in Section 2.2.4.1, given that these datasets are composed of known hard clusters and the true clustering is known. Using this measure we can see that soft methods such as DifFUZZY can also be used to successfully solve real non-fuzzy clustering problems.

Table 4.2 shows the summary of the classification results over the benchmark datasets described in Section 4.1. We can see that DifFUZZY's classification error is the smallest (or second smallest in the case of the *CRC* dataset) among all the clustering algorithms. For the *Iris* and *Wine* datasets the difference between DifFUZZY's classification error and that of the second best algorithm is considerably larger than for the other datasets.

It is worth noting that for some clustering algorithms the final partition is, on average, worse than doing the classification by random assignment. This is the case for HC single linkage over the *CRC* dataset, and HC complete linkage over the *Leukemia* dataset.

These benchmark datasets have become widely used because they are complex and/or difficult to cluster, so it should not be surprising to see the poor performance of K-means clustering.

Other interesting findings using these datasets are that PCM underperforms compared to the rest of the algorithms and different linkage approaches have a major impact on how HC performs (complete versus single linkage in the *CRC* and *Iris* datasets). Additionally, SOM yields intermediate results while spectral clustering yields results better than SOM, and closer to DifFUZZY's performance.

For the *Zoo* dataset a small change in the fuzziness parameter q can drastically affect the performance of FCM (classification error of 29.68 versus 42.49%). The same observation is true for PCM for the *Zoo* dataset for $q = 2.0$ versus $q = 2.5$.

	A	B	C	D	E	F	G
	CRC	*Iris*	*Leukemia*	*Ruspini*	*WDBC*	*Wine*	*Zoo*
No. of objects	62	150	72	75	569	178	101
No. of variables	370	4	70	2	32	13	16
No. of clusters	2	3	2	4	2	3	7
Random	46.11	63.60	45.04	68.31	48.31	61.80	74.28
K-means	41.00	20.60	20.58	13.78	14.59	32.04	30.27
HC, single linkage	62.90	32.00	33.33	**0.00**	37.08	57.30	32.69
HC, average linkage	45.19	9.33	33.33	**0.00**	33.74	38.76	24.75
HC, complete linkage	**22.58**	16.00	52.78	**0.00**	33.74	32.58	24.75
FCM, $q = 1.5$	35.61	11.33	**13.89**	0.96	14.94	30.34	29.68
FCM, $q = 2.0$	34.66	10.67	**13.89**	**0.00**	14.59	31.46	42.49
FCM, $q = 2.5$	34.34	10.00	**13.89**	**0.00**	14.41	31.46	44.97
PCM, $q = 1.5$	35.48	47.00	34.72	0.76	41.83	58.43	32.14
PCM, $q = 2.0$	35.48	48.00	34.72	**0.00**	42.36	58.43	39.28
PCM, $q = 2.5$	35.48	46.00	34.72	**0.00**	40.07	58.99	70.30
SOM	40.01	11.47	16.10	3.53	13.81	32.71	26.21
Spectral	32.26	11.65	**13.89**	9.44	13.53	34.22	18.82
DifFUZZY	24.19	**2.67**	**13.89**	**0.00**	**10.72**	**22.72**	**17.82**

Table 4.2: Classification error (%) from 7 clustering benchmark datasets using different clustering methods and parameters for the same known number of clusters: K-means (random initialisation), Hierarchical clustering (Euclidean distance), FCM (random initialisation), PCM (random initialisation), SOM (Euclidean distance, hexagonal lattice with 15×15 neurons, 100 time steps, K-means, random initialisation, linear learning rate, Gaussian neighbourhood), Spectral ($\beta_A = 32$, $\beta_B = 2 \times 10^{-4}$, $\beta_C = 200$, $\beta_D = 2 \times 10^4$, $\beta_E = 12.5$, $\beta_F = 2 \times 10^{-4}$ and $\beta_G = 0.72$) and DifFUZZY ($M_A = 11$, $M_B = 14$, $M_C = 13$, $M_D = 15$, $M_E = 25$, $M_F = 12$ and $M_G = 4$). We also include the classification error of random assignment, this is randomly choosing one of the K clusters for each data point (as illustrative examples we show results of random assignment in test datasets in Fig. B.8, Appendix B.5.3). The results are the average over 100 realisations. In bold we show the best clustering results for each dataset.

4.2 Synthetic test datasets

In this section we study how DifFUZZY deals with a number of synthetic datasets for which FCM succeeds (dataset comprised of globular clusters both in 2- and 3-D) and for others where it does not (datasets non-linearly separable such as the *Half moons* and *Interlaced rings* datasets). We compare both soft techniques using HCT-plots.

4.2.1 *Globular clusters* dataset

In Fig. 4.13 (a) we present the results obtained by applying DifFUZZY to the *Globular clusters* dataset, previously shown in Fig. 3.3 (a).

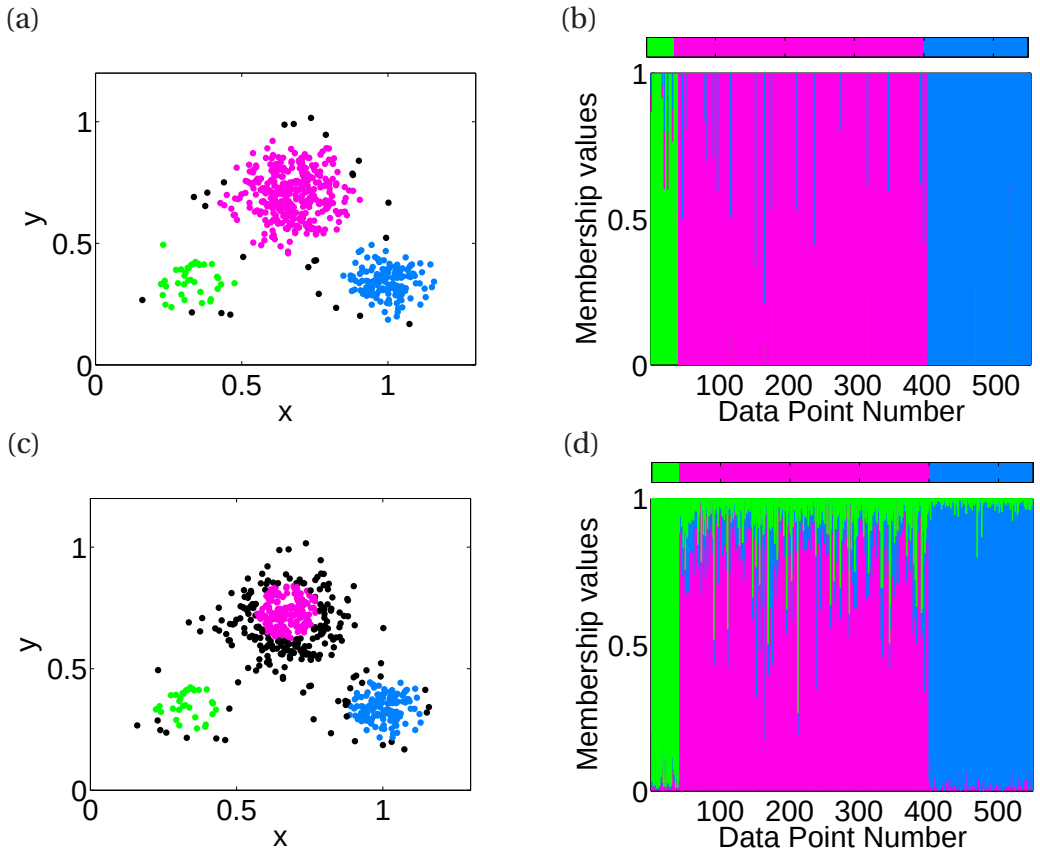


Figure 4.13: DifFUZZY and FCM on the *Globular clusters* dataset. (a) DifFUZZY HCT-plot, $z = 0.9$, M $= 35$. (b) DifFUZZY membership values, M $= 35$. (c) FCM HCT-plot, $z = 0.9$, $q = 2$, $K = 3$. (d) FCM membership values, $q = 2$, $K = 3$. Colour code: green, red and blue correspond to the membership function of the data points in the three clusters, with the corresponding colour code as in (a). The colour bars above (b) and (d) show the real membership values of the data points.

In Fig. 4.13 we can see that the data points form three well defined clusters. Any soft or hard standard clustering technique should identify these clusters. However, if we introduce intermediate data points, the clusters are less well defined and some hard clustering techniques may have difficulty in separating the clusters. FCM can successfully handle this problem (Fig. 4.13 (b)). The same is true for DifFUZZY.

4.2.2 *Concentric rings* dataset

A classical example where the K-means algorithm, presented in Section 2.1.1.1, fails (see Fig. 2.3) [Filippone et al., 2008] is shown in Fig. 4.14 (a). This is a two-dimensional dataset formed by three concentric rings, similar to the one used by a number of authors such as Guha et al. [2001] and Ben-Hur et al. [2002].

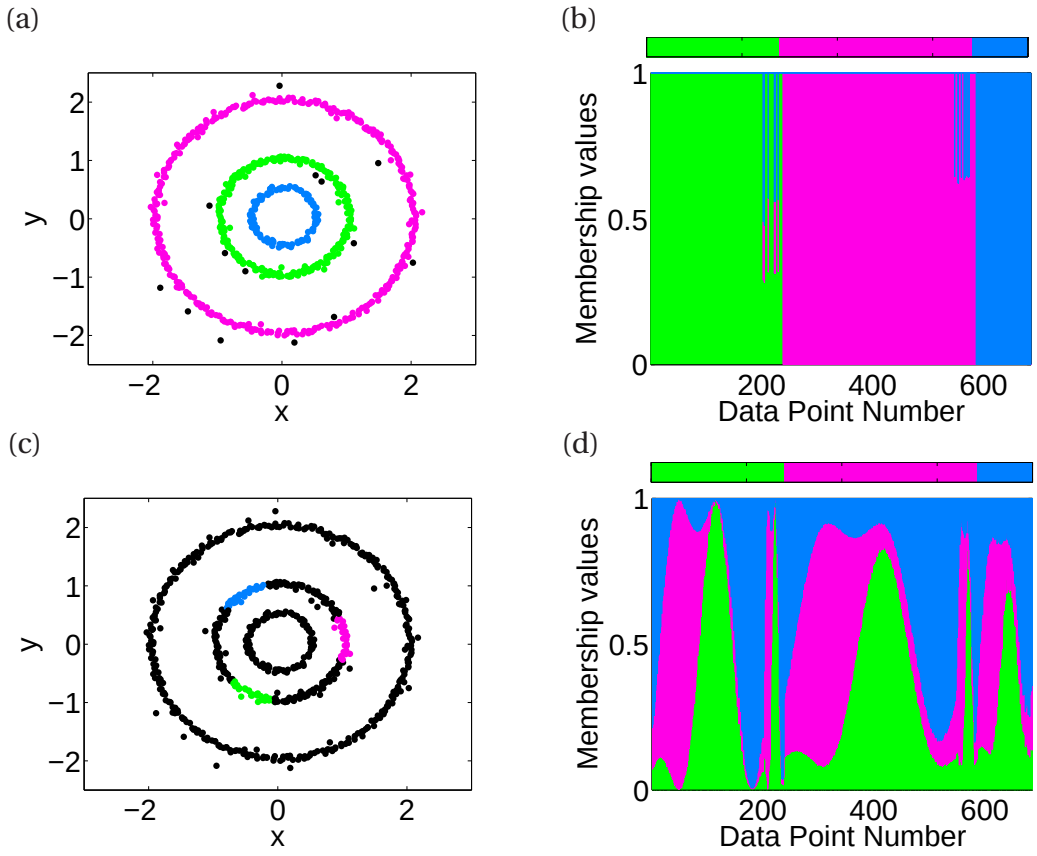


Figure 4.14: DifFUZZY and FCM on the *Concentric rings* dataset. (a) DifFUZZY HCT-plot, $z = 0.9$, M = 90. (b) DifFUZZY membership values, M = 90. (c) FCM HCT-plot, $z = 0.9$, $q = 2$, $K = 3$. (d) FCM membership values, $q = 2$, $K = 3$. Colour code for (a) and (c) is the same as in Fig. 3.3 (a). Colour code for (b) and (d) is the same as for Fig. 3.3 (b).

Using DifFUZZY we identify each ring as a separate cluster, as can be seen in Fig. 4.14 (a)-(b). The membership values obtained with FCM are plotted in Fig. 4.14 (d). In Fig. 4.14 (c) we present the corresponding HCT-plot with a threshold value of 0.9. Comparing Figs. 4.14 (a)-(b) with 4.14 (c)-(d) we clearly see that DifFUZZY gives better results than FCM.

4.2.3 *Elongated clusters* dataset

Another dataset where the K-means algorithm fails is presented in Fig. 4.15 (a) (Fig. 2.4 shows K-means clustering results when applied to the *Elongated clusters* dataset). This two-dimensional dataset contains two elongated clusters, one in a diagonal orientation and the other in a cross-shaped position.

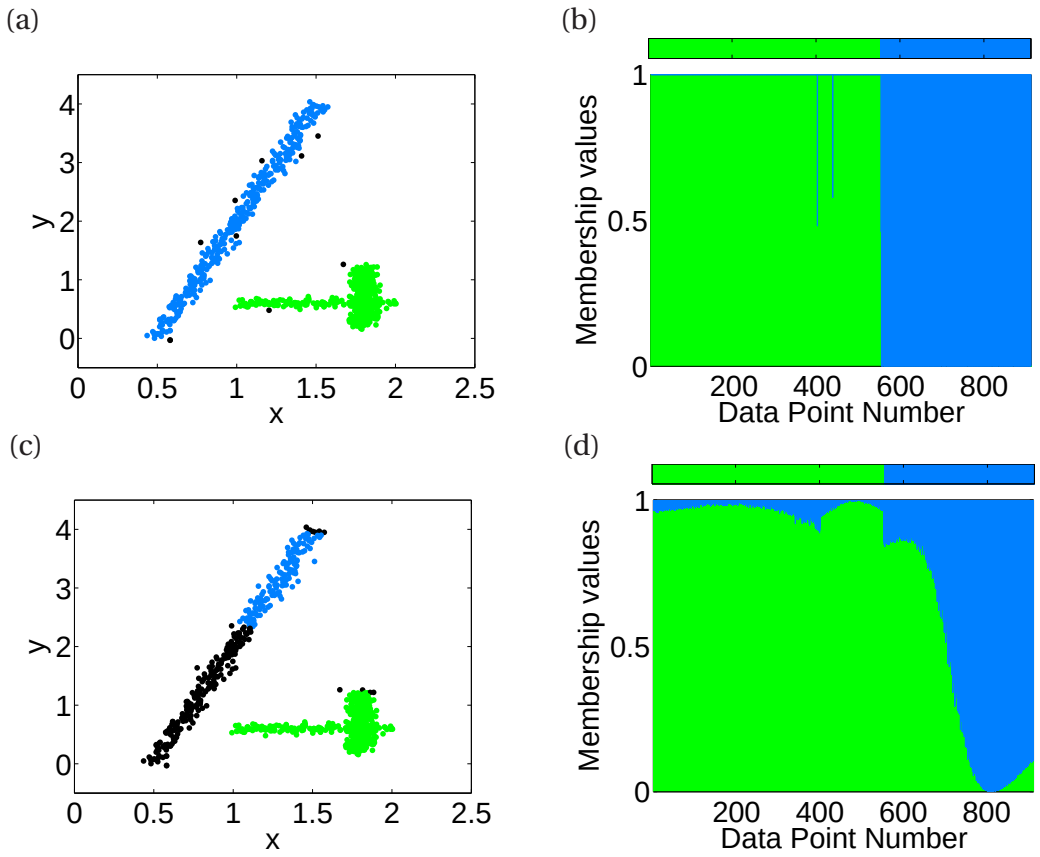


Figure 4.15: DifFUZZY and FCM on the *Elongated clusters* dataset. (a) DifFUZZY HCT-plot, $z = 0.9$, M = 300. (b) DifFUZZY membership values, M = 300. (c) FCM HCT-plot, $z = 0.9$, $q = 2$, $K = 2$. (d) FCM membership values, $q = 2$, $K = 2$. Colour code for (b) and (d) is the same as for Fig. 3.3 (b).

The results of DifFUZZY and FCM applied over this dataset are summarised in the membership value plots in Figs. 4.15 (b) and (d), respectively. DifFUZZY can separate the clusters remarkably well, as is clear from Fig. 4.15 (a). For this dataset FCM can not separate the clusters, cutting the left cluster (blue) into two parts as can be seen in the HCT-plot in Fig. 4.15 (c), using the threshold value $z = 0.9$. If we compare the membership values given by FCM (Fig. 4.15 (d)) to the one by DifFUZZY in Fig. 4.15 (b), which basically corresponds to the desired membership values of the data points, we see the incorrect assignment of the data points numbered around 550-700, which in the case of FCM have been given a higher membership in the green cluster than in the cluster to which they originally belong.

4.2.4 *3-D globular clusters* dataset

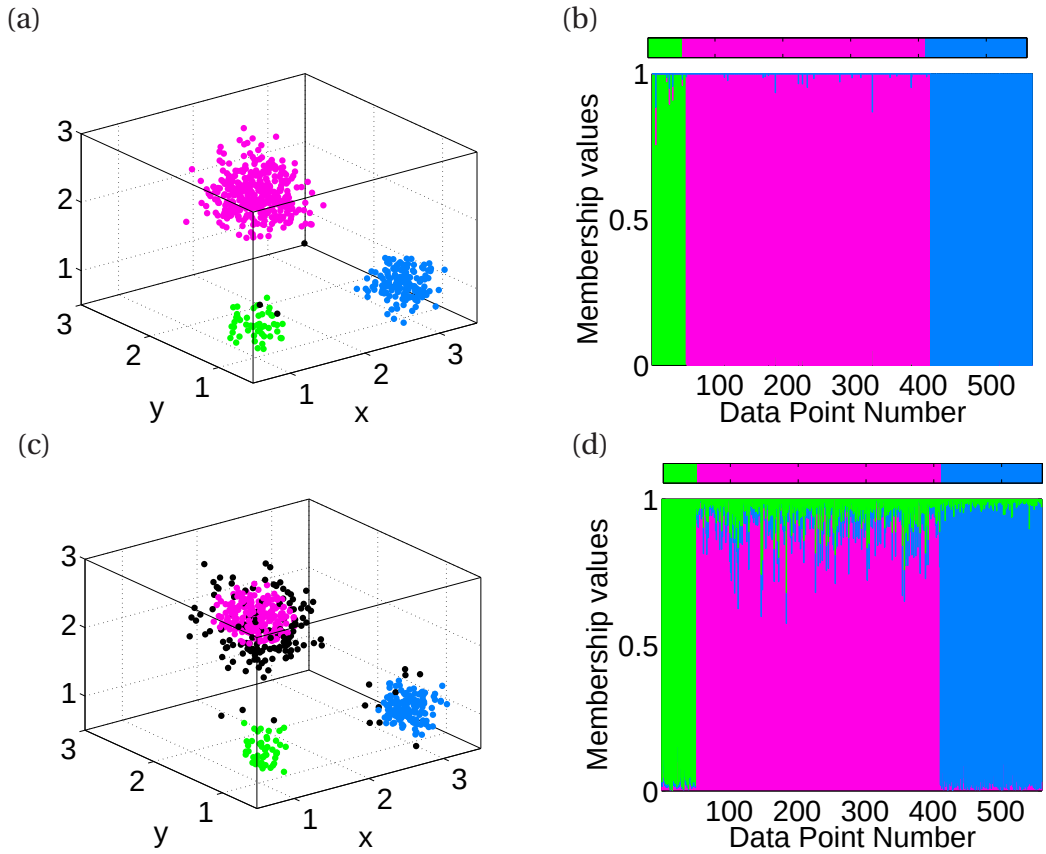


Figure 4.16: DifFUZZY and FCM on the *3-D globular clusters* dataset. (a) DifFUZZY HCT plot, $z = 0.9$, M = 48. (b) DifFUZZY membership values, M = 48. (c) FCM HCT-plot, $z = 0.9$, $q = 2$, $K = 3$. (d) FCM membership values, $q = 2$, $K = 3$. Colour code as in Fig. 3.3 (b).

A 3-D version of the dataset shown in Fig. 2.1 (a) is presented in Fig. 4.16.

Just like with the 2-D *Globular clusters* dataset, we observe that both FCM and DifFUZZY do a very good job in identifying the three dimensional globular clusters. Furthermore, we can see that DifFUZZY gives higher membership values, in particular for data points in the red dataset (see Figs. 4.16 (b) versus (d)), assigning the data points with a higher confidence than FCM or, in other words, giving larger clusters for a given threshold.

4.2.5 *Half moons* dataset

The *Half moons* dataset is another synthetic dataset we generated to test DifFUZZY against FCM, similar to the one used by other authors such as Hammouche et al. [2006] and Hocking et al. [2011].

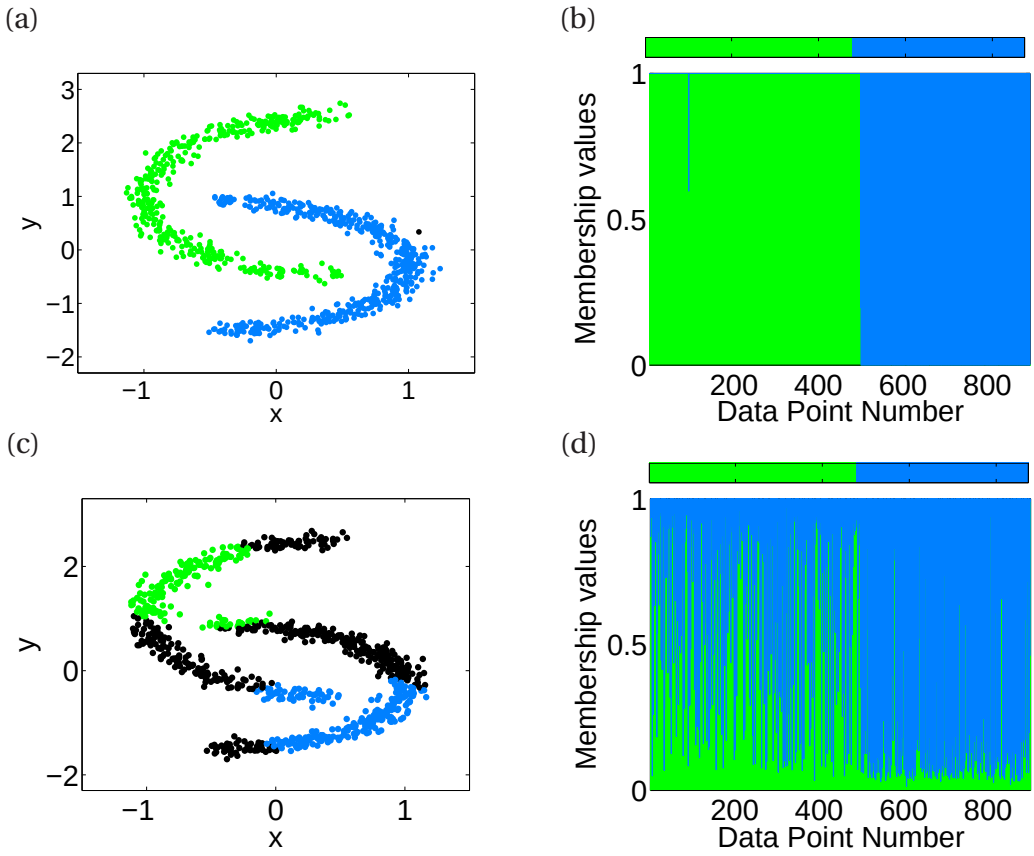


Figure 4.17: DifFUZZY and FCM on the *Half moons* dataset. (a) DifFUZZY HCT-plot, $z = 0.9$, M $= 300$. (b) DifFUZZY membership values, M $= 300$. (c) FCM HCT-plot, $z = 0.9$, $q = 2$, $K = 2$. (d) FCM membership values, $q = 2$, $K = 2$.

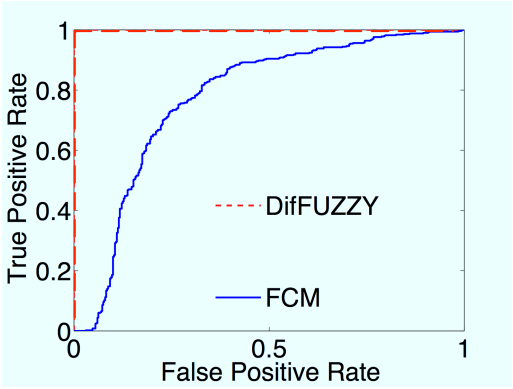


Figure 4.18: DifFUZZY and FCM ROC curves for the *Half moons* dataset (M = 350).

This complex dataset represents a challenge for several clustering methods and in Fig. 4.17 we show that DifFUZZY can identify both clusters remarkably well, whereas FCM misclassifies a large number of data points.

We can also observe this outperformance of DifFUZZY over FCM by the ROC curves in Fig. 4.18. There we can see that for a given FPR the TPR is higher in the results obtained with DifFUZZY than with FCM.

4.2.6 *Interlaced rings* dataset

We generated a 3-D complex dataset with two interlaced ring-shaped clusters similar to the one used by Hammouche et al. [2006] and Newman and Cooper [2010], among many others, which we called the *Interlaced rings* dataset. This dataset presents serious difficulty for clustering methods such as FCM (see Figs. 4.19 (c) and (d), where data points from different rings cluster together forming two clusters), while DifFUZZY separates both clusters and gives intermediate membership values to those data points which are further away from the cores of the rings, illustrated in black in Fig. 4.19 (a).

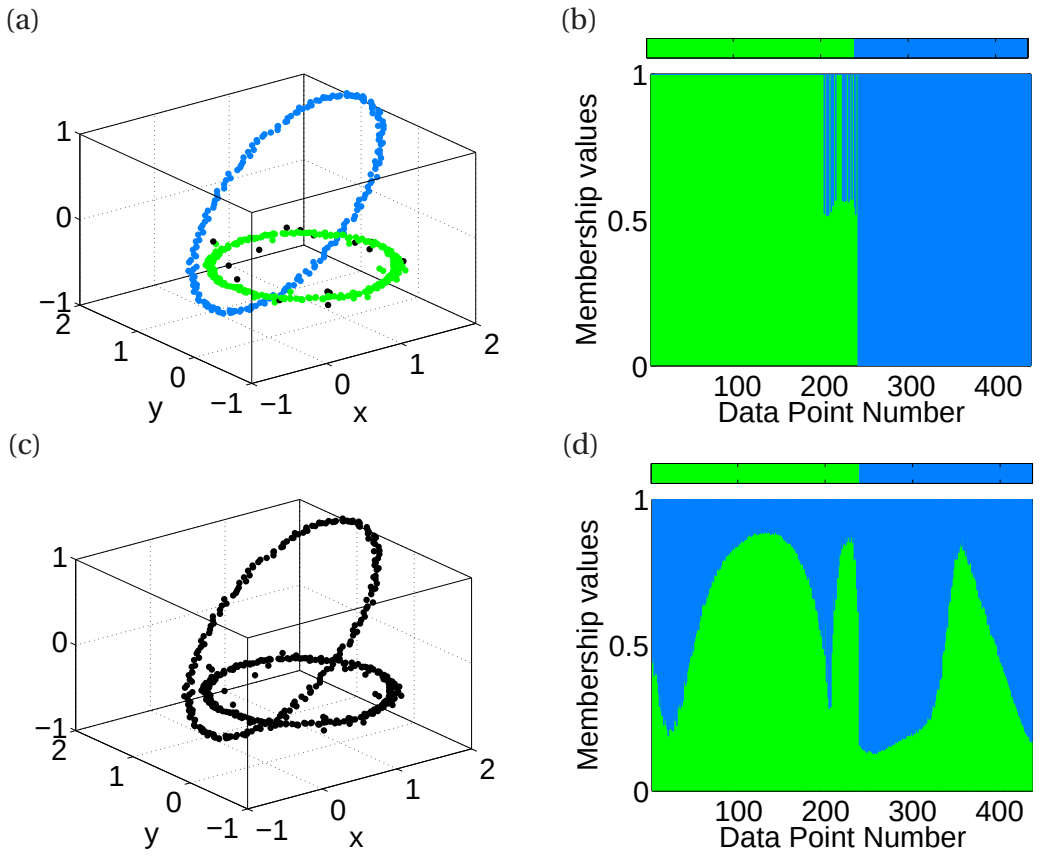


Figure 4.19: DifFUZZY and FCM on the *Interlaced rings* dataset. (a) DifFUZZY HCT-plot, $z = 0.9$, M $= 180$. (b) DifFUZZY membership values, M $= 180$. (c) FCM HCT-plot, $z = 0.9$, $q = 2$, $K = 2$. (d) FCM membership values for this dataset, $q = 2$, $K = 2$. Colour code for (a) and (c) is the same as in Fig. 2.1 (a). In black are the soft data points. Colour code for (b) and (d) as in Fig. 2.1 (b).

We have looked at synthetic datasets and while they are extreme and most likely unrealistic, they illustrate that datasets with certain geometrical properties will present problems to standard clustering algorithms which can be overcome by DifFUZZY. Next we consider a dataset emerging from simulation of a system of molecular reactions.

4.2.7 *Stochastic dynamical system* dataset

The data used in this example were obtained from one realization of the Gillespie stochastic simulation algorithm (SSA) for the following system of reactions [Erban et al., 2007a]:

$$2A \underset{k_2}{\overset{k_1}{\rightleftarrows}} 3A, \qquad \emptyset \underset{k_4}{\overset{k_3}{\rightleftarrows}} A, \tag{A.1}$$

where A represents chemical species and $k_1 = 0.18$ min^{-1}, $k_2 = 2.5 \times 10^{-4}$ min^{-1}, $k_3 = 2200$ min^{-1} and $k_4 = 37.5$ min^{-1} are the rate constants [Erban et al., 2007a]. The number of molecules stochastically switches in time between the two steady state values (100 and 400, which can be seen in a realisation of the stochastic dynamical system in Fig. 4.20) and the amplitude of the fluctuations around the larger steady state is greater on average than that of the fluctuations around the lower steady state.

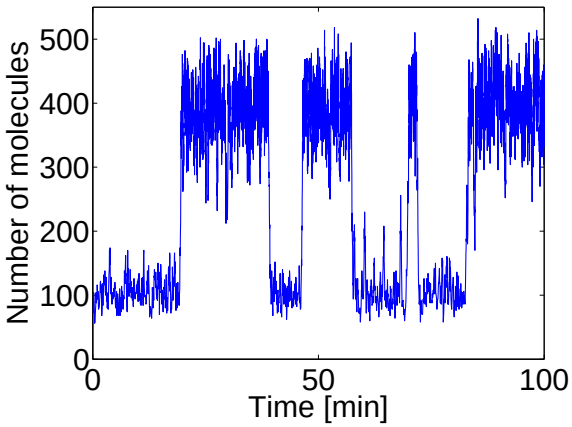


Figure 4.20: One realisation of the SSA for the system of chemical reactions in Eq. (A.1). This figure shows the time evolution over 100 minutes [Erban et al., 2007a], and both steady states of 100 and 400 number of molecules can be identified. The basic Gillespie SSA consists of 4 main steps: (1) generation of uniformly distribute random numbers; (2) computation of propensity functions that give the probability of each reaction taking place between times t and t+Δt; (3) computation of the time τ when the next chemical reaction will occur, using one of the random numbers obtained in step (1); (4) computation of which reaction will occur at time t+τ, updating of the number and reactants of the computed reaction and iterate from step (1) with time t+τ.

Random switching between steady states, as that observed in these data, can be also found in gene regulatory networks, and can not always be described using deterministic models such as differential equations, which is why stochastic simulations are better suited for modelling some types of complex data [Erban et al., 2007a].

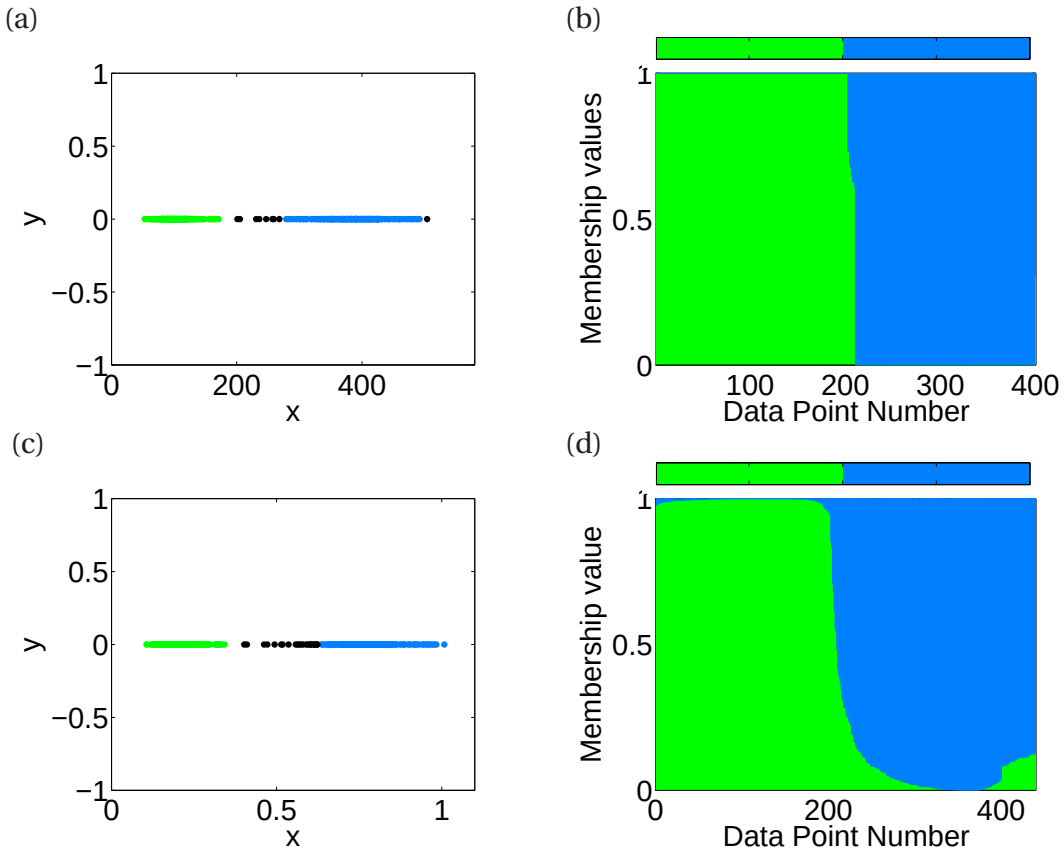


Figure 4.21: DifFUZZY and FCM on the *Stochastic dynamical system* dataset simulated from reaction scheme (A.1). In this 1-D dataset x values represent the sampled number of molecules. (a) DifFUZZY HCT-plot, $z = 0.9$, M = 180. (b) DifFUZZY membership values, M = 180. (c) FCM HCT-plot, $z = 0.9$, $q = 2$, $K = 2$. (d) FCM membership values, $q = 2$, $K = 2$. In black are the soft data points.

Applying DifFUZZY and FCM to the data generated we can see that while both clustering methods perform well and can roughly identify both clusters, DifFUZZY is better at classifying the data points on the border of the clusters and in the far right of the x-axis (Fig. 4.21 (a)), assigning to them a high membership value in the blue cluster.

4.3 Discussion

While it is true that higher membership values for soft data points do not necessarily mean a better result than a lower membership function or higher fuzziness of the points, "good" clusters are generally crispier. The goal of clustering is to identify clusters within the data, and highly fuzzy sets of points do not help to identify those clusters. This is why DifFUZZY distinguishes cluster cores from soft data points, giving the possibility of identifying regions of well connected, dense data points, and allowing a soft membership to all the remaining data points that do not belong to any of these regions.

When comparing benchmark datasets for which we know the true hard clusters, crispier results are better than fuzzier ones. This is why we chose to compare the results of the methods using the hard assignments of the soft partitions obtained using FCM, PCM and DifFUZZY and why we compared the MV plots against the true membership values in the form of colour bars.

We have demonstrated that DifFUZZY gives results superior to those from FCM, based on their contrasting outcomes when applied to non-linear datasets. For some of these datasets, which are shown in Figs. 4.14 (c) and (d), FCM fails to identify clusters in the data, whereas DifFUZZY identifies the three main clusters and assigns high membership functions to the borderline data points. This remarkable performance was expected since DifFUZZY was formulated so as to handle "curved" datasets.

DifFUZZY performs satisfactorily for a number of datasets, with sizes ranging from tens to hundreds of data points of dimensions as high as hundreds. For example, DifFUZZY competently handles microarray data, where a typical dataset can have dozens or even hundreds of points (number of samples, conditions or patients in medical applications) and its dimension ranges from hundreds to several thousands (number of genes on the chip) [Quackenbush, 2001]. It is worth mentioning that the dimension d of the data points in Eq. (3.1) does not represent a bottleneck for our method, since it is used only one time

when computing the pairwise distances. The number of data points, N, determines the dimension of the matrices (i.e. computational intensity). In genetic expression data N is often smaller than the value of d.

We can classify the clustering methods tested into two broad categories, those which cluster according to *compactness* or, in other words, those which minimise the intra-cluster variation [Handl et al., 2005], such as K-means, SOMs, FCM and PCM, and those which cluster by *connectedness*, i.e. ensuring that neighbours are grouped together [Handl et al., 2005], such as HC, spectral clustering and DifFUZZY. However, upon comparison of performance across benchmark datasets, we see that in both groups there are methods which perform well (DifFUZZY and HC, average linkage over the *Iris* dataset, HC, complete linkage and DifFUZZY over the *CRC* dataset and DifFUZZY, spectral and FCM over the *Leukemia* dataset, to name some) and others that perform poorly. In fact, some methods perform so poorly that on average they do worse than randomly assigning each data point to a cluster (HC, single linkage over the *CRC* dataset).

One of the issues that was briefly discussed was the pre-processing of data, a crucial step for some clustering applications. Noisiness and scale of a given dataset can have a high impact on the output of clustering procedures [Kim et al., 2006; Karthikeyani and Thangavel, 2009]. The choice of normalisation scheme will depend on the characteristics of the data themselves. When additional information is available, this should be used to improve the quality of the input data. A good example of this is the genetic expression case shown in Fig. 4.6 (d), where filtering, thresholding, log normalisation and gene selection were some of the common pre-processing steps used to improve the input data [Tan et al., 2004]. Gene selection reduces the dimensionality of the feature space by discarding redundant information. Another option is to weight the different features in order to make the dimensions of different features comparable, or sometimes to augment the influence of data carrying more or better information about the data structure. The use of independent feature scaling has been described in the context of similarity matrices in Erban et al. [2007b], where a

single value of the β parameter in Eq. (3.3) is not necessarily appropriate for all the components (variables), given that these may vary over different orders of magnitude. Two common examples of natural weights are; to assign the same weight (hence equal influence) to the absolute values of each feature; and also to rescale each variable so they range between the same minimum and maximum values.

Using the *Zoo* dataset we showed that DifFUZZY can also handle binary data, given that it has clustered the animals in a biologically meaningful way, whereas FCM has failed dramatically in this task. We have to point out however, that the results of any clustering method depend on the quality of the data. If the attributes would have been more specific, then the identification of animal clusters could have been even more successful.

As a general observation we note that DifFUZZY makes use of the neighbourhood information of the data (through diffusion distances) and does not rely on the representation of the clusters by their centroids, as does K-means, FCM and similar clustering algorithms, which we have shown not to always be meaningful. On the contrary, for each soft data point it uses all data points via the diffusion process. This makes DifFUZZY more robust to outliers, as well as able to resolve complex geometries on the data.

Chapter 5

Clustering Clinical Data: Severe Malaria

In this following chapter we describe the use of DifFUZZY and other clustering and network techniques to analyse a clinical dataset of severe malaria affecting children from The Gambia.

5.1 Background

5.1.1 About severe malaria

Severe malaria (SM) is a major global health concern, causing high mortality and morbidity worldwide. Around 3.3 billion people are at risk of contracting malaria [WHO, 2011], although more than 90% of all life-threatening cases of malaria occur in African children [WHO, 2000]. Malaria is transmitted by parasite-infected anopheline mosquitoes. *Plasmodium falciparum* is the deadliest among the four species of the *Plasmodium* parasite that can cause malaria in humans.

Malaria parasites are transmitted through infective mosquito bites. The saliva of infected mosquitoes carries parasite sporozoites that target the human liver. The parasites develop and multiply in the liver for 3-6 weeks. Tens of thousands of parasites are then released into the bloodstream where they invade human red blood cells causing the classical symptoms of malaria: a flu-like syndrome with fever, headache and muscle pain [WHO, 2011]. The life-threatening form of malaria is what is known as severe malaria, which usually develops

as a result of delay of treatment or in patients who have not yet developed immunity like children and visitors from non-endemic areas [Maitland et al., 2005].

5.1.1.1 Clinical features of severe malaria in children

Patients with uncomplicated malaria generally present fever and other non-specific symptoms, but those with severe malaria show distinctive clinical features. Cough, breathing difficulties and hypoglycaemia are common clinical features of severe malaria in children. However, unlike adults, children with severe malaria rarely present jaundice, renal failure or pulmonary oedema [WHO, 2000; Mohan et al., 2008].

The World Health Organization (WHO) has determined standard criteria for identifying severe and complicated malaria. This standard is comprised of 10 defining clinical laboratory conditions (coma, severe malarial anaemia, respiratory distress or pulmonary oedema, hypoglycaemia, circulatory collapse, renal failure, spontaneous bleeding, repeated convulsions, acidosis and haemoglobinuria) along with 5 supporting criteria (impaired consciousness, jaundice, prostration, hyperpyrexia and hyperparasitaemia) [Anstey and Price, 2007].

In 1995, Marsh et al. identified three specific clinical subgroups (shown in Fig. 5.1) among 1844 Kenyan children with severe malaria, namely: impaired consciousness or cerebral malaria (CM), respiratory distress (RD) and severe malarial anaemia (SMA). These syndromes have distinct characteristic symptoms and require intensive clinical management along with anti-malarial drugs [Mohan et al., 2008]. Children presenting with SMA require blood transfusions to replace the red blood cells destroyed by the parasite and restore oxygen delivery to cells.

However, these syndromes can overlap and children can present with two or even all three of these syndromes. The presence of multiple syndromes is associated with increased mortality rates in children with SM [Marsh et al., 1995].

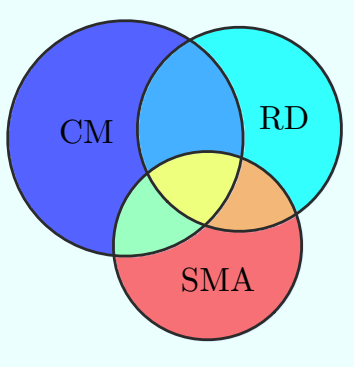


Figure 5.1: Severe malaria groups. The central yellow region represents children that exhibit all three syndromes, and is labelled as CMRDSMA. The orange RDSMA region is comprised of children with both RD and SMA syndromes. The green area contains children with both CM and SMA syndromes, and is labelled as CMSMA. The mid-blue is the CMRD region, comprised of children manifesting CM and RD syndromes. Finally, the complement of these groups is categorised as uncomplicated malaria. These colours will be used to denote each group and subgroup in the rest of this chapter.

5.1.2 About the data

The data described in this chapter were obtained from 2,915 Gambian children admitted to hospital from 1997 to 2008. The number of children with the different types of severe malaria syndromes and its combinations are presented in Table 5.1. Children who could not be classified into any of the classical SM syndromes (CM, RD or SMA) were classified as 'Others'.

	CM	CMRD	RD	CMSMA	CMRDSMA	RDSMA	SMA	Others	**Total**
Number of children	438	460	365	56	111	234	264	987 (34%)	**2915**

Table 5.1: Number of cases of different severe malaria syndromes in The Gambia dataset.

The data are described using 100 variables, many of which are redundant. Additionally, there are three variables of outcome (death, dsdeth, sequelae), 18 variables describing the

treatment given (such as quinine, chloquin, atsunate, diazepan) and a variable of case definition. A subset of the variables is given in Appendix C.1. Most of the variables are clinical symptoms such as number of fits, vomit, cough; some are laboratory results such as parasitaemia, lactate, anaemia, or hypoglycaemia, and the rest are demographic descriptors of the patients (age, weight, male/female, father/mother alive, ethnicity). Among these variables, the vast majority are binary, a few are categorical and the rest are continuous.

5.2 Methodology

The methodological workflow to analyse these data is outlined in Fig. 5.2. The steps, which will be explained in more detail in the following subsections, are:

Step 1 **Pre-process**. To decide how to deal with mixed data, clean the data (identify outliers, modify conflicting variable values and fix unrealistic values), treat missing values, and normalise the data.

Step 2 **Select most informative features**. To find a smaller subset of variables which is best to distinguish between different groups of data.

Step 3 **Obtain 'distances' between patients**. To calculate how similar patients are given the reduced set of variables previously selected and an appropriate distance measure.

Step 4 **Choose a distance threshold**. To determine a threshold below which data points will be linked in a network. Such a threshold should give an adequate degree of cluster densities.

Step 5 **Visualising the data**. Use an efficient layout algorithm to plot the network obtained from Step 4.

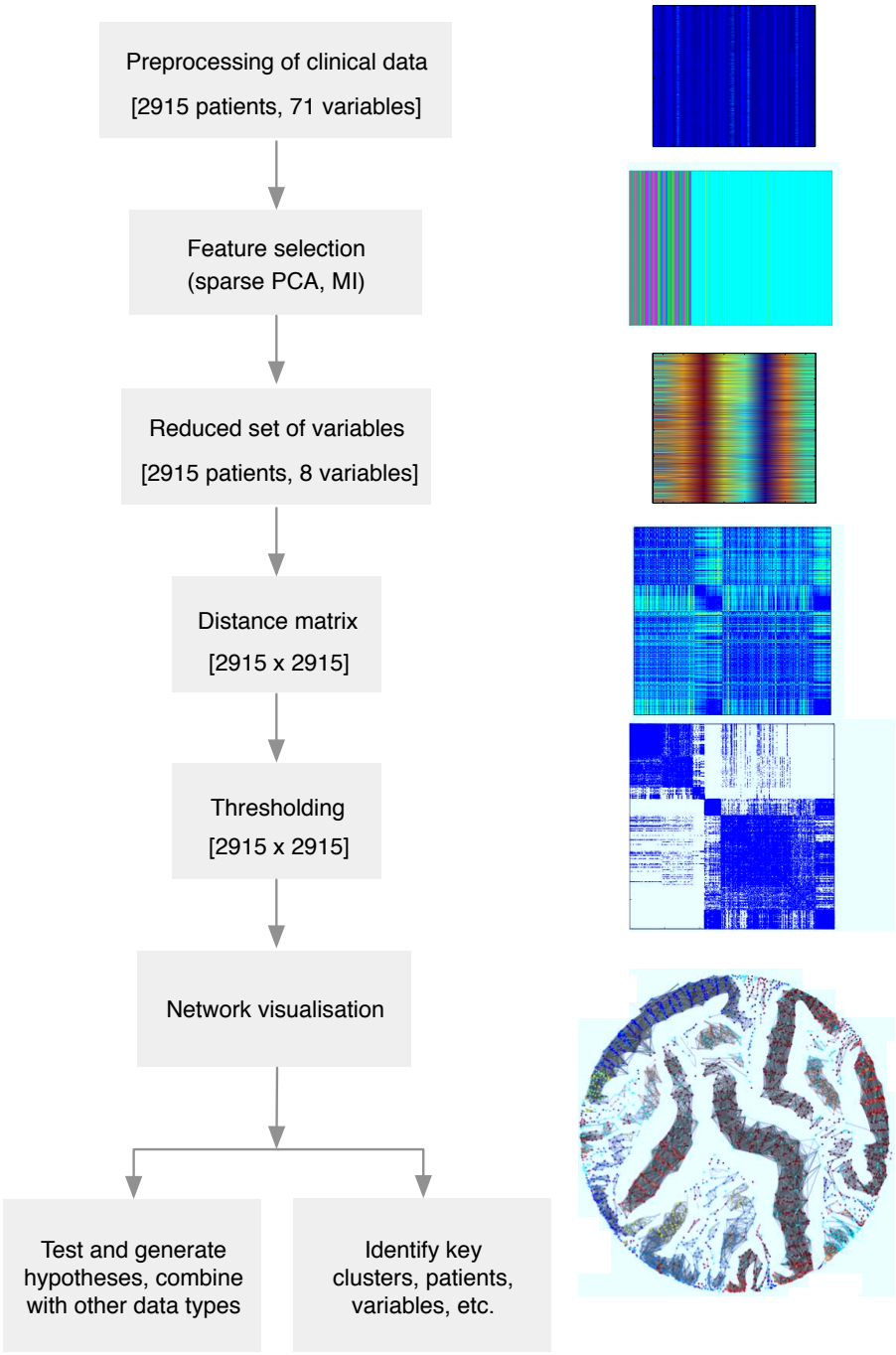


Figure 5.2: Flowchart summary of steps taken to analyse The Gambia dataset.

5.2.1 Pre-processing

Before analysing the data we begin with an exploratory data analysis, cleaning the data, dealing with missing values and normalisation.

5.2.1.1 Mixed data

Most clustering algorithms work only for continuous or categorical data. A large number of real life experiments/surveys produce mixed datasets (those which include some categorical or binary variables as well as continuous variables), which makes the clustering of mixed data an important topic [He et al., 2005].

One alternative to using mixed datasets for clustering and data analysis is to build a similarity measure which deals with mixed types of data, for example, one that combines similarity measures for the different types of data with appropriate weighting, transformation and normalisation. Another is to dichotomise all the variables, so that we end up with a pure categorical dataset, and use a similarity measure for binary data. A third approach, which is the one we used, is to rescale all the variables to the same scale and treat them all as continuous data [Wright et al., 2003; Everitt et al., 2011].

Which approach is best to use depends on the data. In the case of the malaria dataset, most features are binary and the only categorical ones represent a natural scale of a symptom. For example in the case of totalbcs, or total comascore, we can assume that there is a 'comatoseness' feature which can be measured continuously, so that a child can be 'a fifth comatose' if his level of comascore is 20% of the highest level of coma. For simplicity of the analysis and application of the methods we used continuous variables, but after testing the other alternatives and obtaining highly close similarities between the patients, we are confident the results were not affected by this choice.

5.2.1.2 Data cleaning

Data validation is a critical step to prevent downstream errors in the analytical workflow. The dataset was first inspected in order to detect and correct missing, corrupt or inconsistent entries.

A first approach to assess the quality of the data is through visualisation. Using histograms, boxplots and other common visualisation techniques, it is possible to identify general trends and structure in the data, and to identify potential outliers.

The identification of outliers can lead to the detection of mistakes in the data acquisition or data recording as well as in other steps in the process of gathering the data and building the dataset. Verifying that data are compatible and reasonable can be done by setting automatic checks on the data, however, expertise is required in order to evaluate the accuracy of the data, and in the case of very large sets of data, this process cannot be done value by value in the dataset, so a random selection of data points must be chosen.

Data duplication is a common source of bias when large datasets are merged [Maletic and Marcus, 2000]. We ensured that dataset merging did not result in duplicate observations, by careful examination/nomenclature of unique variables to link multiple datasets and identification and management of duplicate observations using standard statistical packages.

Although this can be a time consuming and tedious step, it is paramount for the success of the analysis. Working with uncleaned data can lead to misleading results, such as the presence of outliers, which would make the rest of the data points appear closer than they really are (if we normalise by the maximum pairwise distance which, in the existence of outliers, can be disproportionately large to average pairwise distances). In complex or multi-step data analysis, errors can be propagated from one step to the next, making it very difficult to trace the source of error.

5.2.1.3 Missing data imputation

The identification and treatment of missing values is important when pre-processing the data. A large proportion of biological data, such as gene expression microarray and clinical data, have missing values, which can be an obstacle for downstream data analysis [Tuikkala et al., 2008].

Several missing value imputation techniques exist and the optimal choice largely depends on the particular dataset. It has been shown that the approaches to deal with missing data vary drastically in their performance among different datasets [Tuikkala et al., 2008].

As a general first step it is essential to identify the nature of the missing values. In analogy with measurement errors, missing values might arise systematically or at random (MAR). One example of the former case is measurements falling outside the range of sensors (e.g. astronomical data) [Wagstaff and Laidler, 2005]. These fundamentally distinct classes of missing values require different treatment.

In the case of the Malaria dataset, missing values appear in most of the variables and do not show correlation, which is why we make the assumption that the missing values are missing at random.

Listwise deletion is the simplest way to manage missing values. This technique is also known as case deletion (CD) or marginalisation. For practical reasons, this method was not used in the malaria dataset to ensure that all biologically relevant variables were present in the analysis [Acuña and Rodríguez, 2004].

A common practice is to use the average of all the known values for a given variable. This imputation method, called mean imputation or row average (RAVG), does not affect the mean of the variables and it has proved to give good experimental results in supervised classification tasks [Chan and Dunn, 1972; Mundfrom and Whitcomb, 1998]. However, this method has been shown to distort the shape of the data distribution, underestimate the variance and to be influenced by outliers [Wright et al., 2003; Acuña and Rodríguez, 2004].

Median imputation (using median instead of mean of the known variable values) is often used as an improvement over the mean, since it is more robust and it is not affected by outliers. It is suggested as the best choice for skewed feature distributions [Acuña and Rodríguez, 2004].

Other missing value imputation techniques use clustering methods, through soft constraints for missing values or by using similar values of clustered data: first clustering the data using the known values, and then choosing a missing value imputation method to apply within the different clusters [Wagstaff and Laidler, 2005; Zhang et al., 2008].

More complex methodologies which use the rest of the variables to compute missing values and which have proven very accurate are Bayesian principal component analysis (BPCA) and expectation maximisation (EM). BPCA does not rely on parameters and has been shown to have a high success rate when compared with other imputation methods and is particularly suitable for large datasets [Oba et al., 2003]. On the other hand, EM is an iterative procedure which calculates maximum-likelihood estimates using the data available [Acuña and Rodríguez, 2004].

Following Acuña and Rodríguez [2004]'s strategy to compare missing value imputation methods, we contrasted the clustering results when using mean imputation, median imputation, BPCA and EM, and found that the classification error of the results obtained using those four methods was extremely low (median to mean imputation: 1.06%, BPCA to mean imputation: 1.28% and EM to mean imputation: 1.33%), which tells us that for our dataset and analysis, the results are very similar regardless of the chosen imputation method, and that using the simple mean imputation technique is a valid choice for this particular dataset.

5.2.1.4 Normalisation

It is important to normalise the different variables so that we can compare them accurately, without bias toward some of the features. This is particularly relevant when variables are measured on different scales and when mixing binary with categorical data.

One approach is to use global normalisation, where a constant scaling factor is applied to every measurement on the dataset, so that a constant median value is obtained. Since they are scaled by a constant factor, the relative values between the data points remain unchanged for the same variables, but the values themselves change.

Another option is per-variable normalisation, where the scaling factors chosen differ from feature to feature. With this type of normalisation it is possible to obtain the same maximum value for every variable.

5.2.2 Feature selection

Reducing the dimensionality of large datasets such as the malaria one is important for several reasons. First of all, a number of features may be redundant and/or irrelevant, the data's intrinsic dimensionality is likely to be smaller than the number of variables chosen to describe them, and the analysis and visualisation of data is facilitated by using a reduced number of features [Leskovec, 2006].

Dimensionality reduction can be accomplished by feature selection or feature extraction. The former deals with selecting a subset of the original features, whereas the latter deals with reducing the dimensionality by projecting the original data onto a smaller dimensional space. Examples of feature extraction methods are PCA and SOMs [van Hulle and Davis, 1998].

Feature selection and reduction as a preprocessing of clustering is a useful tool, as having less redundant, noisy and/or non-informative features improves the quality of partitioning algorithms [Dash and Liu, 2000]. On the other hand clustering methods can also be used

to select features [Butterworth, 2005].

In this section we describe the use of DifFUZZY to select the most informative clinical features that best discriminate between different groups of patients.

By interrogating one variable at a time, DifFUZZY identifies how many compact clusters there are for each variable, and how many soft data points we can use to get a sense of the data structure: if there are a few large dense clusters and a small percentage of soft points, then we can say that such variable is highly informative for clustering. If, on the other hand, DifFUZZY returns several dense clusters and a high number of soft data points, then the variable is not in itself very informative. Choosing sets of variables which give us similar results allows the identification of variables that are highly correlated.

The DifFUZZY silhouette width DSW, introduced in Section 3.5.1, could be used as a quantitative measure for comparing sets of variables. This is a clustering quality index that indicates how good a clustering result is in terms of the compactness of the clusters and the presence of soft data points. If the data have a cluster structure, then a good clustering result would be one with large compact clusters and very few soft points, and the DifFUZZY silhouette width for such a case will be near 1. On the contrary, a very poor partitioning results in a large proportion of soft data points and small compact clusters, and the DSW would be near zero.

We identified 8 non-redundant features (shown in Fig. 5.3) which gave us high DSW in the separation of the syndromes: acmuscle, deepbr, convadm, hbtotal, itrecess, totalbcs, post and usleepy. These variables describe relevant symptoms associated with the progression of malaria: the use of accessory muscles (acmuscle), deep breathing (deepbr), and intercostal recession (itrecess) are symptoms of RD, as well as presenting a posture (post) which ranges from lethargic to prostration. Children with low levels of haemoglobin (hbtotal) are anaemic (SMA). Low values of totalbcs indicate a reduced level of consciousness. Children with CM can also present seizures (convulsions in admission, convadm) and be unusually sleepy (usleepy).

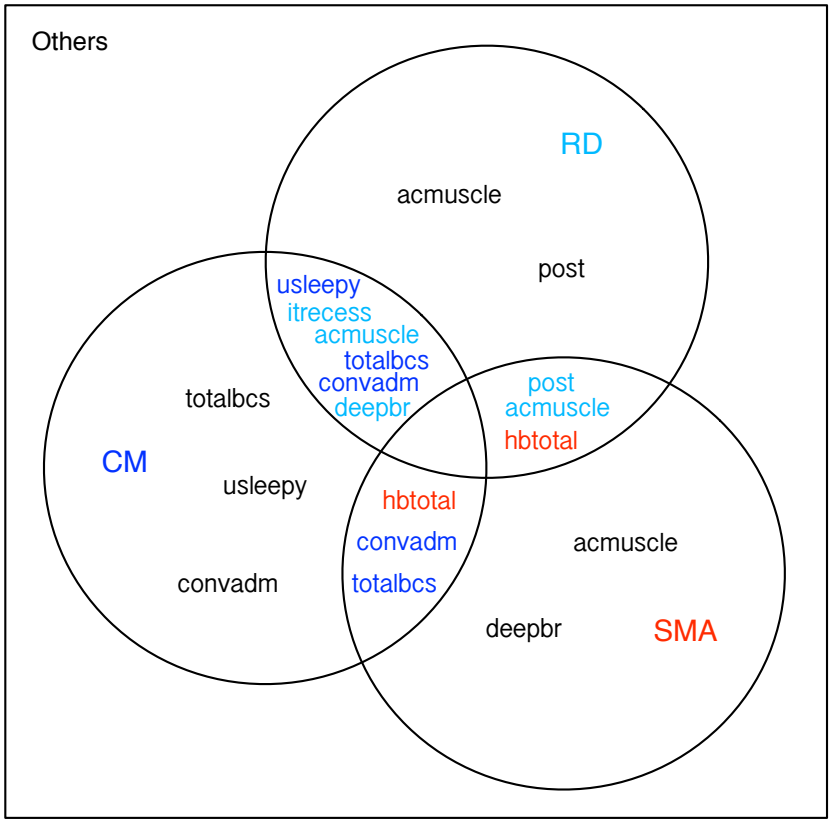


Figure 5.3: Summary of chosen discriminant variables. Colour represents variables which discriminate against the other pair member (e.g. in the intersection of the RD and CM classes, the cyan variables, such as itrecess, acmuscle, and deepbr are able to discriminate between CMRD and CM patients, but not between CMRD and RD patients)

By looking at one variable at a time we are not taking into account the possibility that features interact and their combination could be more or less informative than each variable separately. However, studying every possible combination of features is not a practical alternative given the large number of variables. We verified that by using the 8 selected variables we obtained a high DSW for the dataset, and we also verified that these variables could have been obtained by other standard feature selection or extraction approaches, such as PCA and mutual information (MI).

PCA is used as a dimensionality reduction and visualisation technique which groups variables rather than objects by separating these as far away as possible (through identifying

directions of highest variability of the data). On the other hand, clustering techniques such as DifFUZZY are often used to cluster groups of objects rather than variables. This, however, can be easily overcome by exchanging variables for objects, and for our dataset this means obtaining clusters of variables that present similar values across patients, and the feature selection would be achieved by removing redundant co-clustered variables. Furthermore, unlike PCA, clustering-based feature selection can return a set of variables, and not a linear combination of them.

Malaria researchers and clinicians have largely agreed with the variables selected as the most discriminant among severe malaria types. In particular, the variables hbtotal, totalbcs and deepbr are the ones that clinicians would use in order to identify the syndromes, using the variables usleepy, itrecess, acmuscle and convadm as supporting variables, with post being the variable less used.

5.2.3 Constructing the distance matrix

After identifying the most informative features we build a distance matrix (**DM**) $\in \mathbb{R}^{N\times N}$:

$$\mathrm{DM}_{i,j} = \left|\left|\mathbf{x}_i - \mathbf{x}_j\right|\right| \tag{5.1}$$

where $\mathbf{x}_i \in \mathbb{R}^F$, contains the values of the original i^{th} data point in the F features selected. These are the distances between every pair of nodes/patients in the fully connected network. The large size of this matrix makes it very difficult to study and visualise using standard methods.

A useful strategy to deal with such large distance matrix is to sparsify the data by just keeping the most important ones. In this case, the smallest distances are the ones which give us the local information regarding the groups of patients. In the next section we address how to determine the distance threshold above which we will omit the matrix entries.

5.2.4 Thresholding

In order to choose a suitable threshold that yields a network which is both meaningful and tractable, we take into account two key factors: the distribution of cluster sizes and the average clustering coefficient of points in the network. Note that if we choose a very small threshold, we will end up with a few small clusters and a large number of isolated points. Alternatively, if we choose a very large threshold value, we will link most nodes, obtaining a single large compact cluster and very few isolated points. Neither of these extreme cases provides information about the data structure.

We first find a superior limit for our threshold by not allowing networks with a single large component. Subsequently, we choose the threshold that yields the network with the highest average clustering coefficient of the points in the network. This second step is also unsupervised, and it corresponds to finding a network where clusters are formed of 'triangles' of data points, where if A is connected to both B and C, then B and C are also connected. When a network presents a high average clustering coefficient, the clusters are dense and well interconnected.

The identification of meaningful clusters requires the finding of groups of data points which share similarities within the cluster and are different to the data in other clusters. In our problem, when clustering nearly 3000 patients, a partition which gives only a very large cluster, comprising more than 80% of the data, and the rest of the data being scattered in groups of a few data points is not meaningful, given that the single large group includes very dissimilar data and does not give as much information on the structure of the data as a partitioning with more large/medium size clusters.

We can verify the threshold using connectivity, adjacency or sparsity pattern matrices (plotted through the "spy" plotting command in Matlab), which are obtained by thresholding the distance values between every pair of points. Points $\mathbf{x}_i$ and $\mathbf{x}_j$ will be connected only if their distance is less than a given threshold, and such a link will be represented as a 1 in the

connectivity matrix and plotted as a blue square, such as in Fig. 5.5, while the non-existing connections will be depicted as empty white squares. For 5% of the maximum of the pairwise distances, (the data have been normalised so that the maximum pairwise distance corresponds to 1), the diagonal block structure gives stronger connected clusters than for the other thresholds (high inter-connectivity, indicated by darker blocks). For larger thresholds, such as the bottom-right most matrix, most data points are connected, whereas in the top left-most matrix there are several small clusters of data points.

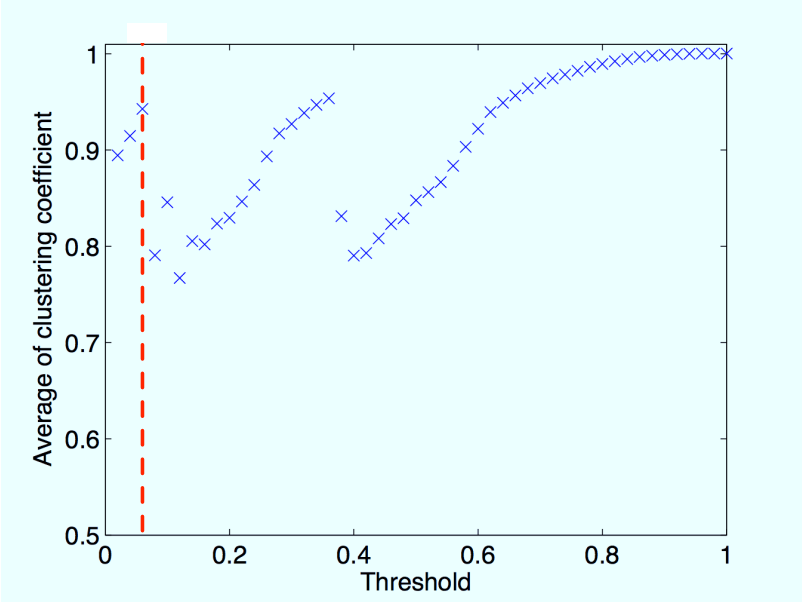


Figure 5.4: Clustering coefficient versus threshold. Given that we are considering only threshold values lower than 0.2 in order to avoid one large cluster with most of the data (see Appendix C.3), the threshold, shown by a red dotted line, which gives us the highest clustering coefficient, is 0.05.

Furthermore, this selected threshold is the smallest distance at which there is a gap in the cumulative distance frequencies between all patients, indicating that this is an intrinsic or natural threshold (scale) for the data at which data points cluster. This means that a larger proportion of patients are within a distance of roughly z units among themselves, vs. patients that lie within distances smaller or larger than z.

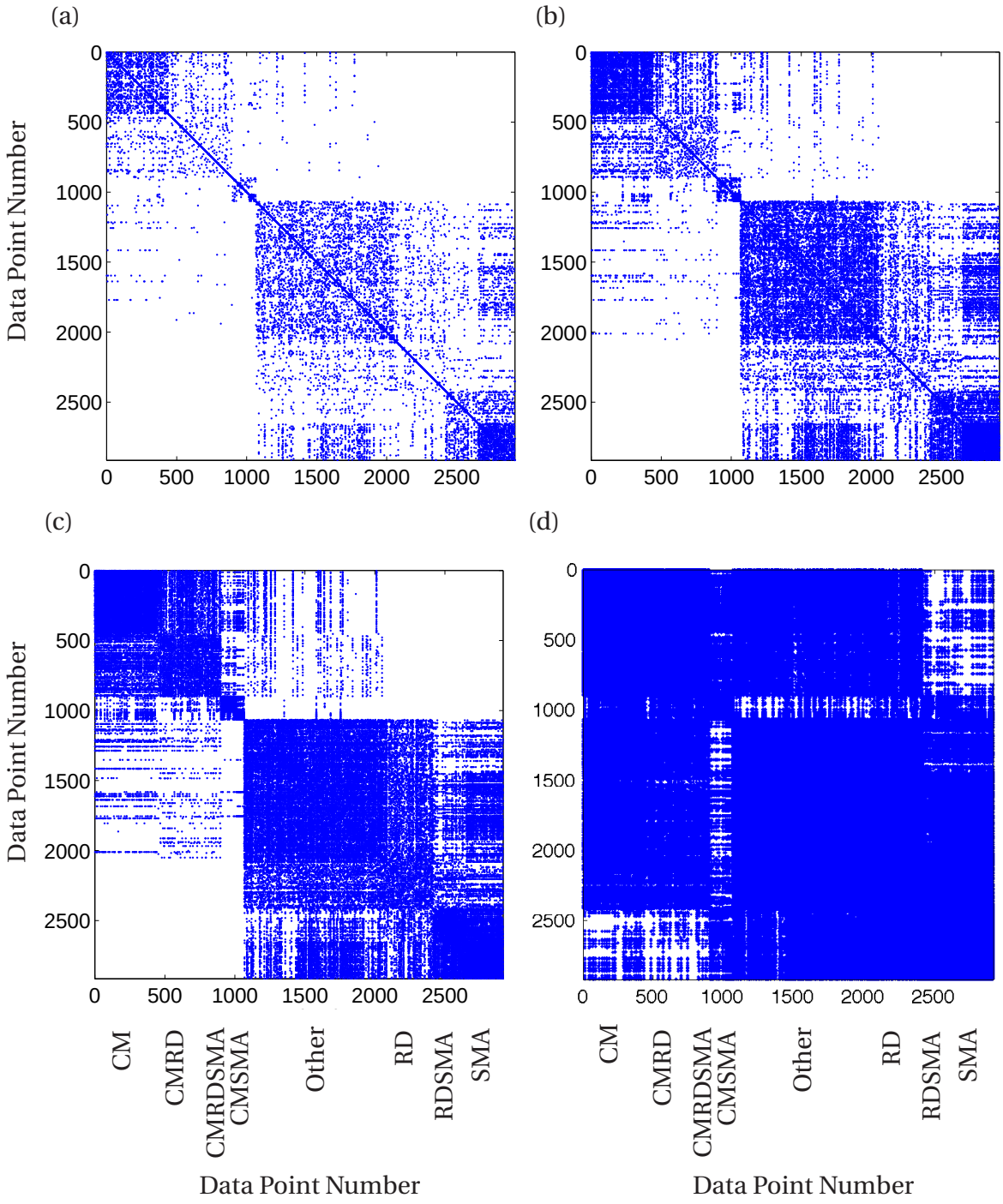


Figure 5.5: Sparsity pattern plots. Threshold distances: (a) 0.8, (b) 2.5, (c) 5 and (d) 10% maximum pairwise distances. The axis labels below (c) and (d) comprised of the syndrome types are also valid for (a) and (b), and correspond to the types shown in the left axes of the matrices.

5.2.5 Network visualisation

After determining a threshold for the pairwise distances among the data points we create an unweighted network by linking the data points whose distance falls within that threshold. This linkage is reflected by the similarity matrix **S** with entries

$$S_{i,j} = \begin{cases} 1 & \text{if } \left|\left|\mathbf{x}_i - \mathbf{x}_j\right|\right| \leq z \\ 0 & \text{otherwise,} \end{cases} \tag{5.2}$$

where z is the threshold found in Section 5.2.4 and $\mathbf{x}_i$ and $\mathbf{x}_j$ are the data points represented using the variables selected in Section 5.2.2.

This z-neighbourhood graph is then visualised using a fast spring-based network layout algorithm developed by D. Smith [Smith, 2012].

This force-directed algorithm is a fast and flexible tool for drawing unweighted graphs and is comprised of two main steps: (1) introducing a mechanical model which describes the nodes as bodies with forces acting between them and (2) determining the position of the nodes by minimising the energy of the system.

In the first step all nodes are connected by one of two types of linear springs: strong and short springs are placed between connected nodes ($S_{i,j} = 1$), and weaker long springs are placed between all pairs of unconnected nodes ($S_{i,j} = 0$).

The following step is an optimisation method to identify the minimum of the overall energy of the system of springs, which is given by

$$E = \frac{1}{4} \sum_{i=1}^{N} \sum_{j=1}^{N} k_{i,j} \left(\|\mathbf{x}_i - \mathbf{x}_j\| - l_{i,j} \right) \tag{5.3}$$

where $\|\mathbf{x}_i - \mathbf{x}_j\|$ is the Euclidean distance between nodes i and j and $k_{i,j}$ $l_{i,j}$ represent, respectively, the spring constant and the natural rest length of the spring between nodes i

and j [Smith, 2012].

Repulsive forces between unconnected data points will keep them apart, while attractive forces between similar data points ($S_{i,j} = 1$) will bring them closer together [Kobourov, 2012].

At the beginning all the nodes are randomly positioned so that no two nodes overlap. The layout is determined iteratively by moving the nodes due to the effect of the springs [Kobourov, 2012]. In every iteration shorter spring lengths and tolerances are used, achieving even further speed-up of the code.

5.3 Results

In Fig. 5.6 we present the network obtained using the method detailed in Section 5.2.5 for The Gambia dataset. To facilitate comparison with other networks we will refer to this one as the original network. Here we can observe large elongated clusters of children with similar syndromes: one large CM cluster with a CMSMA subgroup, three large clusters with 'Others' and SMA along with a few RD and RDSMA data points. We also see that CMRD patients cluster mainly with CMRDSMA patients, and that RD patients cluster with RDSMA patients in four distinct connected components highlighted in Fig. 5.7.

We define RD clusters as those mainly comprised of RD types (RD, RDSMA, CMRD or CMRDSMA). We are interested in RD clusters of size larger than a few data points. In Fig. 5.7 we show the four RD clustering identified in the global network for The Gambia dataset (Fig. 5.6), labelled $\widetilde{C}_{126}$, $\widetilde{C}_{125}$, $\widetilde{C}_{124}$ and $\widetilde{C}_{132}$[1].

[1] The indices of the RD clusters were given by the algorithm used to find connected components.

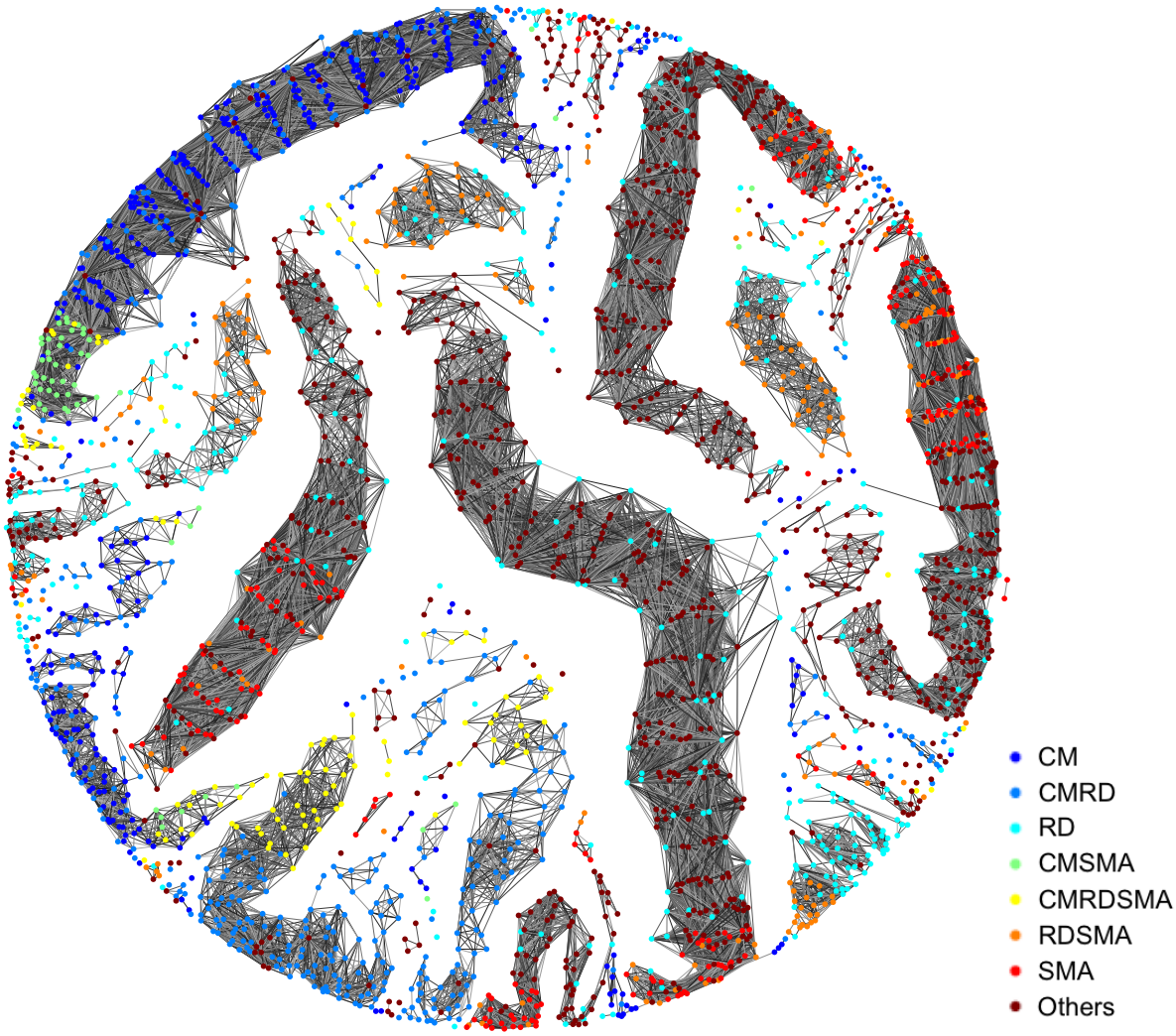


Figure 5.6: Global network for The Gambia dataset. The threshold is set at 5% of the maximum pairwise distance. Since this is an unweighted network, a shorter edge in this layout does not imply higher similarity between the connected nodes.

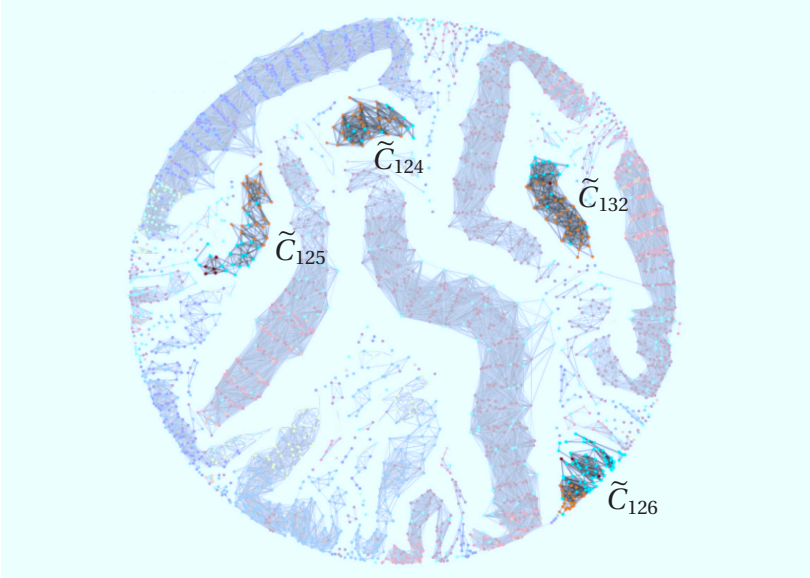


Figure 5.7: RD clusters from The Gambia network in Fig. 5.6. These four clusters are labelled starting from the bottom-most cluster in a clockwise fashion as $\widetilde{C}_{126}$, $\widetilde{C}_{125}$, $\widetilde{C}_{124}$ and $\widetilde{C}_{132}$.

Since the RD clusters from Fig. 5.7 did not merge together when we increased the threshold, we looked further into what variables separated them, and if there were other differences in features which were not selected in Section 5.2.2, so did not contribute to their separation in the network but could shed light on how these children differ from children in the other RD clusters (Table 5.2).

In Table 5.2 we can see that clusters $\widetilde{C}_{124}$ and $\widetilde{C}_{132}$ can be separated by totalbcs from clusters $\widetilde{C}_{125}$ and $\widetilde{C}_{126}$, and that the variable usleepy distinguishes children in cluster $\widetilde{C}_{124}$ from children in cluster $\widetilde{C}_{132}$.

We note that clusters $\widetilde{C}_{124}$ and $\widetilde{C}_{132}$ present high values of tfus, indicating that most of them have received blood transfusion, yet children in $\widetilde{C}_{132}$ have a 5-fold higher mortality rate than those in $\widetilde{C}_{124}$, as well as presenting a higher average value in the variable that denotes increased liver size (hepatomegaly). Such differences between clusters $\widetilde{C}_{124}$ and $\widetilde{C}_{132}$ suggest that children from either cluster have a different pathobiology of malaria, and should therefore be identified and treated differently.

	$\widetilde{C}_{124}$	$\widetilde{C}_{125}$	$\widetilde{C}_{126}$	$\widetilde{C}_{132}$
Selected features				
totalbcs	5.00 ± 0.00	4.00 ± 0.00	3.00 ± 0.02	5.00 ± 0.00
deepbr	0.83 ± 0.16	0.88 ± 0.16	0.91 ± 0.15	0.77 ± 0.14
usleepy	0.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	0.97 ± 0.07
itrecess	0.74 ± 0.20	0.78 ± 0.21	0.76 ± 0.21	0.68 ± 0.18
hbtotal	4.39 ± 0.74	5.08 ± 1.42	6.59 ± 2.00	4.42 ± 0.88
acmuscle	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
convadm	0.74 ± 0.08	0.79 ± 0.17	0.80 ± 0.18	0.76 ± 0.11
post	0.49 ± 0.08	0.52 ± 0.14	0.54 ± 0.17	0.50 ± 0.10
Additional variables				
death	0.03 ± 0.16	0.16 ± 0.37	0.21 ± 0.41	0.10 ± 0.30
tfus	0.97 ± 0.11	0.76 ± 0.43	0.53 ± 0.49	0.98 ± 0.10
age	32.10 ± 30.76	31.75 ± 20.30	42.52 ± 28.77	28.49 ± 25.07
liver	2.55 ± 2.15	3.23 ± 2.41	2.35 ± 2.42	3.80 ± 2.45

Table 5.2: RD clusters: mean values and standard deviations of selected features (totalbcs, deepbr, usleepy, itrecess, hbtotal, acmuscle, convadm and post) and additional variables (death, tfus, age and liver).

We hypothesised that given this combination of clinical features, a plausible explanation for the increased mortality could be that children in cluster $\widetilde{C}_{132}$ were in heart failure. To test this hypothesis, laboratory analysis were performed by Dr. Huang from the Wellcome Trust Centre for Human Genetics: stored plasma samples from these patients were retrieved and used to measure a biomarker associated heart failure (B-type natriuretic peptide, BNP) and found that the concentration of this marker was indeed significantly higher in patients from cluster $\widetilde{C}_{132}$ compared with those of cluster $\widetilde{C}_{124}$. This finding, along with the knowledge that heart failure is deadly [Cleland, 1997], supports our hypothesis of heart failure in cluster $\widetilde{C}_{132}$. Additional laboratory tests are being carried out to confirm this finding.

5.3.1 Network cluster validation

We verified that the clusters we identified using our stepwise approach summarised in Fig. 5.2 could have been obtained using a traditional clustering method. Using hierarchical clustering with average linkage, $K = 150$, we reproduced the partition from the original cluster shown in Fig. 5.6, yielding an RI of 0.98 (Rand Index was defined in Section 2.2.2.2). Furthermore, using hierarchical clustering with single and complete linkage, but a different value for K, would have also allowed us to obtain a very similar partition, but with slightly lower RI values.

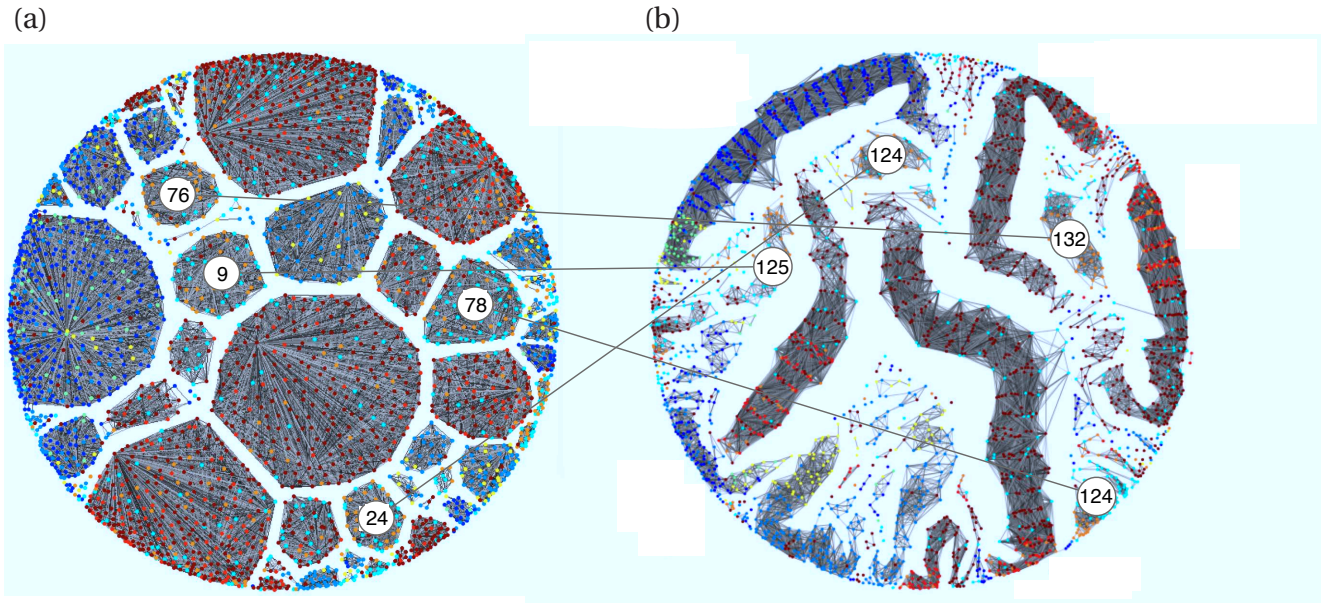


Figure 5.8: Correspondence of clusters from the (b) global network and a network obtained using hierarchical clustering (a), average linkage, $K = 150$ (left). Lines indicate the corresponding HC clusters to the individual RD clusters from the original network.

We built a network from the clustering results obtained using hierarchical clustering, average linkage, by creating an edge between any pair of points that belong to the same cluster. This yields fully connected components, with a large total number of edges

$$\sum_{k=1}^{K} \frac{1}{2} \left(|\mathbf{C}_k|^2 - |\mathbf{C}_k| \right), \tag{5.4}$$

where $|\mathbf{C}_k|$ is the number of elements in the k^{th} cluster and K the number of clusters or components in the partitioning/network.

Such large number of edges make the visualisation more computationally expensive. Also, given that all the HC clusters are fully connected, the visualisation method renders clusters of polygonal shape with a well mixed distribution of different syndrome types. This differs from the elongated clusters in the original network, where the syndromes appear segregated in some clusters due to the difference of densities of edges between the subgroups.

In Fig. 5.8 we see the correspondence of RD clusters between both networks as indicated by the lines.

		HC clusters					
		C_{24}	C_9	C_{78}	C_{76}	Other clusters	TOTAL
Original clusters	$\widetilde{C}_{124}$	39	0	0	0	1	40
	$\widetilde{C}_{125}$	0	47	0	0	4	51
	$\widetilde{C}_{126}$	0	0	74	0	11	85
	$\widetilde{C}_{132}$	0	0	0	39	12	51
	Other clusters	9	6	1	5	0	21
	TOTAL	48	53	75	44	28	248

Table 5.3: Confusion Matrix of RD clusters in HC and Original network in Fig. 5.6.

In Table 5.3 we can see the good agreement between the RD clusters in both networks. Additionally, the few remaining data points from the original cluster, $\widetilde{C}_{126}$, which do not belong to cluster C_{78} in the HC network, form a smaller independent cluster. The same is true for cluster $\widetilde{C}_{132}$.

5.3.2 Network validation

We were interested in knowing if the features in Fig. 5.3 could also give a similar network partition in malaria data from different African regions, or if they are intrinsic to the dataset studied. For this we requested data from the Severe Malaria in African Children (SMAC) network [Taylor et al., 2006] that was set up to conduct mortality-based trials from six sites in five countries across Africa, shown on the map in Fig. 5.9. The cities and countries where the sites are located are: Banjul (The Gambia), Blantyre (Malawi), Kumasi (Ghana), Kilifi (Kenya), Lambaréné (Gabon) and Libreville (Gabon).

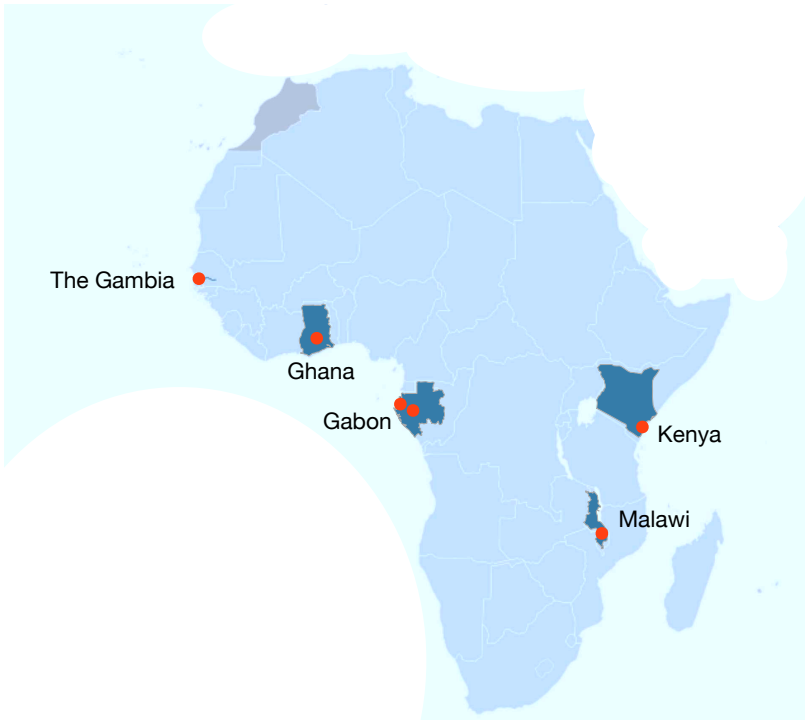


Figure 5.9: SMAC network countries on a map of Africa. The sites are: Banjul (The Gambia), Blantyre (Malawi), Kumasi (Ghana), Kilifi (Kenya), Lambaréné (Gabon) and Libreville (Gabon).

The SMAC dataset is comprised of data from 26,106 children described with around 20 variables, most of which coincide with those from The Gambia dataset. Some of the key differences between the SMAC and The Gambia datasets is that this second dataset was compiled over 12 years, whereas The SMAC dataset includes data from one year only (2001), and

the fact that The Gambia dataset corresponds to only one country while the SMAC dataset includes data obtained across a variety of epidemiological settings [Taylor et al., 2006].

The data were processed as above, with the difference that three of the selected features for The Gambia dataset were not present in the SMAC dataset (acmuscle, usleepy and post), so the analysis was performed only with the other 5 variables (deepbr, itrecess, hbtotal, convadm and totalbcs). The threshold (8% maximum pairwise distance) was identified following the strategy described for The Gambia dataset in Section 5.2.4. In Appendix C.3 we include the relevant figures.

In Figs. 5.10 and 5.11 we present four site-specific SMAC networks. Like the global SMAC network in Fig. 5.12 they do not include 'Others'.

The networks for the two sites in Gabon are presented in Appendix C.4.1.1. Because of their low number of data points the resulting networks are more sparse and the RD clusters are comprised of very low numbers, which makes a statistical comparison between the clusters not significant.

(a) Banjul

(b) Blantyre

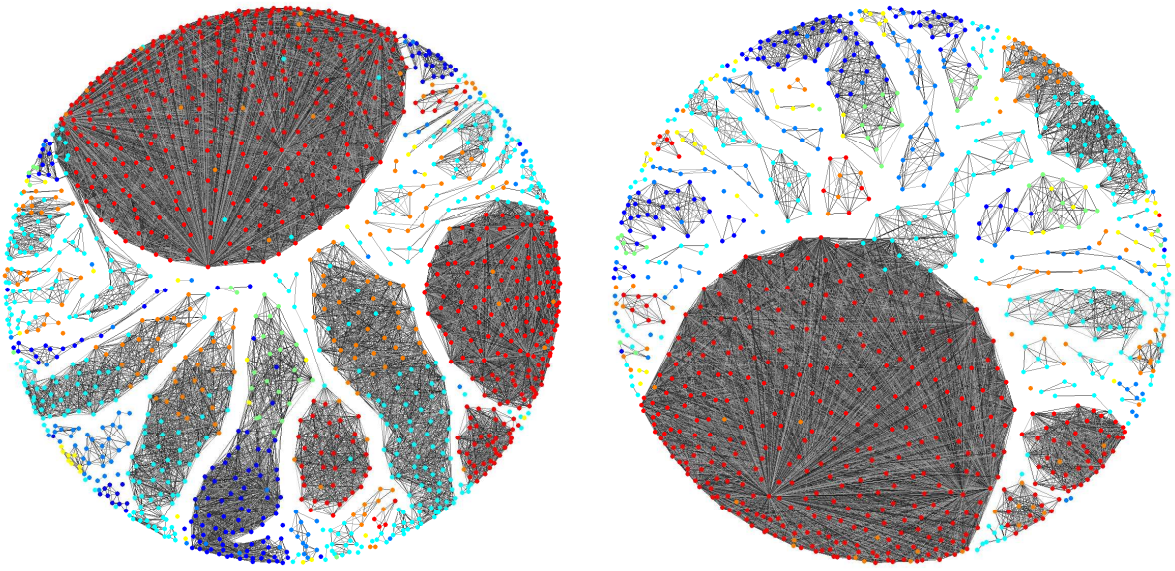


Figure 5.10: Networks for the Banjul (Gambia) and Blantyre (Malawi) sites.

(a) Kumasi (b) Kilifi

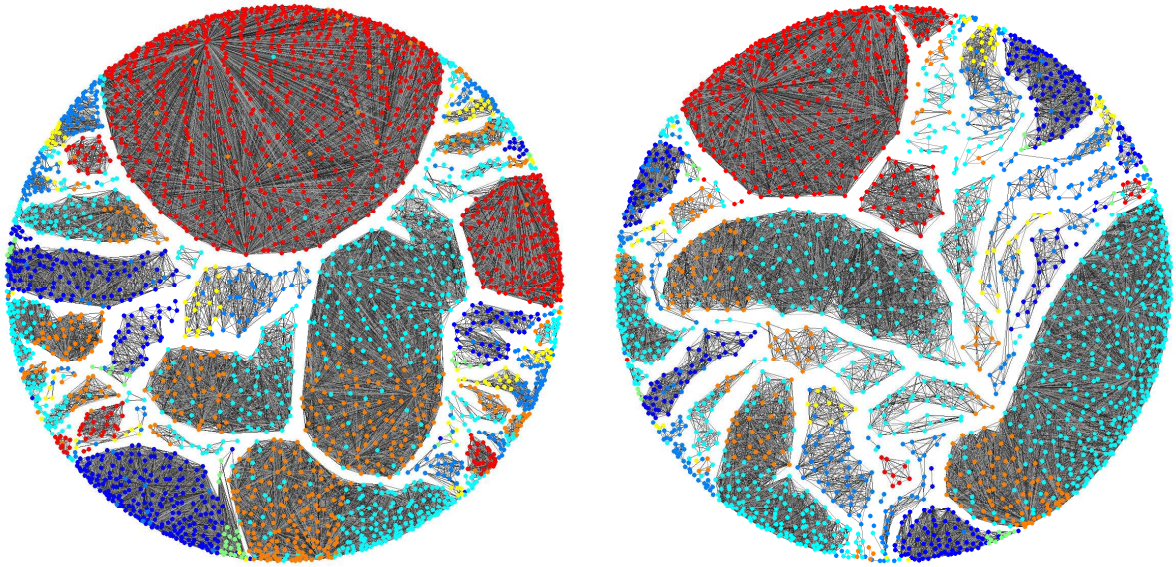


Figure 5.11: Networks for the Kumasi (Ghana) and Kilifi (Kenya) sites.

	C_1	C_2	C_3	C_4	C_5
Selected features					
totalbcs	5.00 ± 0.00	5.00 ± 0.00	5.00 ± 0.00	5.00 ± 0.00	5.00 ± 0.00
deepbr	0.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	0.00 ± 0.00
itrecess	1.00 ± 0.00	1.00 ± 0.00	0.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
hbtotal	6.49 ± 2.56	5.99 ± 2.71	6.37 ± 2.63	6.23 ± 2.54	6.72 ± 2.35
convadm	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
Additional variables					
death	0.03 ± 0.17	0.09 ± 0.29	0.06 ± 0.24	0.10 ± 0.30	0.04 ± 0.20
tfus	0.42 ± 0.49	0.57 ± 0.50	0.47 ± 0.50	0.57 ± 0.50	0.46 ± 0.50
age	23.88 ± 22.3	26.25 ± 27.1	32.55 ± 31.0	27.69 ± 22.3	24.49 ± 20.5

Table 5.4: SMAC RD clusters: mean values and standard deviations of The Gambia's selected features available in the SMAC dataset (totalbcs, deepbr, itrecess, hbtotal and convadm) and additional variables (death, tfus and age).

Qualitatively we see that in Fig. 5.12 there are RD clusters in the SMAC dataset similar to those in the original network (Fig. 5.6).

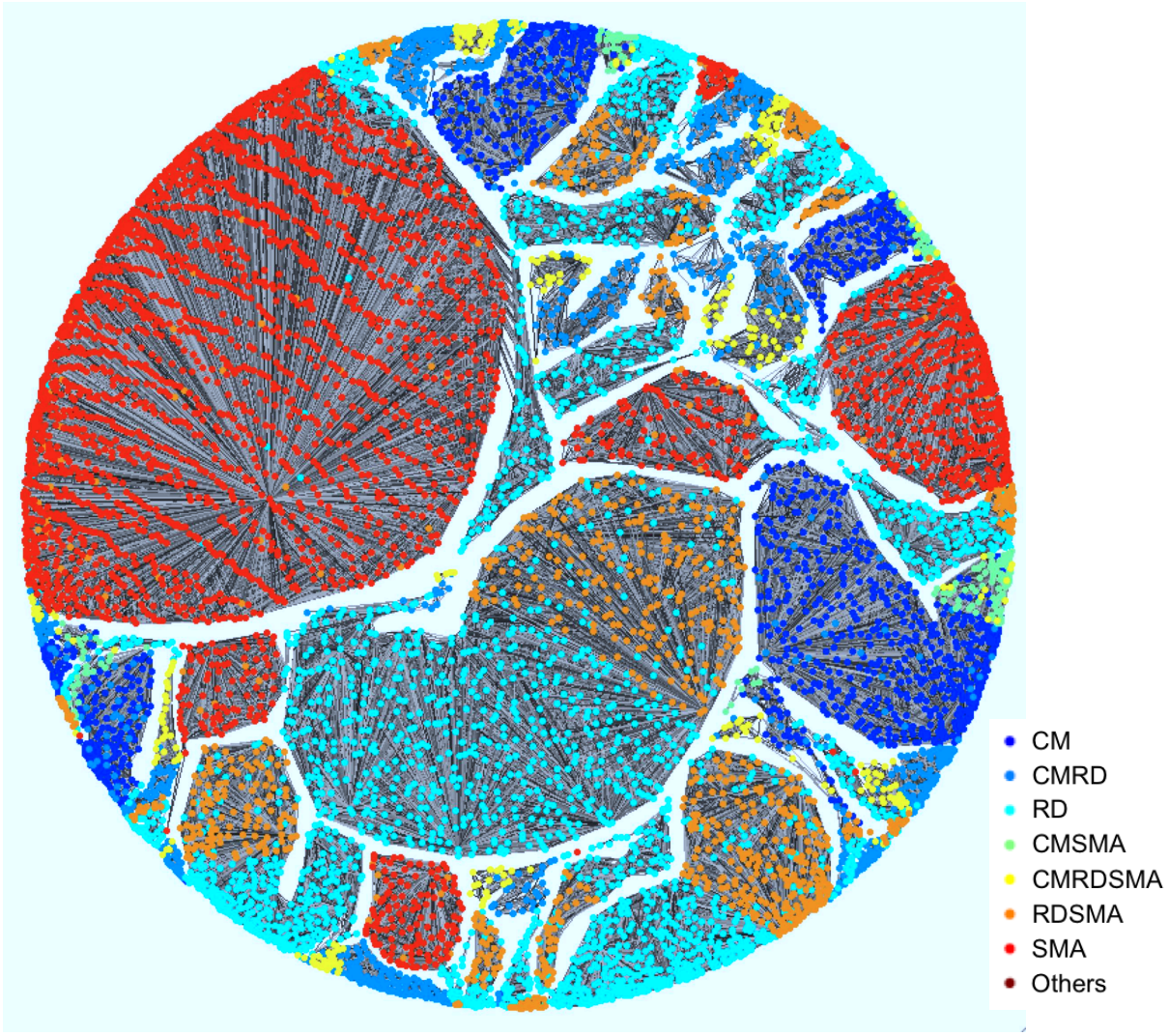


Figure 5.12: SMAC network. Threshold distance: 8% maximum pairwise distances. 'Others' not included.

From Table 5.4 we see that the SMAC RD clusters (five largest clusters comprised mainly of RD and RDSMA patients) vary in deepbr, itrecess and convadm, but not in the totalbcs feature and that they lack the variables usleepy or liver to compare between with RD clusters of The Gambia dataset. Cluster C_4 presents a higher mortality rate than C_1, but the transfusion rates are similar between the five clusters. In the different sites there are different transfusion protocols in place: due to the higher HIV prevalence in Southern and Eastern Africa compared to the rest of the continent, blood transfusion is performed much later in children with malaria in Kenya and Malawi than in Gabon and The Gambia.

5.4 Discussion

In this chapter we showcased an alternative use of clustering methods as a feature selection technique in a processing pipeline in clinical malaria data.

We proposed a methodology to analyse, through clustering and network visualisation, complex clinical datasets and showed that using this approach we can generate testable hypotheses.

It is important to note that the features we identified to be good separating variables are not necessarily the best features or best combinations possible. Using different feature selection approaches would likely lead to a different set of variables. However, the selected variables were able to discriminate between the malaria syndromes without adding unnecessary noise.

The question of how many groups of patients exist, is an emergent property from the data, which is uncovered by the methodology used. The maximum average clustering coefficient was used to determine the intrinsic clustering structure of the dataset, allowing us to identify subgroups of children that may not have been detected if we had specified the number of clusters *a priori*.

We have been able to reproduce the WHO classification of severe malaria syndromes and we have identified different clustering patterns for children with respiratory distress and severe anemia. One of these clusters has an average mortality rate 5 times higher than the other clusters. Further analysis of this cluster revealed that most children in this group exhibit hepatomegaly, indicating the possibility that this group of children are in heart failure. The hypothesis was subsequently confirmed using stored plasma samples from patients by measuring a biomarker associated with heart failure. This suggests such patients need a different treatment than the one currently prescribed, and further clinical research must be conducted to underpin our findings and devise a suitable treatment to decrease the mortality of children with similar diagnoses.

Chapter 6

Conclusions and Future Directions

In this chapter we give the main conclusions of this thesis, as well as the future directions. We examine potential applications of DifFUZZY in multiscale modelling, medical image segmentation, time series clustering and introduce preliminary work done along these lines.

6.1 Conclusions

In this thesis we presented a novel soft spectral clustering method, DifFUZZY, freely available for download from `http://www.maths.ox.ac.uk/cmb/diffuzzy`, which outperforms FCM for several test datasets. It was designed to deal with clusters that are non-linearly separable, and complex datasets such as microarray data. DifFUZZY's diffusion distance differs from traditional distance measures, such as Euclidean, where the distance is determined by the length of the shortest path connecting two points disregarding intermediate data points.

We have shown that although there is a vast array of clustering methods, there is still room for improvement and for developing more successful clustering techniques. Likewise, there is still the opportunity to improve DifFUZZY and include new features to make it better suited to deal with microarray data in particular.

Since this new clustering method works remarkably well with odd-shaped clusters, being able to resolve very complex cluster cores, there is a good chance that DifFUZZY would

perform particularly well in pattern recognition tasks, which we plan to look into as future research.

DifFUZZY addresses a number of the traditional challenges of clustering methods: it is able to deal with noisy data (shown by successfully clustering real biological data), and can deal with different types of attributes (by clustering well both numerical, categorical and combinations of them, such as the *Zoo* dataset).

We showed that DifFUZZY can be used in feature selection and contribute towards the discovery of relevant groups of patients in complex clinical datasets.

We can say that there is not a single clustering method that is suitable for every data clustering problem. We showed that all the methods reviewed along with our method perform with different levels of success on a number of datasets. The different algorithms have trade-offs, some are scalable and robust, but can not reproduce all partitionings [Kleinberg, 2002].

We also note that to validate clustering methods is not an easy task, and a large number of datasets must be used to compare the performance of different approaches. Such a lack of standards in assessing clustering algorithms is why, unlike supervised classification methods, it largely remains a field where trial-and-error still have a place, and the choice of appropriate parameters can be considered more art than science, and a much larger effort from the wide scientific community is required to drive it towards a more rigorous/theoretic field [Guyon et al., 2009].

6.2 Future Directions

6.2.1 Clustering in cellular modelling

One of the great challenges in theoretical biology is to establish how processes (e.g. signalling pathways, apoptosis, proliferation) interact across vast spatial and temporal scales. Cells are complex factories whose study is very difficult to undertake, therefore modelling could be a very useful tool as a framework to simulate their dynamics and predict their behaviour under different conditions, such as application of drugs, change of environment, etc. Much effort has been put into modelling cells and their interactions [Murray, 2005]. Cellular modelling, however, still does not utilise the full potential of available bioinformatics data. Model parameters are often chosen in an *ad-hoc* manner [Sarkar and Sobie, 2010], adding to the scepticism of experimentalists and clinicians. We suggest that reliable clustering methods applied to high-throughput data could bring together the fields of cellular modelling and bioinformatics.

For example, clustering microarray data can allow us to identify genes with similar expression patterns and select the most representative genes in each group. This information can be incorporated into a sub-cellular model, via reverse engineering, or by combining with well known pathways and these sub-cellular models can then be embedded into a cell-based model. In this way, there is the possibility of deriving computationally tractable models.

In this section we will present preliminary results of a case study of an experimentally important multiscale system. We want to address how differential adhesion is involved in colorectal cancer progression (Fig. 6.1), and how, in particular, mutations at the gene level can affect the tissue of the intestinal crypt.

The first step would be to obtain suitable crypt microarray data from which to derive information on gene interactions using soft clustering techniques.

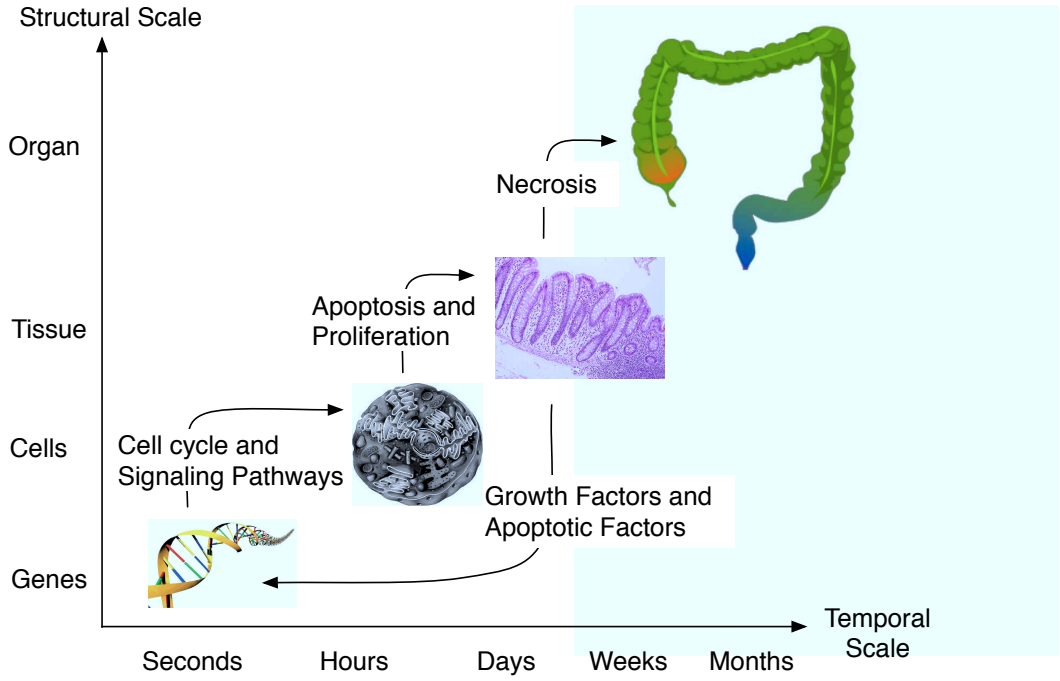


Figure 6.1: Colorectal cancer as a multiscale disease involving various interacting phenomena occurring at different structural and temporal levels. Changes, like mutations at the gene level, can affect the whole organ, starting from modifying the behaviour of individual cells, and then progressing to alter the tissue that contains these cells.

Cancer is a multiscale phenomenon (Fig. 6.1), both temporally and spatially. Mutations taking place at the gene level lead to changes in the function or state of proteins (such as being active or inactive constitutively), which affects the regulatory and signalling pathways in cells, including those controlling the cell cycle and apoptosis. Ultimately, these changes at the cell level affect the tissue (tumour formation, necrosis, loss of vasculature, degradation of extracellular matrix, etc.) and even the organ and the organism as a whole (metastasis and death). Also, changes at the macroscopic level (multicellular or tissue level) affect the sub-cellular level through signals for growth and non-growth. Changes at the sub-cellular level occur on the scale of seconds or minutes, whereas adjustments at the cell level take minutes or hours and the effects at the tissue level are observed after days. The resulting modifications in the organ and organisms take weeks, months or even years.

In order to model the various scales involved in cancer progression different approaches

may be used: discrete, continuum or hybrid [Walter, 2009]. Continuum models are typically used to model the sub-cellular and macroscopic levels [Walter, 2009]. At the sub-cellular level, cell signalling pathways are modelled using ordinary differential equations, and at the macroscopic level nutrient concentration or densities of cells are modelled using partial differential equations [Wishart et al., 2005]. On the other hand, discrete models such as Lattice Boltzmann models or agent-based models (cellular automata, extended Potts, off-lattice models) are chosen to model systems with only a few interacting cells [Deutsch and Dormann, 2004].

Hybrid approaches integrate discrete and continuum models. For example, a discrete model can represent the cells and a continuum approach can define the external concentration of nutrients or the concentration of intracellular molecules.

In the Appendix C.4.1.1 we describe the work that has already been done towards building a cell-level model with embedded sub-cellular pathways via differential equations.

6.2.1.1 Biological motivation

The control of cellular movement in epithelial tissue in morphogenesis, tissue invasion or wound healing is still, largely, an open question. Whilst there are many factors that control cell motion in epithelial tissue, differential adhesion has been proposed as a key regulatory mechanism [Steinberg, 2007]. In the intestinal crypt, cells of different types do not intermingle (differentiated with proliferative cells in particular), and present different levels of activation of Eph/ephrin proteins which regulate cell-cell adhesion in other epithelial tissues such as the neural crest [Kasemeier-Kulesa et al., 2006; Wong et al., 2010]. Eph/ephrin expression also interacts with other proteins such as β-catenin, a key factor in transcriptional activity in proliferative cells in the crypt [Batlle et al., 2002; Wong et al., 2010].

We plan to address the question of how differential adhesion (interlinked with sub-cellular pathways of the cell cycle and Eph/ephrin and β-catenin) can explain coordinated movement and cell sorting in the intestinal crypt.

6.2.1.2 Sub-cellular model

It is here, at the sub-cellular level, where we believe clustering could bring major benefits for cellular modelling. By determining co-expression of genes, clustering tools could be a robust and efficient tool to discover regulatory networks to embed as sub-cellular models. This is especially useful since there are important cost and time constraints associated with carrying out replicates of expression studies for individual genes. Chaste's modularity makes embedding sub-cellular ODE and PDE models into cell-based models an easy task (Chaste is a simulation package used and described in the Appendix in Section D.1.3).

Walter [2009] embedded the cell cycle models of Swat et al. [2004] and Tyson and Novak [2001] into his vertex-based model. At each time step, for every undifferentiated cell, the cell-cycle model is checked to verify that it has reached the end of the phase it is currently undergoing. When a cell reaches the end of the M-phase (mitotic phase), it divides.

Other models combine the effect of extracellular Wnt concentration with the cell-cycle though proteins APC and β-catenin. The extracellular Wnt concentration stimulates β-catenin's level, which in turn affects the length of various cell-cycle stages [Walter, 2009].

As future work we would like to embed an ODE system which represents adhesion proteins such as Eph/ephrin and link them with transcription through β-catenin. Studying how these ligands help regulate cell migration and cell adhesion in the intestinal crypt, may contribute to our understanding on how disruptions of their mechanism could lead to colorectal cancer progression.

In what remains of this chapter we will focus on more common applications of clustering techniques, and in particular on soft alternatives such as image segmentation and time-series data clustering.

6.2.2 Clustering in medical imaging

Major branches of imaging are image segmentation, video compression, object tracking, among others, the first being one of the major research topics in the area. Images are segmented to identify objects or to divide an image into homogeneous regions by either colour, texture, or other characteristics [Horvath, 2006]. Image segmentation consists of assigning to each pixel, a label or cluster membership. In medical images, where tissues join or are superimposed, pixels can belong to more than one region or object, which makes soft clustering methods better suited for handling these tasks [Zhang and Jiang, 2009].

Fuzzy C-means, the most widely used soft clustering method [Thomas et al., 2009] is a standard technique in image segmentation. However, it comes with serious flaws, such as being extremely sensitive to noise and other artifacts, lacking spatial context information, requiring the number of regions/clusters *a priori* and being sensitive to initial values of the centroids, and often converging to local minima [Zhang and Jiang, 2009; Wei-Yi et al., 2010].

Because of the drawbacks stated above, improved versions of soft clustering algorithms could be highly advantageous. As a proof of principle we tested DifFUZZY on a standard test image and demonstrated that DifFUZZY is able to identify the starfish (also known as sea star) after coarse-graining the image (Fig. 6.2). A less coarse mesh would give a better identification of the regions, but would be more computationally intensive.

After preliminary testing on image segmentation, DifFUZZY proved to be less sensitive to noise (data not shown) than FCM and presented the advantage over FCM of not becoming trapped in local optima or requiring the number of regions *a priori*. Instead, the minimum number of pixels in a region needs to be specified (parameter M), and this parameter can also be automatically determined.

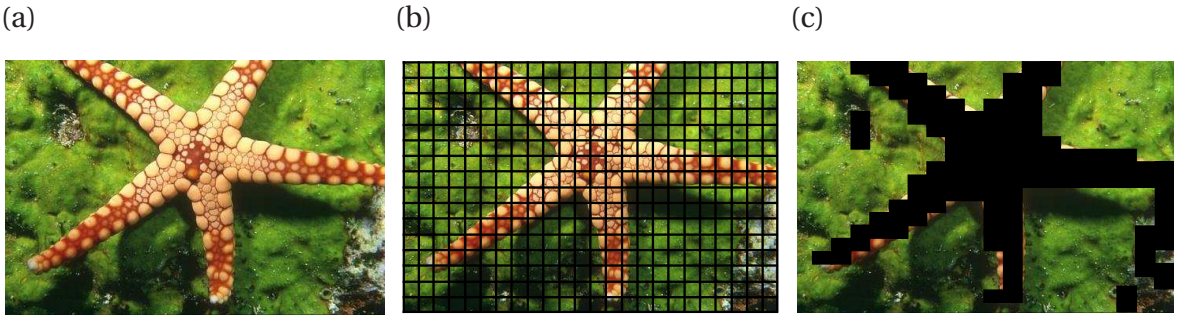


Figure 6.2: DifFUZZY on image segmentation. (a) Original image of starfish. (b) Mesh superimposed over original image. (c) Results of DifFUZZY segmentation of the sea star image (M = 80). Black denotes one of the clusters identified, the other one is the background.

It is possible to further improve this technique by adding spatial constraints so that pixels that are near in the image will have a larger probability of belonging to the same cluster than pixels that are further apart. In the example provided clustering was performed only using the RGB colour information (where each data point specifies the red, green and blue intensities of the colour at a given pixel), hence identifying some darker regions (rocks) as part of the starfish. Such constraints could also involve diffusive distances over the image. One possibility could be to identify sub-clusters whose regions are not contiguous. Edge identification could also be performed by identifying pixels whose membership values are intermediate and that form narrow bands. Furthermore, pre-processing of the image could be used to ensure a normalisation of the colours.

Medical imaging could also be used for validating cellular models. Series of images obtained from biological experiments could be segmented and compared to model predictions in an automated fashion, thereby making evaluation of model variants efficient and rigorous [Anderson et al., 2008].

6.2.3 Clustering time series

Most clustering research has been carried out on static datasets [Singhal and Seborg, 2005]. Very few clustering algorithms have been developed for time series data, and most use the raw data and adapt the distance measure for one suitable to deal with time series [Liao, 2005]. Most of the biological time series datasets are short, containing less than 8 time points but thousands of data points, such as genes in microarray data, which makes their clustering difficult, and often results in random patterns [Ernst et al., 2005].

Euclidean distance, often used to measure similarities between time series, is sensitive to the presence of outliers and noise, both of which are common in microarray data and other high-throughput experimental results. One possible solution to this is to pre-process the data, removing outliers and normalising the time series, which requires knowledge of the domain (stock prices, gene expression, etc.). Other solutions include the use of a diffusive distance, which is less sensitive to outliers and noise [Wittman, 2002].

Furthermore, when using Euclidean-based clustering algorithms such as K-means and FCM, the concept of cluster centroids becomes unclear if there are time series of different lengths [Liao, 2005]. This would not be the case with a diffusion-based distance such as in DifFUZZY, which does not rely on cluster centroids to determine the distance of a data point to a cluster.

Recent research has shown that clustering time-series data can be meaningless in many cases due to the difficulties of clustering series that are auto-similar (scale-invariant or fractal-like signals) and with high levels of noise [Simon et al., 2006]. However, this is not the case when using kernel-density-based clustering methods mainly because of their noise elimination properties [Denton, 2005; Simon et al., 2006]. Since DifFUZZY shares some of the advantages of such methods and uses concepts of diffusion over graphs, it could also potentially give meaningful results when clustering time series data. We have not yet tested this idea, but we believe it is a promising application area.

6.2.4 Discussion

DifFUZZY's versatility has been demonstrated by its successful application by a number of researchers in a wide range of scientific areas, including signalling processing, phylogenetics, gene expression analysis and covert channels, among others (Section B.4).

We have also used DifFUZZY to study additional microarray datasets, in particular ones studying tumours from different types of brain tissues, whose results are being analysed by experimental collaborators.

There are several avenues we envision to take DifFUZZY forwards, such as applying it to medical image segmentation and time series data clustering, but we also think soft clustering could be the bridge that brings together the area of bioinformatics with that of data modelling, to use the wealth of data of high-throughput experiments in the embedded subcellular models to help generate hypotheses for experimentalists.

Another direction to take DifFUZZY forwards is to let its distance diffusion process run until reaching equilibrium, and thus developing a multi-resolution clustering approach, similar to Lambiotte's random walk work in networks, which allows the identification of larger and larger clusters (modules/hierarchies) as time is increased [Lambiotte et al., 2008; Lambiotte, 2010].

Appendices

Appendix A

Clustering Algorithms and Validation

A.1 Fuzzy C-means clustering

A.1.1 Minimisation of the objective function J

To minimise the function J, which is the sum of J_i, $i = 1,2,\dots N$, subject to the probabilistic constraint

$$\sum_{k=1}^{K} u_{i,k} = 1, i = 1,\dots N,$$

we use the Lagrangian multipliers method. Consider the auxiliary function $\Lambda = \sum_{i}^{N} \Lambda_i$, where Λ_i is of the form

$$\Lambda_i = \sum_{k=1}^{K} u_{i,k}^{q} \|\mathbf{x}_i - \widetilde{\mathbf{c}}_k\|^2 + \kappa_i \left(\sum_{k=1}^{K} u_{i,k} - 1 \right), \quad i = 1,\dots N, \tag{A.1}$$

where κ_i are artificial variables. It can be proven that the extrema of the original optimisation problem J are also stationary points of the function Λ [Li, 2008].

Therefore we differentiate $\Lambda = \sum_{i=1}^{N} \Lambda_i$, $i = 1,2,\dots N$ with respect to the unknowns of the function Λ ($u_{i,k}$, $\widetilde{\mathbf{c}}_k$ and κ_i) and set these derivatives to zero:

$$\frac{\partial \Lambda}{\partial u_{i,k}} = q\, u_{i,k}^{q-1} \|\mathbf{x}_i - \widetilde{\mathbf{c}}_k\|^2 + \kappa_i = 0, \tag{A.2}$$

$$\frac{\partial \Lambda}{\partial \widetilde{\mathbf{c}}_k} = -2 \sum_{i=1}^{N} u_{i,k}^{q} (\mathbf{x}_i - \widetilde{\mathbf{c}}_k) = 0, \quad \text{and} \tag{A.3}$$

$$\frac{\partial \Lambda}{\partial \kappa_i} = \sum_{k=1}^{K} u_{i,k} - 1 = 0. \tag{A.4}$$

Isolating $u_{i,k}$ from Eq. (A.2) we obtain

$$u_{i,k} = \left(\frac{-\kappa_i}{q \|\mathbf{x}_i - \widetilde{\mathbf{c}}_k\|^2} \right)^{\frac{1}{q-1}}. \tag{A.5}$$

Substituting the value of $u_{i,k}$ from Eq. (A.5) in Eq. (A.4), we can isolate the value of $(-\kappa_i)^{\frac{1}{q-1}}$ and obtain:

$$(-\kappa_i)^{\frac{1}{q-1}} = \frac{q^{\frac{1}{q-1}}}{\sum_{k=1}^{K} \frac{1}{\|\mathbf{x}_i - \widetilde{\mathbf{c}}_k\|^{\frac{2}{q-1}}}}. \tag{A.6}$$

Changing the index from k to l in Eq. (A.6) and substituting it into Eq. (A.5) we obtain the final equation for $u_{i,k}$:

$$u_{i,k} = \frac{1}{\sum_{l=1}^{K} \left(\frac{\|\mathbf{x}_i - \widetilde{\mathbf{c}}_k\|}{\|\mathbf{x}_i - \widetilde{\mathbf{c}}_l\|} \right)^{\frac{2}{q-1}}}. \tag{A.7}$$

Finally, isolating $\widetilde{\mathbf{c}}_k$ from Eq. (A.3), we obtain the following relationship:

$$\widetilde{\mathbf{c}}_k = \frac{\sum_{i=1}^{N} u_{i,k}^q \mathbf{x}_i}{\sum_{i=1}^{N} u_{i,k}^q}. \tag{A.8}$$

Appendix B

DifFUZZY

B.1 Random walk and diffusion

We show different interpretations of the diffusion process, which will help us to gain a better understanding of this phenomenon, in particular of the fact that diffusion can be thought of as a random walk and that a good approximation of the solution of the diffusion equation can be obtained by taking the first few eigenvectors of the Laplacian matrix.

The simplest form of the diffusion equation in 2-D is:

$$\frac{\partial f(x,y,t)}{\partial t} = D\left(\frac{\partial^2 f(x,y,t)}{\partial x^2} + \frac{\partial^2 f(x,y,t)}{\partial y^2}\right) = D\nabla^2 f(x,y,t), \tag{B.9}$$

where the probability density function $f(x,y,t)$ corresponds to the probability of finding a particle in the vicinity of the point (x,y) at time t, with x and y being the spatial variables and D is the diffusion constant.

For x and y ranging from 0 to 1, using a grid of 50×50 elements in that range (therefore $dx = dy = \frac{1}{50} = 0.02$), $D = 0.25$ and for a final time $t = 0.01$, assuming Dirichlet initial condition,

$$f(x,y,0) = \begin{cases} 1 & x = y = 0.5, \\ 0 & \text{elsewhere}, \end{cases} \tag{B.10}$$

and zero flux boundary conditions;

$$\frac{\partial f(x,y,t)}{\partial x} = 0, \text{ at } x = 0, \quad \frac{\partial f(x,y,t)}{\partial x} = 0, \text{ at } x = 1, \tag{B.11}$$

$$\frac{\partial f(x,y,t)}{\partial y} = 0, \text{ at } y = 0, \quad \frac{\partial f(x,y,t)}{\partial y} = 0, \text{ at } y = 1. \tag{B.12}$$

The steady state of the diffusion equation is uniform and the analytical solution of Eq. (B.9) on an infinite domain is:

$$f(x,y,t) = \frac{1}{\sqrt{4\pi D t}} \exp^{-\frac{(x-x_0)^2+(y-y_0)^2}{4Dt}} \tag{B.13}$$

and it is plotted in Fig. B.3 (a). Next, in Fig. B.3 (b), approximately the same solution is plotted, obtained by the finite differences method using $s = \frac{t}{dt}$ time steps, where the duration of each time step is $dt = 0.00002$. The solution in Eq. (B.13) does not satisfy the boundary conditions in Eq. (B.11), but since the diffusion is run for a time short enough not to reach the boundaries, the results are comparable.

Another approach to solve the diffusion equation (also known as the heat equation) is by using random walks. In Fig. B.3 (c) we present results that were computed by averaging over one million random walks. At each iteration a single particle was placed at the centre of the grid $(\frac{m}{2}, \frac{n}{2})$ and with a given probability $p = \frac{D \cdot dt}{dx \cdot dy}$ it was allowed to jump to one of its four neighbouring cells $(\frac{m}{2}+1, \frac{n}{2})$, $(\frac{m}{2}-1, \frac{n}{2})$, $(\frac{m}{2}, \frac{n}{2}+1)$ or $(\frac{m}{2}, \frac{n}{2}-1)$, and with a probability $(1-4p)$ to remain in its place. The random walk is not propagated over the boundaries.

Finally, Fig. B.3 (c) shows another approximation of this diffusion process given by the first nine eigenvectors of the Laplacian matrix, which represents the diffusion map used by spectral clustering methods [Bah, 2008].

Similarly, we show, in Fig. B.4, results using the finite element method and random walk approximation of the diffusion process over an inverted U shaped lattice [Bah, 2008]. There we show that although the Euclidean distance between the starting point (red section) of the U, and the end point (darkest blue section) is similar to that from the red area to a mid-blue region (on the top of the inverted U), the diffusive distance to this second area is smaller than to the end of the shape.

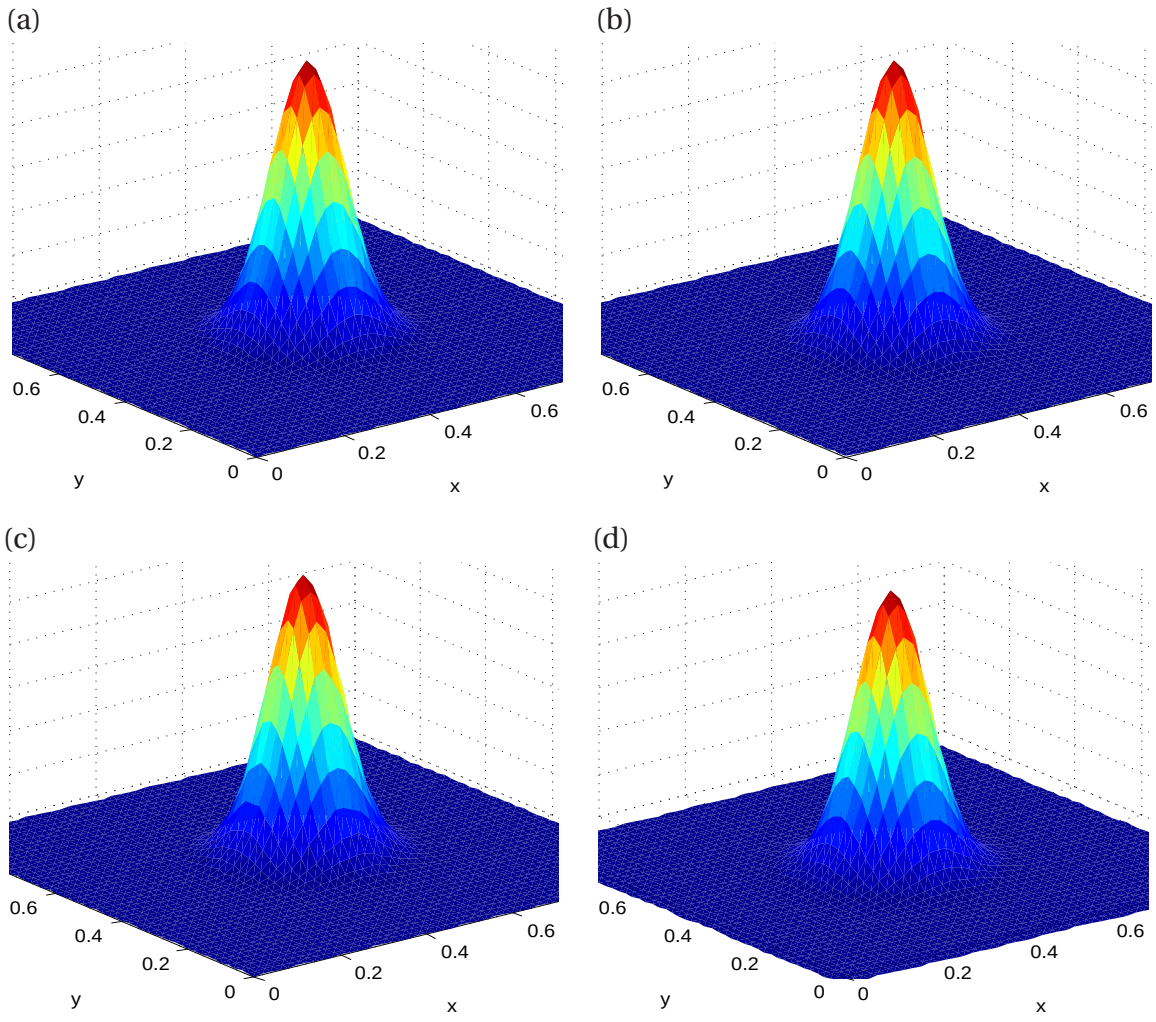


Figure B.3: Solution of the 2-D diffusion equation, Eqs. (B.9)-(B.11), on a square lattice for $t = 0.01$ and $D = 0.2499$. (a) Analytical solution, Eq. (B.13), (b) numerical approximation, (c) random walk approximation and (d) eigenvector approximation.

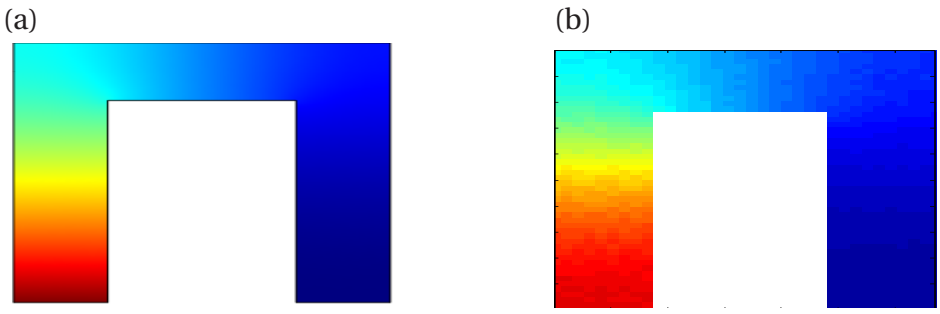


Figure B.4: Solution of the 2-D diffusion equation an inverted U lattice. (a) Finite element methods solution and (b) random walk approximation.

B.2 Schematic representation of DifFUZZY

In Fig. B.5 we show a schematic diagram representing the diffusion distance of the DifFUZZY algorithm, which is used to determine the membership degrees of soft data points (Eq. (3.13) in Section 3.1.3).

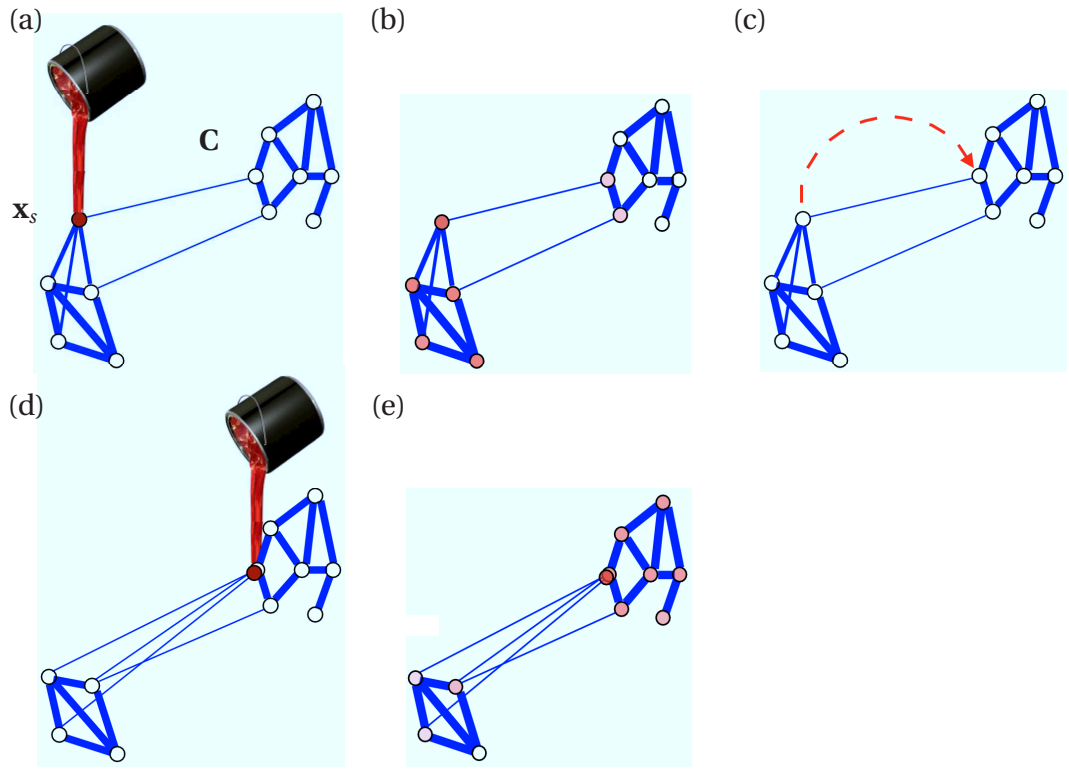


Figure B.5: Schematic diagram of the calculation of DifFUZZY's diffusion distance between a soft data point $\mathbf{x}_s$ and a cluster core **C**. The bottom left subnetwork of four nodes represents a second cluster core. In this cartoon network the diffusion process is illustrated as the diffusion of paint through pipes (edges). Steps: (a) start the diffusion over the network from the soft data point, (b) stop the diffusion process after a given number of time steps, (c) move the soft data point to the nearest point of cluster **C**, (d) initiate another diffusion process from the new position of the soft data point and (e) stop the diffusion process after the same number of time steps. The diffusion distance between $\mathbf{x}_s$ and **C** is then computed as the Euclidean distance between the diffusion profiles in (e) and (b).

B.3 Exploring the effect of DifFUZZY's internal parameters

In Fig. B.6 we present results from applying DifFUZZY to the *Elongated clusters* dataset using a reduced random walk time scale (internal parameter γ_1) and M values 40 and 60. This modification of the default parameter value allows the diffusion over the densely packed clusters to occur at a lower speed, hence allowing the soft data points in the diagonal cluster to belong to that cluster with a much higher probability than to the right cross-shaped cluster (shown in blue in Fig. B.6 (a) and in yellow and blue in Fig. B.6 (c)).

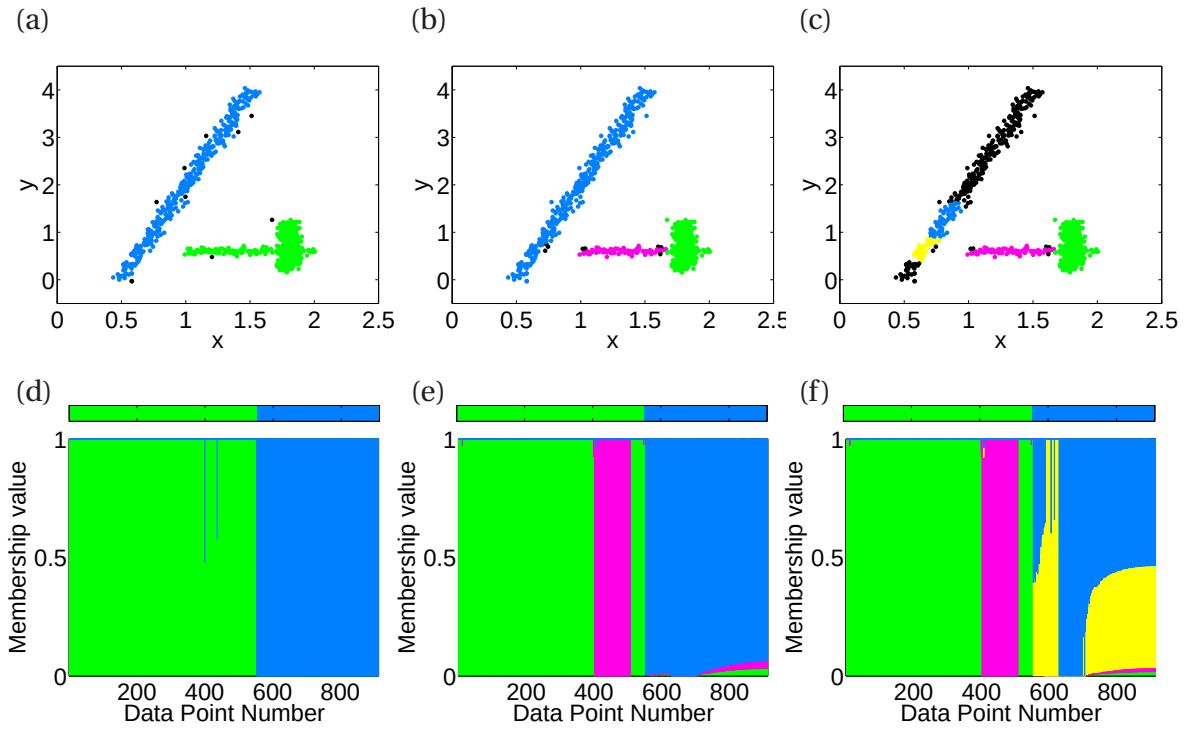


Figure B.6: DifFUZZY applied to the *Elongated clusters* dataset (Fig. 2.1 (b)). (a) HCT-plot, $z = 0.9$, M $= 300$, (b) HCT-plot, $z = 0.9$, M $= 60$, $\gamma_1 = 0.01$, (c) HCT-plot, $z = 0.9$, M $= 40$, $\gamma_1 = 0.01$, (d) membership values, M $= 300$, (e) membership values, M $= 60$, $\gamma_1 = 0.001$ and (f) membership values, M $= 40$, $\gamma_1 = 0.001$.

B.4 Implementation of DifFUZZY

DifFUZZY has been implemented both in Matlab and in C++ and is freely available for download from `http://www.maths.ox.ac.uk/cmb/diffuzzy`. The Matlab version takes advantage of the highly optimised full and sparse matrix operations and other efficient built-in functions provided by the Matlab framework. The time complexity of each substep of the DifFUZZY algorithm is

Main step	Sub-step	Complexity
1. Identification of cluster cores	Finding σ^*	$\Theta\left((\overline{\sigma}+d)N^2\right)$
	Reordering of cluster labels	$\Theta\left(N^2\right)$
	Assigning core points	$\Theta\left(KN^2\right)$
	Step 1 overall complexity	$\Theta\left((\overline{\sigma}+d+K)N^2\right)$
2. Auxiliary matrices computation	Finding β^*	$\Theta\left((\overline{\beta}+d)N^2\right)$
	Computation of **D**, **W** and **P**	$\Theta\left(dN^2\right)$
	Step 2 overall complexity	$\Theta\left((\overline{\beta}+d+K)N^2\right)$
3. Fuzzy membership values	Finding auxiliary parameter a	$\Theta\left(N^3\right)$
	Distances calculation	$\Theta\left(N_{\text{und}}dN\right)$
	Memberships calculation	$\Theta\left(N_{\text{und}}KN^3\right)$
	Step 3 overall complexity	$\Theta\left(N_{\text{und}}KN^3\right)$
	DifFUZZY overall complexity	$\Theta\left(N_{\text{und}}KN^3\right)$

Table B.1: Time complexity of DifFUZZY. The overall complexity of DifFUZZY is polynomial, $\Theta\left(N_{\text{und}}KN^3\right)$, where N is the number of data points, and N_{und} and K are the number of soft points and the number of clusters found by DifFUZZY. $\overline{\sigma}$ denotes the number of σ values that are used to evaluate the function $F(\sigma)$, described in Section 3.1.1. This is 200 by default. Likewise, $\overline{\beta}$ denotes the number of β values used to evaluate $L(\beta)$ (Section 3.1.2), which is 50 by default. We assume that $(\overline{\sigma}+\overline{\beta}+d) = O(N_{\text{und}}KN)$, where d is the dimensionality of the data.

Using Google Analytics [Ledford et al., 2010] we have been able to track and measure the impact of DifFUZZY and its dedicated website, averaging over hundred and fifty visits to its website per month, of which 30% corresponds to new visitors and the rest to returning ones. As of May 2012 the DifFUZZY package has been downloaded nearly two hundred times from over twenty countries since April 2010 (the date when the tracking system was set up).

The method is currently being used by researchers and graduate students across a wide spectrum of disciplines and a number of countries, in applications ranging from integrating DifFUZZY into a signal processing framework which breaks new ground in the discovery of patterns in bioinformatic signals (Dr. Tim Hutcheson, Research Associate Florida Institute for Human and Machine Cognition), improving building phylogenetic trees (Dr. Steven Kelly, Oxford Centre for Integrative Systems Biology), demodulating frequencies of operations in timing covert channels (graduate thesis "Practical Considerations in Covert Channels" at the University of Pennsylvania), to plant gene expression analysis at Dongguk University in Seoul and to clustering research (thesis titled "Web based Fuzzy C-means Clustering software", IASRI in New Delhi), among several others.

B.5 DifFUZZY: testing and validation

B.5.1 Benchmark datasets

	Class	x	y
1	1	53	153
2	1	55	139
3	1	57	153
4	1	59	147
5	1	61	154
6	2	78	98
7	2	83	100
8	3	29	75
9	3	30	60
10	3	31	77
11	3	32	55
12	4	58	18
13	4	62	20
14	2	100	124
15	2	100	130
16	2	102	120
17	2	110	118
18	2	110	122
19	1	40	140
20	1	40	142
21	1	40	150
22	1	43	148
23	1	46	140
24	1	46	146
25	3	33	61
26	3	35	62
27	3	39	75
28	1	46	155
29	1	48	138
30	1	50	150
31	4	67	30
32	4	69	22
33	4	69	24
34	4	71	20
35	3	10	77
36	3	12	56
37	3	15	48
38	2	75	100
39	4	75	33
40	4	78	28
41	4	79	18
42	4	80	22
43	3	5	52
44	3	7	60
45	2	98	127
46	2	98	122
47	1	36	157
48	1	37	137
49	1	38	155
50	3	18	75
51	3	20	60
52	3	21	63
53	3	23	75
54	3	25	57
55	3	28	55
56	2	112	118
57	2	115	125
58	2	117	120
59	3	15	70
60	3	15	85
61	4	71	25
62	4	72	9
63	2	118	118
64	1	59	128
65	1	62	135
66	1	64	138
67	4	64	26
68	4	67	23
69	2	85	120
70	2	88	135
71	2	96	125
72	1	30	145
73	1	35	150
74	1	35	140
75	4	83	25

Table B.2: *Ruspini* dataset.

B.5.2 ROC curves

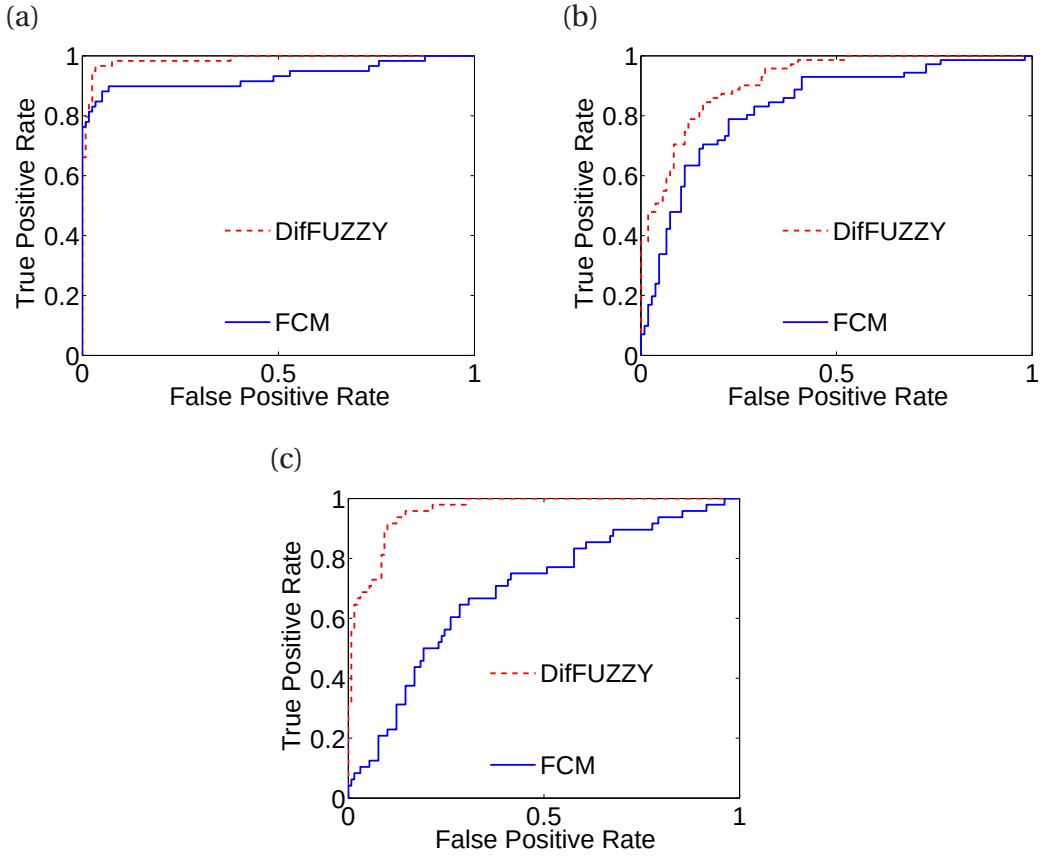


Figure B.7: DifFUZZY and FCM ROC curves for the *Wine* dataset (M = 12): (a) Nebbiolo, (b) Barberas and (c) Grignolino grape cultivars.

B.5.3 Random clustering

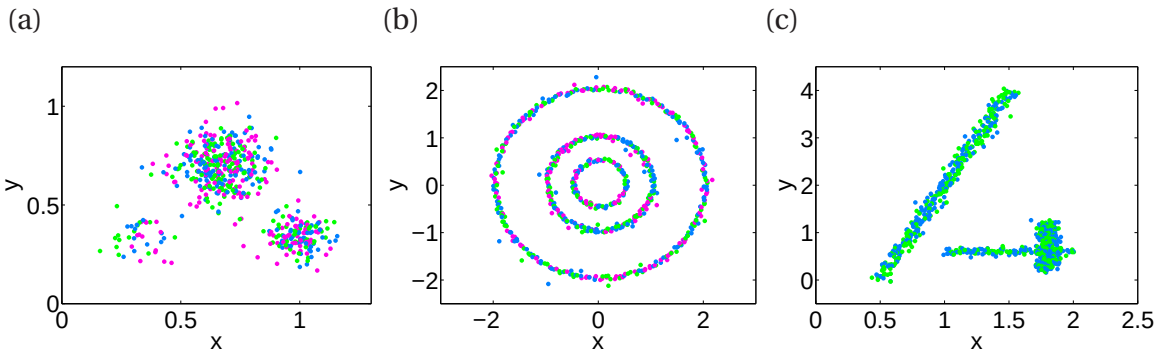


Figure B.8: Random clustering of synthetic test datasets. (a) *Globular clusters* dataset, $K = 3$. (b) *Concentric rings* dataset, $K = 3$. (c) *Elongated clusters* dataset, $K = 2$. Different colours represents the different cluster assignments.

Clustering Clinical Data: Severe Malaria

C.1 The Gambia dataset

Variable	Description	Values	Format	# MV	% MV
acmuscle	accessory muscle	1 = yes, 0 = no	binary	66	2.26
agemth	months of age	0 – 180	numeric	11	0.38
coma	comatose	1 = yes, 0 = no	binary	97	3.33
comatime	hours of coma	0 – 96	numeric	1449	49.64
convadm	fits in admission	1 = yes, 0 = no	binary	1907	65.42
convuls	presence of fits	1 = yes, 0 = no	binary	159	5.45
deepbr	deep breathing	1 = yes, 0 = no	binary	1888	64.77
diarea	diarrhoea symptoms	1 = yes, 0 = no	binary	67	2.30
falive	father alive	1 = yes, 0 = no	binary	24	0.82
fever	presence of fever	1 = yes, 0 = no	binary	38	1.30
fevertime	hours/days of fever	1 – 72	numeric	254	8.71
hbtotal	haemoglobin level	0.59 – 15.24	continuous	62	2.13
malive	mother alive	1 = yes, 0 = no	binary	14	0.48
numberfits	number of fits	0 – 20	numeric	447	15.33
post	abnormal posture	1 = yes, 0 = no	binary	2243	76.95
totalbcs	Blantyre coma score	0 – 5	numeric	5	0.17
usleepy	unusually sleepy	1 = yes, 0 = no	binary	51	1.75
vomit	vomit symptoms	1 = yes, 0 = no	binary	60	2.06

Table C.3: The Gambia dataset variables summary (extract). # MV and % MV indicate the number and percentage of missing values, respectively.

C.2 Feature selection

C.2.1 DifFUZZY MV plots

C.2.1.1 CM and RD

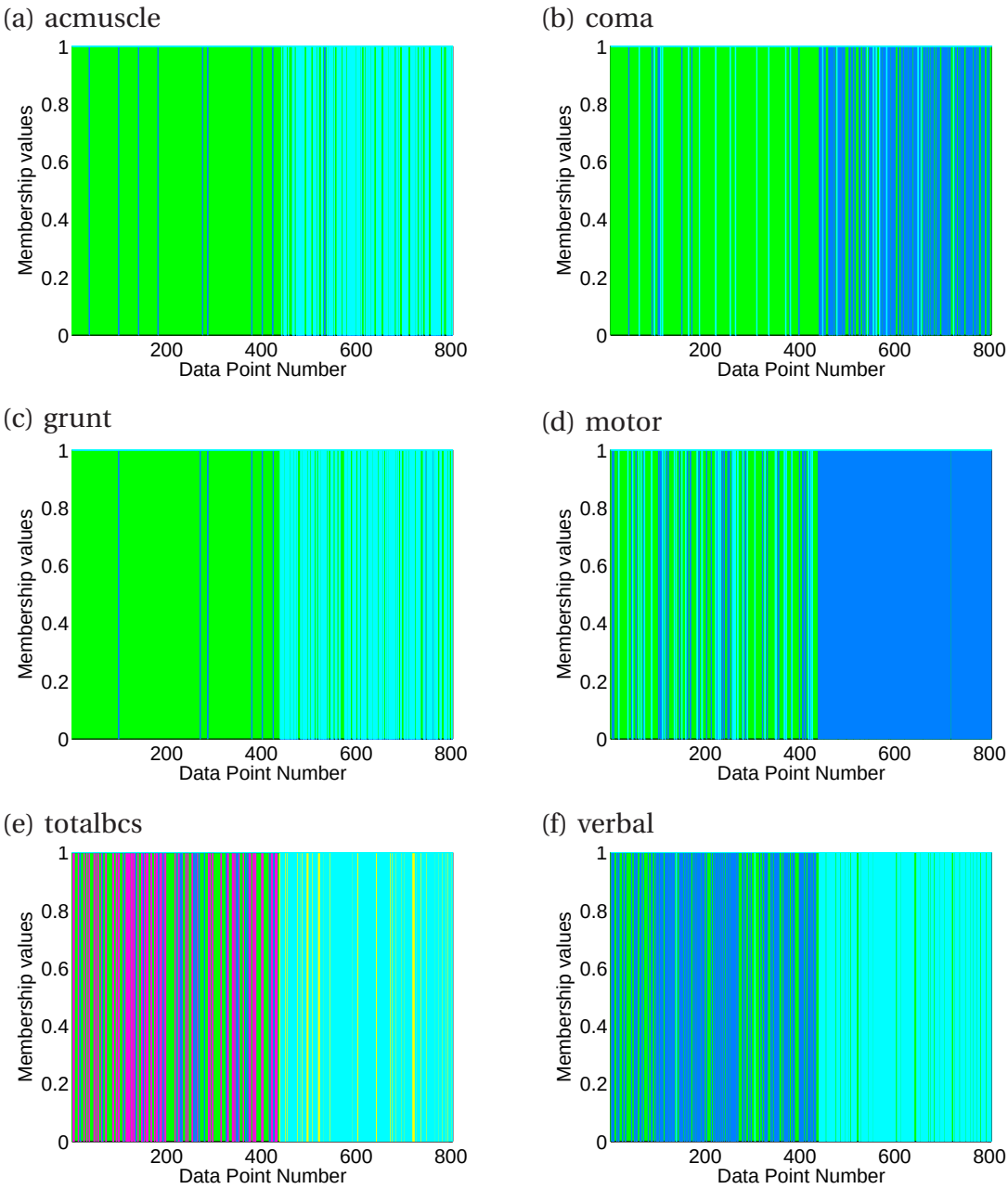


Figure C.9: DifFUZZY MV plots on selected variables of The Gambia dataset (M=10). Data points up to number 438 describe CM, those after 438 describe RD.

C.2.1.2 CM, CMRD and RD

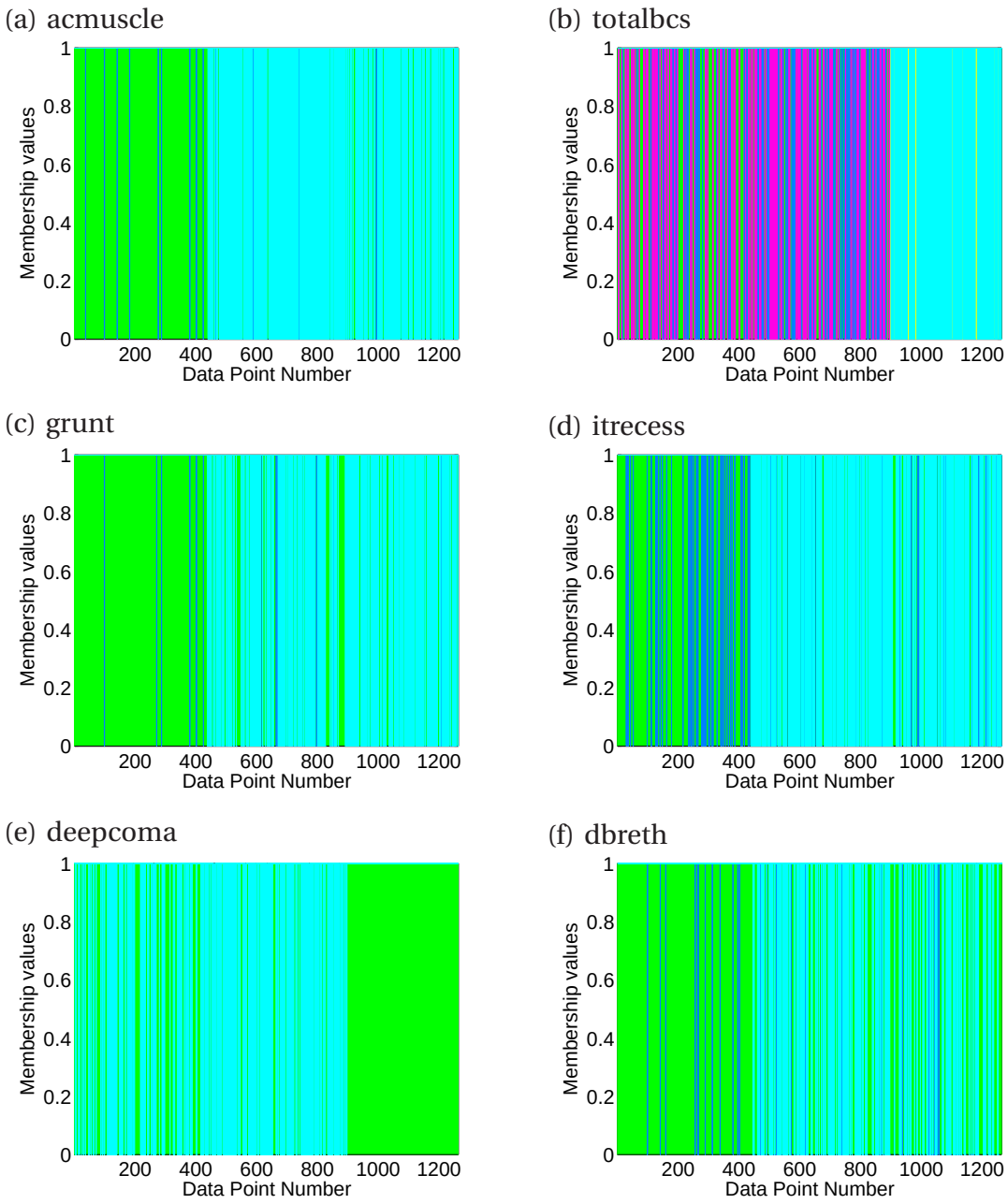


Figure C.10: DifFUZZY MV plots on selected variables of The Gambia dataset (M=10). Data points up to number 438 describe CM, those after 898 describe RD and those between 438 and 898 describe CMRD.

C.3 Thresholding

Fig. C.11 shows the plot of the size of the largest cluster size against the distance thresholds for The Gambia dataset and Fig. C.12 shows the clustering coefficient versus threshold for the SMAC dataset.

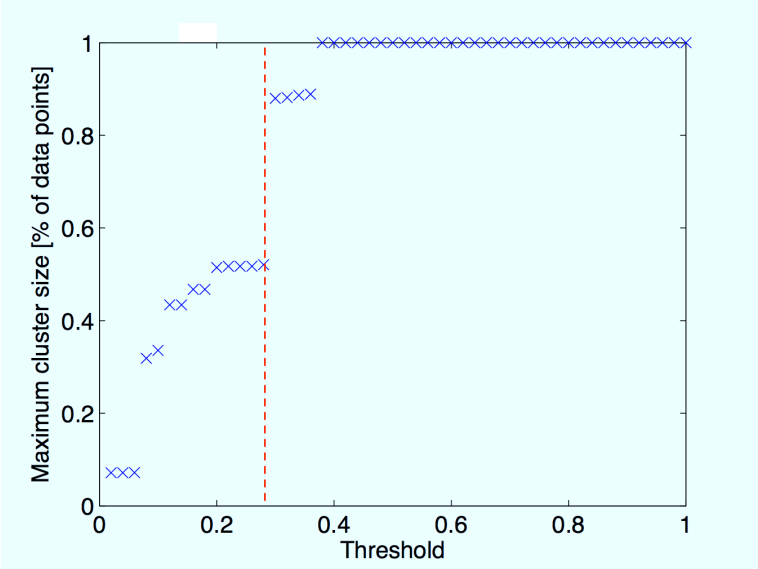


Figure C.11: Size of largest cluster plotted against threshold. For a threshold higher than 0.28, the maximum cluster size is larger than 88% of the data points. Given that such a partitioning with one very large cluster is not meaningful for our clustering purposes, we will not consider thresholds beyond 0.28 for the network visualisation.

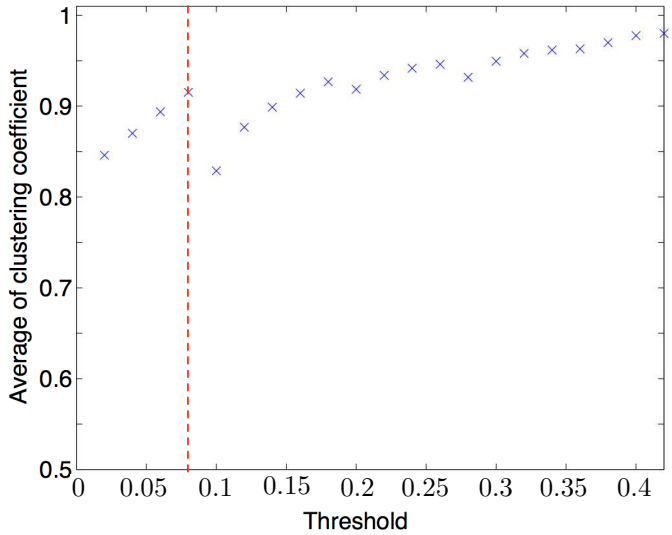


Figure C.12: Clustering coefficient versus the threshold for the SMAC dataset. The threshold which yields the highest clustering coefficient, shown by a red dotted line, is 0.08 and a larger cluster less than 60% of the total data points.

C.4 Network validation

C.4.1 SMAC dataset

	CM	CMRD	RD	CMSMA	CMRDSMA	RDSMA	SMA	Others	**Total**
Banjul	144	75	410	29	29	163	612	1933 (57%)	**3395**
Blantyre	109	102	238	36	47	86	389	4356 (81%)	**5363**
Kumasi	396	284	754	47	109	483	1047	3824 (55%)	**6944**
Kilifi	296	314	942	26	75	232	421	4603 (67%)	**6909**
Lambaréné	34	28	38	13	17	15	200	1450 (81%)	**1795**
Libreville	88	51	68	20	33	60	347	1033 (61%)	**1700**
SMAC	1067	854	2450	171	310	1039	3016	17199 (66%)	**26106**
Gambia	438	460	365	56	111	234	264	987 (34%)	**2915**

Table C.4: Number of cases of severe malaria syndromes in the SMAC (per site and global) and The Gambia datasets.

C.4.1.1 SMAC site-specific networks

(a) Lambaréné

(b) Libreville

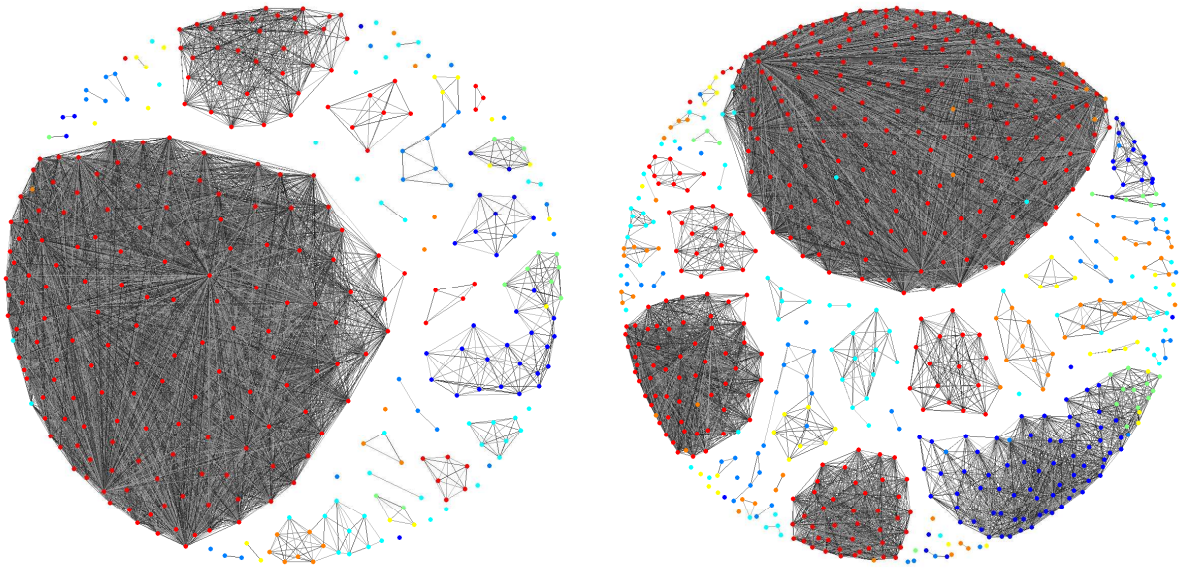


Figure C.13: Networks for the Lambaréné and Libreville sites of the SMAC dataset.

Future Directions

D.1 Cellular modelling

D.1.1 Cell-vertex model

We chose a cell-based model instead of a continuous approach because we are dealing with a small number of cells (in the human a single crypt is typically 40-60 cells in length and contains 1000-4000 epithelial cells [Arnki et al., 1996; Nowak et al., 2002], while in the mouse the crypt is 20-25 cells in height and 16 cells in circumference [Li et al., 1994]) very much at the lower limit for validity of a continuum approach. Moreover, we are interested in representing the migration and cell-cell interaction of individual cells. More specifically, we opted for a cell-vertex model given that it can describe cell movement in a realistic fashion as well as giving convincing cell shapes [Brodland, 2004; Walter, 2009].

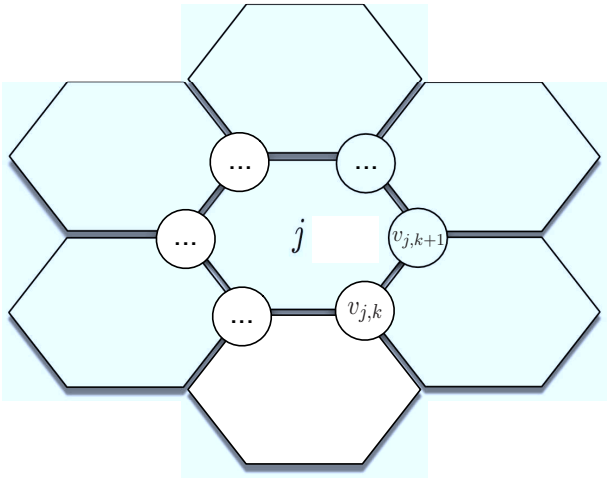


Figure D.14: Schematic representation of cell j with its defining vertices, $v_{j,k}, v_{j,k+1}, \ldots$ and surrounding cells.

The model developed is a dynamic lattice-free model, where each cell is represented by a given number of vertices which define the cell perimeter (see Fig. D.14). We used Nagai and Honda [2001]'s model in which the equations of motion for the vertices adjust cell shapes and direction of migration at each time step.

D.1.1.1 Implementation of the cell-vertex model

Below we outline the algorithm used to simulate a cell population using the cell-vertex model. At every time step the code checks each cell's area

for $i := 1 \rightarrow TS$ (for every time step) **do**
  for $j := 1 \rightarrow M$ (go through all cells) **do**
    if cell i has reached the end of the cell cycle **then**
      divide cell
      sort vertices in mother and daughter cells
    end if
    for $k := 1 \rightarrow M_j$ (go through all vertices of j^{th} cell) **do**
      compute the equations of motion for vertex $v_{j,k}$
      find neighbouring cell by edge $\overline{v_{j,k}\ v_{j,k+1}}$, $\mathrm{N}_{j,k}$
      if length of edge $\overline{v_{j,k}\ v_{j,k+1}} < T_{\alpha_j,\alpha_{N_{j,k}}}$ **then**
        apply T1 swap (see next subsection) over edge $\overline{v_{j,k}\ v_{j,k+1}}$
      end if
    end for
  end for
end for

where TS stands for number of time steps, M the total number of cells in the simulation, α_j the type of j^{th} cell, $A_{j,N}$ is the target area of j^{th} cell and $T_{\alpha_j,\alpha_{N_{j,k}}}$ is the threshold for T1 swaps (defined below in Section D.1.1.2) between cells of type α_j and $\alpha_{N_{j,k}}$. $N_{j,k}$ represents the neighbouring cell of the j^{th} cell by the edge $\overline{v_{j,k}\ v_{j,k+1}}$.

Every time step represents 7.2 seconds ($\frac{1}{500}$ hr) of the simulation (this time step was chosen to be small to guarantee convergence and stability). In the simulation results we will include the time instead of number of time steps run.

D.1.1.2 T1 vertices swaps

If two neighbouring cells become sufficiently separated such that their shared edge is very small, the vertices are allowed to move and modify the connection with their neighbours such that they will no longer be adjacent neighbours as in Fig. D.15. Such rearrangements have been seen in epithelial tissues [Nagai and Honda, 2001], and allow cells to have more movement by letting them lose and regain new neighbours/connections.

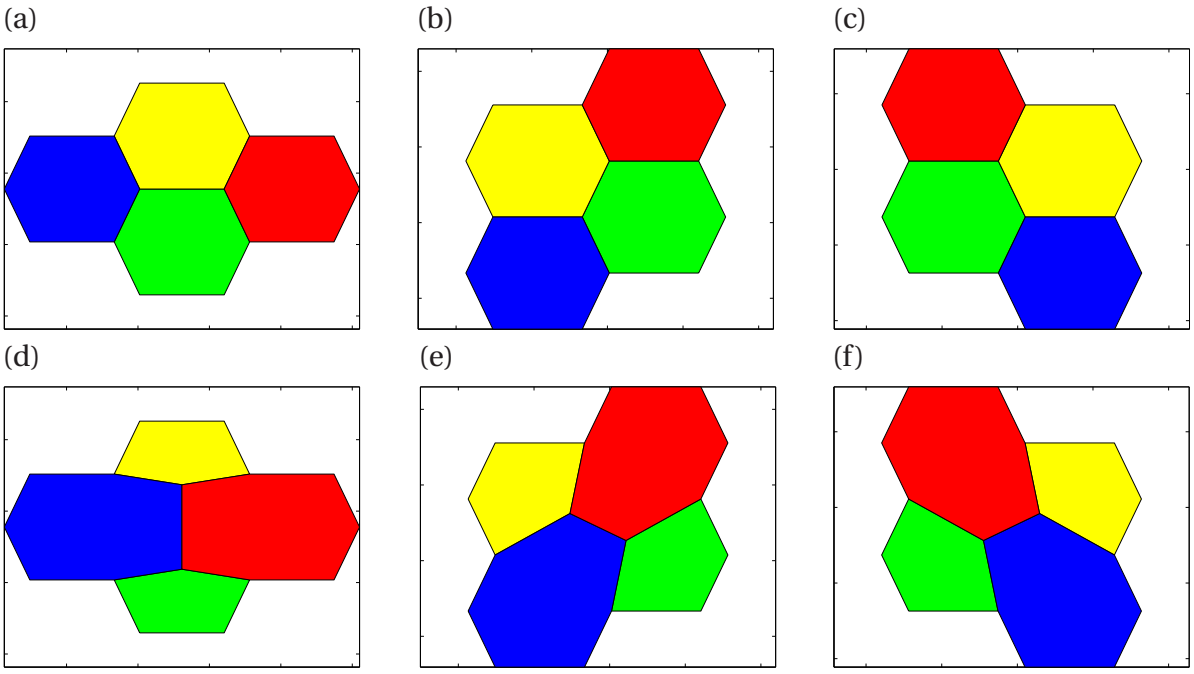


Figure D.15: T1 swap of edge between the yellow and green cells depicted in (a), (b) and (c). (a) and (d): horizontal edge, (b) and (e): diagonal edge and (c) and (f) back diagonal edge. (a), (b) and (c) show the original cells while (d), (e) and (f) show the cells after the T1 swap of the edge between the yellow and green cells. T1 swaps will only occur between very small edges, these examples are just for illustrative purposes.

There are three different T1 swaps possible on a hexagonal grid: horizontal vertex, diagonal and back diagonal (see Fig. D.15). During simulations swaps only occur when the length of an edge is sufficiently small.

The results in Fig. D.15 were obtained using a code written in Matlab, as an exercise to understand the implications of the model, how the vertices move, and how they are allowed

to swap among neighbour vertices. However, for long simulations, our Matlab code proved too slow, it was not easily extensible or tested, which is why we decided to use Chaste, a simulation package developed at the Department of Computer Science at the University of Oxford[1] which is collectively owned and is in constant development and refactoring using agile programming techniques. We describe Chaste in more detail in Section D.1.3.

D.1.1.3 Cell sorting

We start by testing if the cell-vertex model can reproduce cell sorting experiments. For simplicity we do not allow cells to proliferate, they adapt their vertices according to the sum of deformation energy, cell-cell adhesion energy and energy surface tension between the edges, which is calculated using the cell perimeter and a natural perimeter of the cell as in Eq. (D.18), where the natural perimeter is the perimeter that corresponds to the target area of the cell, restricting cells from deviating from an average cell area. Cell-cell adhesion energy is calculated using the length between the edge of neighbouring cells and the cell-adhesion parameter γ, which in turn depends on the cell types.

In the following figures (Figs. D.16-D.17) it is possible to observe differences between differential adhesion in a monolayer with two different populations. For a monolayer composed equally of two cell types initially randomly positioned, a checkerboard pattern evolves (Fig. D.16). In the case of preferential adhesion between cells of the same type (Fig. D.17), cells become more rounded, but do not form clusters: the islands of cells do not join into a single aggregate, which agrees with experimental results [Cortina et al., 2007], where the number of islands is related to how likely it is for cells of the same type to encounter each other. When adding cell proliferation[2], same-type cell encounters are facilitated, and we obtain one single and compact cluster (Fig. D.18) resembling a compartmentalised tumour [Clevers and Batlle, 2006].

[1] The code can be downloaded under the LGPL 2.1 open source licence from `http://www.cs.ox.ac.uk/chaste/download.html`.
[2] Details about the implementation of the cell cycle model are included in Section D.1.2.

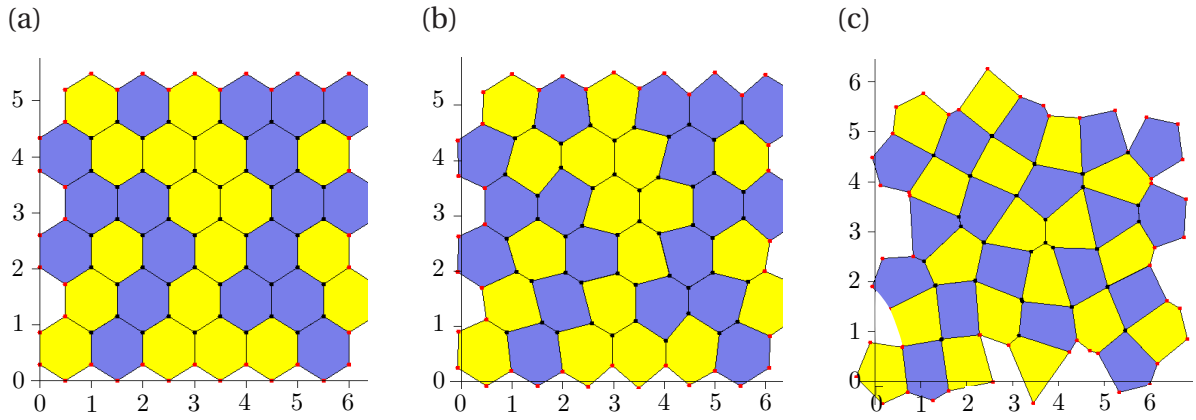


Figure D.16: Simulation of a monolayer of two populations of cells, showing preferential adhesion between pairs of cells from different populations (blue with yellow), with adhesion parameters $\gamma_{y,y} = \gamma_{b,b} = 100 \cdot \gamma_{y,b} = 1.0$. Starting from a random layout of cells, half of which are blue, the cells reorganise themselves to acquire a checkerboard pattern, maximising the contact length between yellow and blue cells. Times (a) t=0.0, (b) t=0.1 and (c) t=0.489.

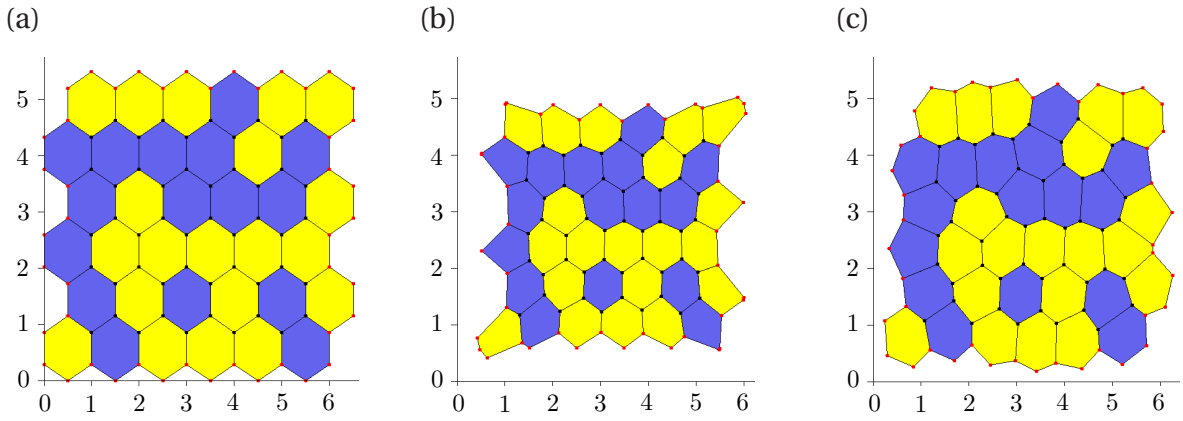


Figure D.17: Simulation of a monolayer of two populations of cells, showing preferential adhesion between pairs of cells from the same population (blue with blue and yellow with yellow), with adhesion parameters $\gamma_{y,y} = \gamma_{b,b} = 0.01$, $\gamma_{y,b} = 1.00$. Starting from a random layout, the cells become more rounded, but given that cell vertices are only affected by adjacent neighbours, the cells do not sort out. Times (a) t=0.0, (b) t=0.24 and (c) t=1.0.

We then run simulations starting from two proliferating cells of a different type to the rest, located at the middle of a monolayer. Using different adhesion parameters, in one case with preferential adhesion of cells of the same type, and in the other for cells of different type, we expect to find compact clusters and a salt-and-pepper pattern to emerge respectively. (see Figs. D.18 and D.19).

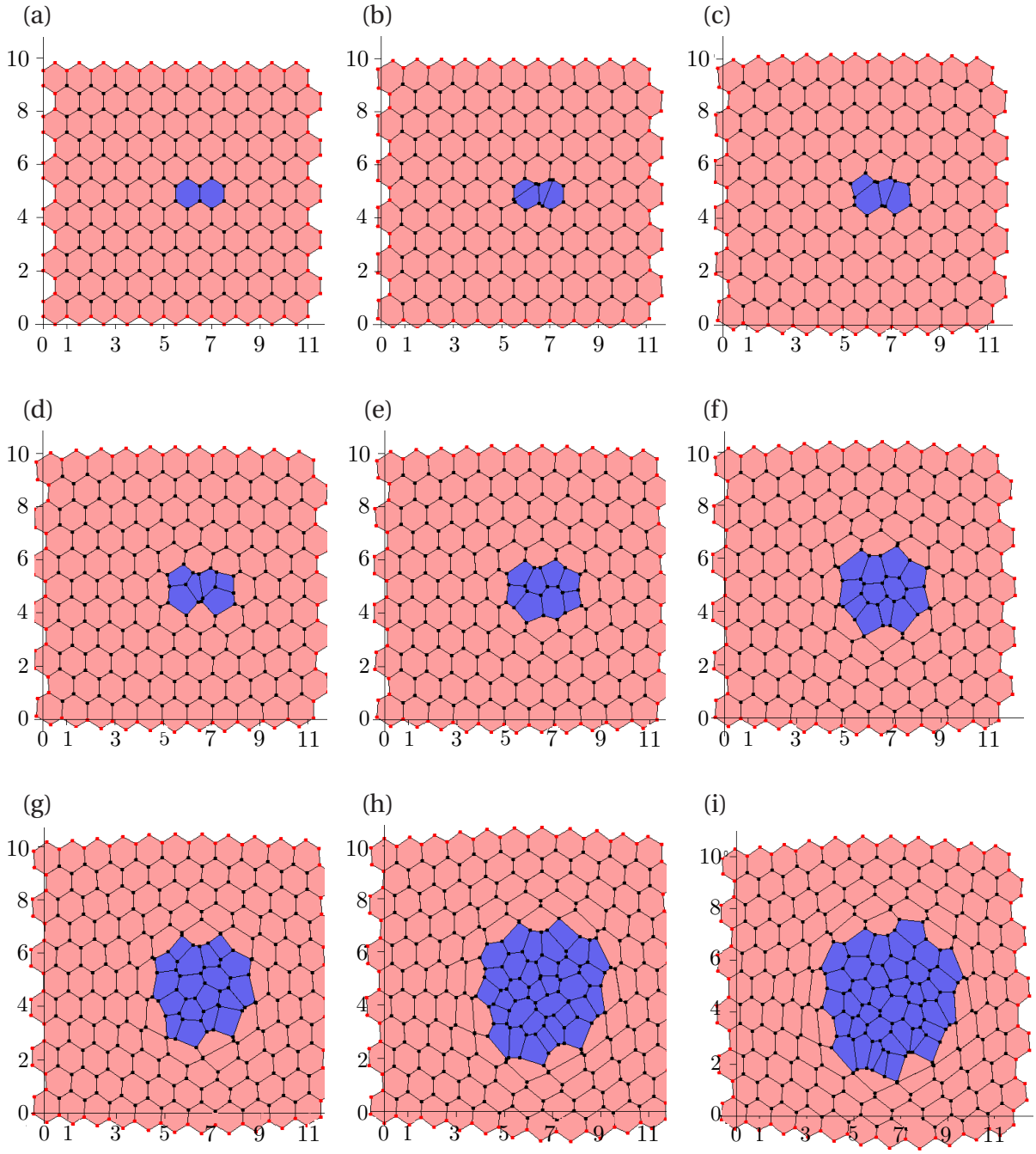


Figure D.18: Monolayer of two populations of cells, showing preferential adhesion between pairs of cells from the same populations (blue with blue and pink with pink), with adhesion parameters $\gamma_{b,b} = \gamma_{p,p} = 0.01$, $\gamma_{b,p} = 1.00$. Starting from an adjacent pair of proliferating blue cells, the newborn blue cells organise themselves in a compact cluster, resulting in a small total contact perimeter between pink and blue cells. Times (a) t=0.0, (b) t=0.06, (c) t=0.18, (d) t=0.52, (e) t=1.042, (f) t=1.986, (g) t=2.652, (h) t=3.936 and (i) t=4.48.

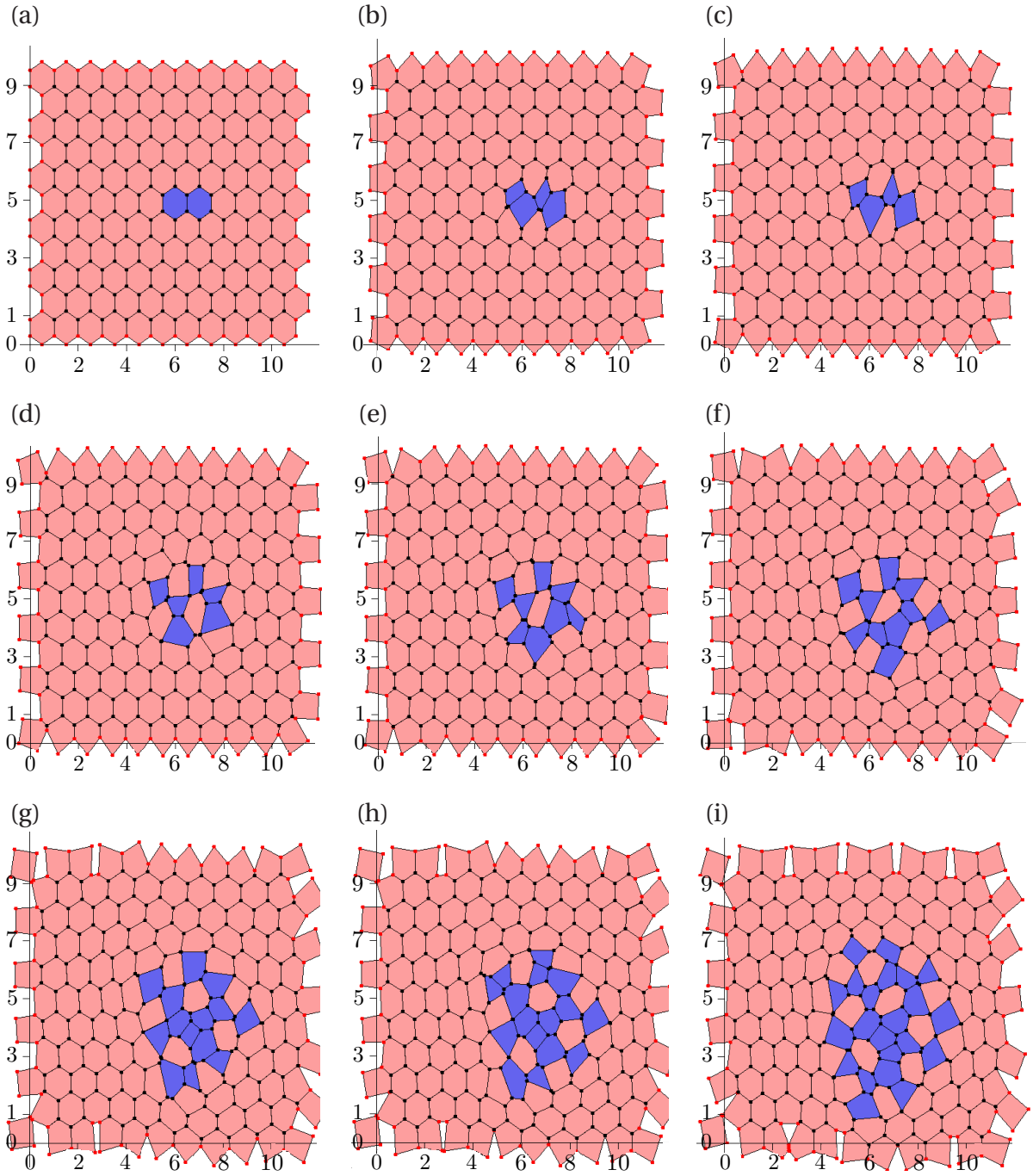


Figure D.19: Monolayer of two populations of cells, showing preferential adhesion between pairs of cells from different populations (blue with pink), with adhesion parameters $\gamma_{b,b} = \gamma_{p,p} = 3.0$, $\gamma_{b,p} = 0.01$. Starting from a pair of adjacent proliferating blue cells, the newborn blue cells organise themselves in a salt-and-pepper pattern, attaining a large total contact perimeter between pink and blue cells. Times (a) t=0.0, (b) t=0.14, (c) t=0.26, (d) t=0.60, (e) t=0.942, (f) t=1.51, (g) t=2.14, (h) t=2.394 and (i) t=3.0.

The proliferative driving force helps to create the compact cluster which was not attainable by only having cells adjust their vertices by minimising the surface free-energy (Fig. D.17). This is because the movement of cells through vertex rearrangement alone is limited and this force only acts locally.

D.1.2 The Crypt Model

The crypt is modelled as a cylindrical 2-D sheet of cells with periodic boundary conditions along the vertical edges. Through the base of the crypt no cells are removed, only cells which reach the top (lumen) are shed off [Osborne et al., 2010].

The crypt is represented by three cell types, stem (S), differentiated (D) and transit amplifying (TA). Stem cells are pinned in the bottom of the crypt.

We used a stochastic cell cycle model, in which the total duration of the different phases (G_1, S, G_2 and M) is sampled from a normal distribution with mean 17±1 h for stem cells and 13±1 h for transit cells [Meineke et al., 2001]. In Fig. D.20 we show the transition of cell types: when a stem cell divides it creates another stem cell and one transit amplifying generation one TA1. TA1 cells, on the other hand divide to create two TA2 cells (transit amplifying second generation). Each TA2 cell gives rise to two TA3 cells and each TA3 cell divides to create two TA4 cells. All this fourth generation of transit cells will divide to create two fully differentiated cells which will no longer divide [Wong et al., 2010].

The different cell types have different preferential adhesion between themselves and with other cell types which seem to depend on the level of Eph activation: low levels of EphB activation promote cell adhesion while high levels reduce it [Batlle et al., 2002]. Cells in the bottom of the crypt appear to have the highest EphB expression which decreases as cells reach the top, and ephrin ligands show the opposite distribution [Wong et al., 2010; Batlle et al., 2002].

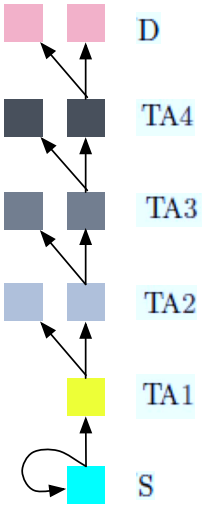


Figure D.20: Model's cell type transitions.

The value of the adhesion constant for every pair of cell types is given in the matrix **J**, Eq. (D.14). The modelled scenario is the one of differential adhesion, where the same cell types adhere preferentially to each other. The values, which represent the adhesion-free energy per unit contact area between cells, based on the experimental findings of Wilkinson [2003] and Poliakov et al. [2004], are those used by Wong et al. [2010]:

$$\mathbf{J} = \begin{array}{c} \\ \text{D} \\ \text{TA4} \\ \text{TA3} \\ \text{TA2} \\ \text{TA1} \\ \text{S} \end{array} \begin{array}{c} \begin{array}{cccccc} \text{D} & \text{TA4} & \text{TA3} & \text{TA2} & \text{TA1} & \text{S} \end{array} \\ \begin{bmatrix} 5 & 15 & 30 & 40 & 40 & 45 \\ 15 & 10 & 20 & 30 & 35 & 40 \\ 30 & 20 & 15 & 20 & 30 & 35 \\ 40 & 30 & 20 & 10 & 15 & 20 \\ 40 & 35 & 30 & 15 & 5 & 12 \\ 45 & 40 & 35 & 20 & 12 & 2 \end{bmatrix} \end{array} \tag{D.14}$$

Small values, such as those of same-cell contact, mean a stronger adhesion, while the ones between different types of cells are larger, representing a weaker adhesion. The surface free energy between S and D cells is large, i.e. they do not show a strong cell adhesion. These cell types are the ones further away in the crypt from each other, S cells are located in the bottom and D in the top [Wong et al., 2010].

The values and references of parameters used in the crypt cell-vertex model, chosen to ensure a quasi-steady state for the crypt (balance between cellular birth rate and cellular shedding rate) are included in Table D.5.

Parameter	Description	Value	Reference
$A_{j,N}$	Target area of j^{th} cell	$\frac{\sqrt{3}}{2}\left(\text{cell lengths}^2\right)$	Walter [2009]
$C_{j,N}$	Target perimeter of j^{th} cell	$2\sqrt{\pi A_{j,N}}\left(\text{cell lengths}\right)$	Walter [2009]
η_i	Drag over vertex v_i	$1\left(\frac{\text{Kg}}{\text{hr}}\right)$	Osborne et al. [2010]
κ	Elastic constant	$10\left(\frac{\text{Kg}}{\text{hr}^2\text{cell lengths}^4}\right)$	Osborne et al. [2010]
τ	Cell size constraint	$100\left(\frac{\text{Kg}}{\text{hr}^2\text{cell lengths}^2}\right)$	Osborne et al. [2010]
h	Cell height	$1\left(\text{cell lengths}\right)$	Walter [2009]
$\gamma_{j,k}$	Cell-cell adhesion constant	$\mathbf{J}\left(\frac{\text{Kg}}{\text{hr}^2\text{cell lengths}}\right)$	Wong et al. [2010]

Table D.5: Crypt cell-vertex model parameters

D.1.2.1 Crypt simulations

We then tested differential adhesion on a crypt layout with three cell types: stem, transit amplifying and differentiated, and track the lineage of one labelled stem cell. We wish to test different adhesion coefficients and see how the cells and their movement up the crypt change.

By making blue cells more adherent to each other we expected to see a continuous ribbon of cells migrating upwards, which is what we observe in Figs. D.21, but we did not see a large difference in simulations where the preferential adhesion occurred between cells of different types, like the checkerboard pattern seen in the monolayers of Fig. D.18.

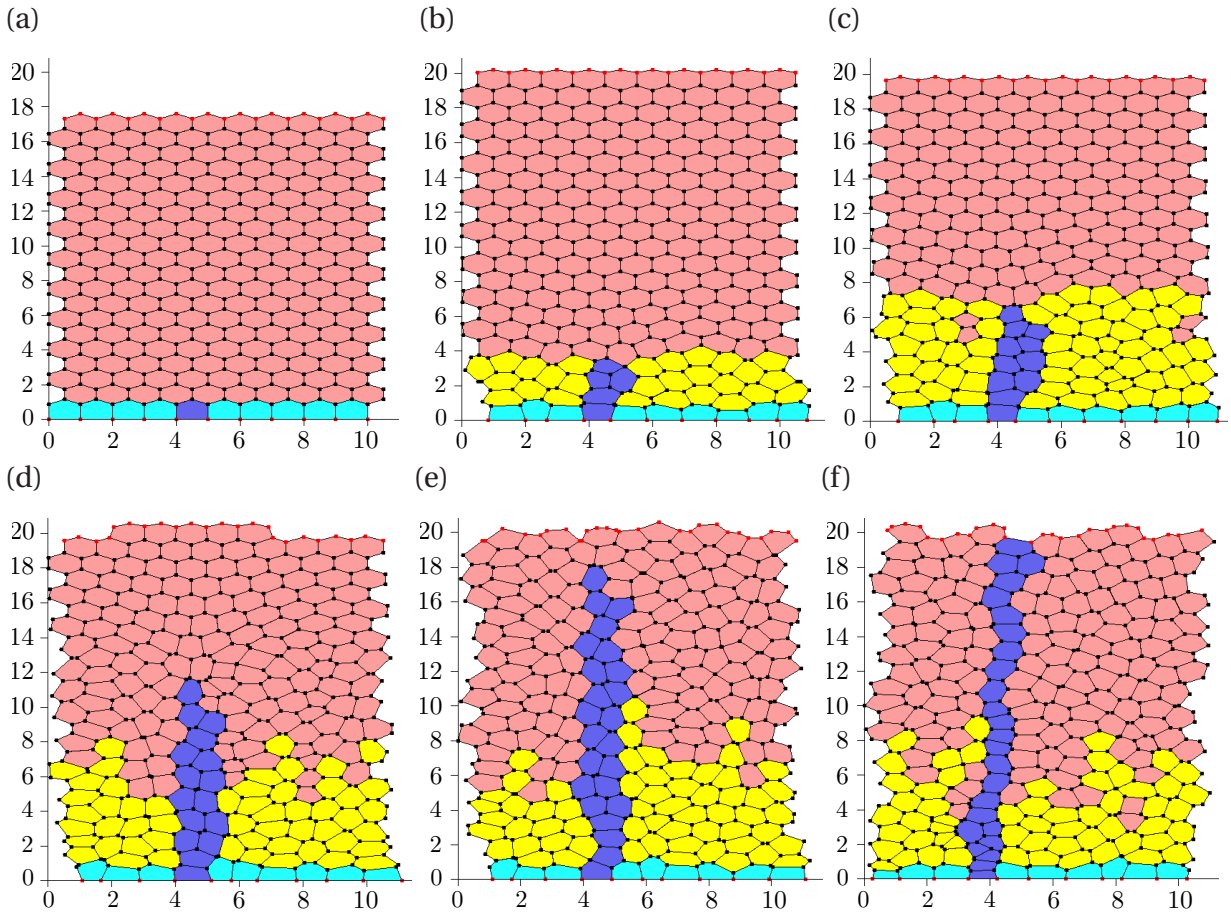


Figure D.21: Crypt simulation with preferential adhesion between pairs of cells from the same population. A stem cell is labelled dark blue at the beginning of the simulation and all its progeny are also represented by this colour. The rest of the cells are coloured cyan–stem, yellow–transit, pink–differentiated. The adhesion parameters are $\gamma_{\text{blue,blue}} = \gamma_{\text{wild,wild}} = 0.01$, $\gamma_{\text{blue,wild}} = 1.00$ where 'wild' denotes 'normal' cells. Note the cell progeny from the blue cell are denoted blue, regardless of whether or not they are transit or differentiated cells. Times (a) t=0.0, (b) t=23, (c) t=35, (d) t=44, (e) t=56, (f) t=112.7

D.1.2.2 Modelling migration in the crypt

We then modelled migration in the crypt using differential adhesion and the parameters used in the Potts model of Wong et al. [2010]. We were able to obtain comparable results to those obtained experimentally and *in silico* [Tsubouchi, 1983; Wong et al., 2010].

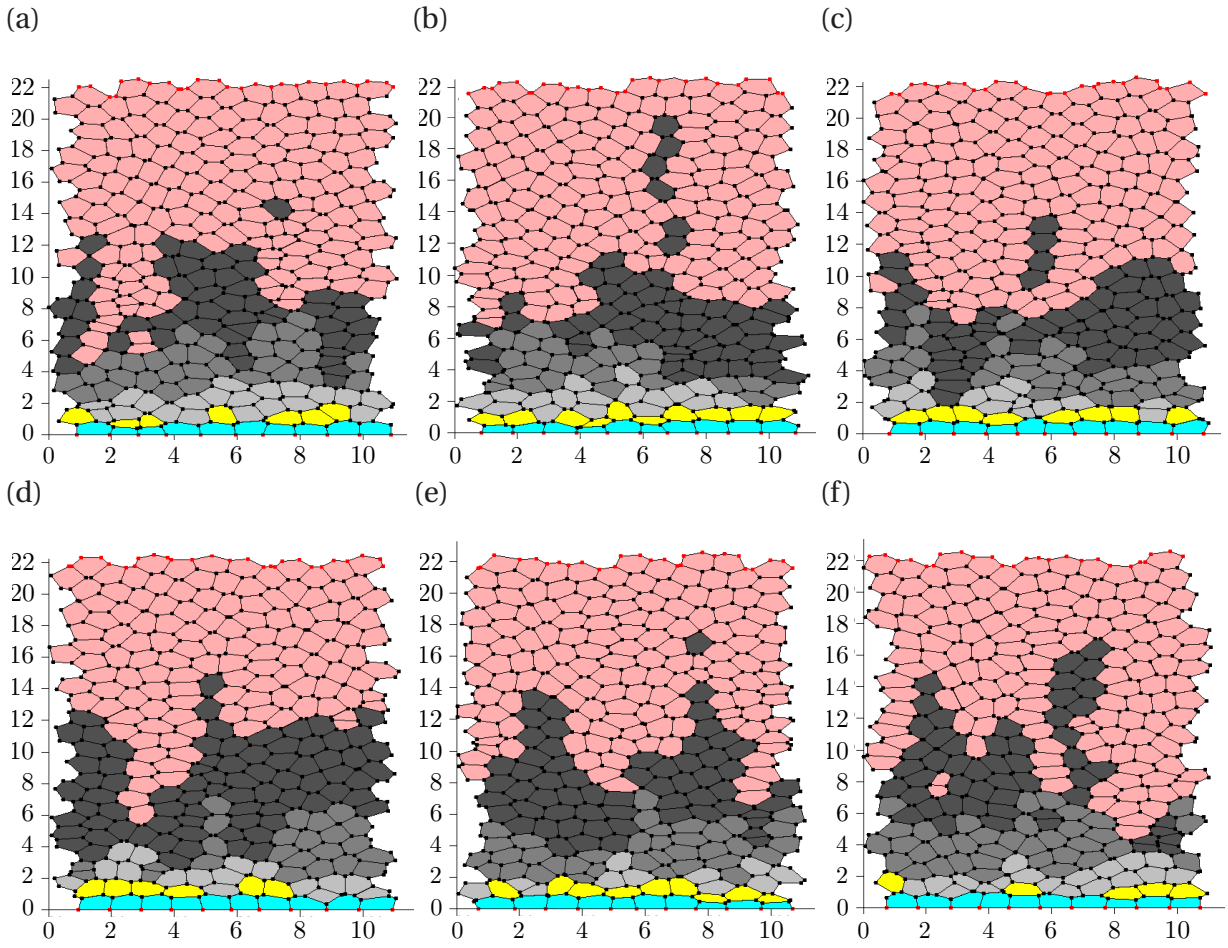


Figure D.22: Crypt cell-vertex model simulation at times (a) 0, (b) 20, (c) 40, (d) 60, (e) 80 and (f) 100 hrs. Colour code: pink - differentiated (D), dark grey - transit amplifying fourth generation (TA4), grey - transit amplifying third generation (TA3), light grey - transit amplifying second generation (TA2), yellow - transit amplifying first generation (TA1) and cyan - stem (S) cells.

D.1.3 Chaste

The Cancer, Heart and Soft Tissue Environment (Chaste) project is a general modelling framework written in C++, developed towards modelling multiscale heart and soft tissue and cancer [Pitt-Francis et al., 2008, 2009]. It is open-source, extensible, and testable, which makes it ideal to test the effects of embedding models of intracellular signalling pathways into a cell-based model [Pitt-Francis et al., 2008, 2009].

Chaste has been developed using agile techniques, methodologies which consist of incremental and iterative software development, such as pair-programming, test-driven development, code refactoring, continuous integration, amongst others, making the development of large and complex projects fast and more reliable [Abrahamsson et al., 2002].

A key feature of Chaste is that it is *test-driven*. By this we mean that before any piece of code is written, a test is created, making the programmers think beforehand what the functionalities of the new code will be and what tests will allow verification of the new code [Pitt-Francis et al., 2009].

Chaste is constantly refactored and updated through *iterations* where pairs of programmers work together on a particular task (using a single keyboard and screen – while one is programming, the other is checking the code as it is written and tested) [Pitt-Francis et al., 2008, 2009]. Specifically I worked on adding functionality, testing and refactoring a wide array of areas in the code, becoming familiar with different aspects of the environment, from the cell-vertex code to the spring and cellular automata code. For the implementation of the differential adhesion cell-vertex model I pair programmed with Prof. Angela Shiflet.

A common criticism of agile software development is that it is more developer-oriented rather than user-oriented, which in the case of Chaste I also consider to be the case so, unless the speed and modularity is critical, I would still recommend users to opt for their own code, given that it takes some effort to become familiar with and adapt Chaste to user requirements.

D.1.4 Equations of motion for vertices in the cell-vertex model

The movement of each cell will be given by the displacement, at each time step, of each of its vertices. Such displacement depends on the free energy of adjacent cells, which is comprised of a deformation energy, U_D, a membrane surface energy, U_S, and a cellular adhesion energy, U_{Adh} [Nagai and Honda, 2001; Walter, 2009].

$$\eta_i \frac{dr_i}{dt} = -\nabla_i U \tag{D.15}$$

where the total free energy, U, is the sum of three energy terms:

$$U = U_D + U_S + U_{Adh} = \sum_{j=1}^{M} \left(U_{j,D} + U_{j,S} + U_{j,Adh} \right), \tag{D.16}$$

where $U_{j,D}$, $U_{j,S}$ and $U_{j,Adh}$ represent the deformation energy, surface energy and cellular adhesion energy of cell j, respectively.

The deformation energy for a given cell is:

$$U_{j,D} = \kappa h^2 \left(A_j - A_{j,N} \right)^2, \tag{D.17}$$

where κ is an elastic constant or cell-size constraint, h is the height of the cell (we assume every cell has the same height), A_j and $A_{j,N}$ are, respectively, the area and target area of the j^{th} cell.

The surface energy of cell j is given by

$$U_{j,S} = \tau h^2 \left(C_j - C_{j,N} \right)^2, \tag{D.18}$$

where τ is a cell perimeter constraint and C_j and $C_{j,N}$ are the perimeter and the natural or target perimeter of the cell, respectively.

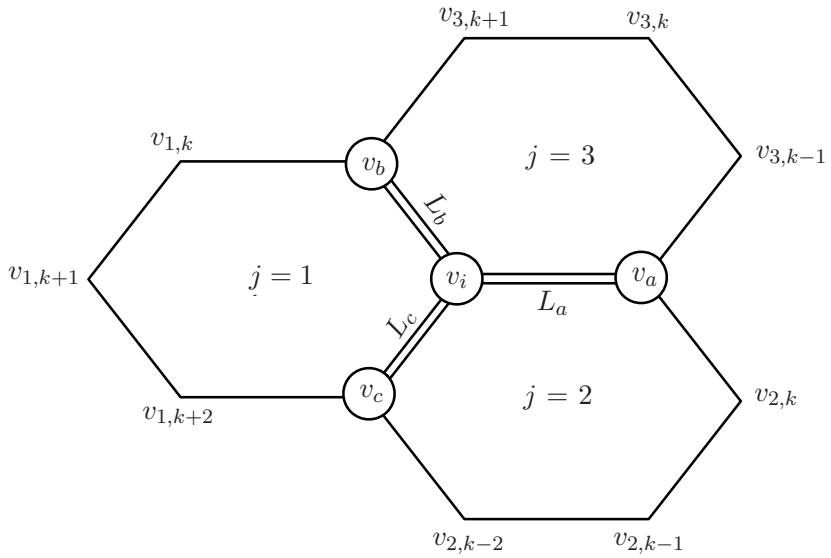


Figure D.23: Schematic representation of vertex v_i with three surrounding cells. In this example, $v_i = v_{1,k-2} = v_{2,k+2} = v_{3,k+3}$, $v_a = v_{2,k+1} = v_{3,k-2}$, $v_b = v_{1,k-1} = v_{3,k+2}$ and $v_c = v_{1,k+3} = v_{2,k-3}$.

The cellular adhesion of cell j is

$$U_{j,Adh} = \sum_{k=1}^{M_j} \gamma_{j,k} L_{j,k}, \tag{D.19}$$

where $\gamma_{j,k}$ is the adhesion constant between cell j and its neighbouring cell through edge $\overline{v_{j,k}\ v_{j,k+1}}$ and $L_{j,k}$ is the length of edge between vertices $v_{j,k}$ and $v_{j,k+1}$.

The movement of vertex v_i does not directly affect cells that are not surrounding it, the gradient of the free energy with respect to vertex i is only dependent on terms of cells labelled from 1 to 3, as in Fig. D.23 [Walter, 2009]:

$$\begin{aligned}
\nabla_i U &= \nabla_i \sum_{j=1}^{M} \left(U_{j,D} + U_{j,S} + U_{j,Adh} \right) = \nabla_i \sum_{j=1}^{3} \left(U_{j,D} + U_{j,S} + U_{j,Adh} \right) \qquad \text{(D.20)} \\
&= \sum_{j=1}^{3} \left(\nabla_i \left(\kappa h^2 \left(A_j - A_{j,N} \right)^2 \right) + \nabla_i \left(\tau h^2 \left(C_j - C_{j,N} \right)^2 \right) + \nabla_i \left(\sum_{k=1}^{M_j} \gamma_{j,k} L_{j,k} \right) \right) \\
&= \sum_{j=1}^{3} \left(2\kappa h^2 \left(A_j - A_{j,N} \right) \nabla_i A_j + 2\tau h^2 \left(C_j - C_{j,N} \right) \nabla_i C_j + \nabla_i \left(\sum_{k=1}^{M_j} \gamma_{j,k} L_{j,k} \right) \right).
\end{aligned}$$

Since the area of a polygon-shaped cell is given by [Nagai and Honda, 2001; Walter, 2009]

$$A_j = \frac{1}{2}\sum_{k=1}^{M_j}\left(x_{j,k}\cdot y_{j,k+1} - x_{j,k+1}\cdot y_{j,k}\right), \tag{D.21}$$

where $\left(x_{j,k}, y_{j,k}\right)$ are the coordinates of the vertices of the cell, the gradient of the area of the cell is given by

$$\nabla_k A_j = \begin{pmatrix} \frac{\partial A_j}{\partial x_k} \\ \frac{\partial A_j}{\partial y_k} \end{pmatrix} = \begin{pmatrix} y_{j,k+1} - y_{j,k-1} \\ x_{j,k-1} - x_{j,k+1} \end{pmatrix}. \tag{D.22}$$

The perimeter is the sum of the edge lengths $L_{j,k}$ of a cell, which according to the coordinates of its vertices is

$$C_j = \sum_{k=1}^{M_j} L_{j,k} = \sum_{k=1}^{M_j}\sqrt{\left(x_{j,k} - x_{j,k+1}\right)^2 + \left(y_{j,k} - y_{j,k+1}\right)^2}, \tag{D.23}$$

and its gradient,

$$\nabla_k C_j = \begin{pmatrix} \frac{\partial C_j}{\partial x_k} \\ \frac{\partial C_j}{\partial y_k} \end{pmatrix} = \begin{pmatrix} \frac{x_{j,k}-x_{j,k+1}}{L_{j,k}} + \frac{x_{j,k}-x_{j,k-1}}{L_{j,k}} \\ \frac{y_{j,k}-y_{j,k+1}}{L_{j,k}} + \frac{y_{j,k}-y_{j,k-1}}{L_{j,k}} \end{pmatrix}. \tag{D.24}$$

Finally, the gradient of the vertex lengths is

$$\nabla_k L_{j,k} = \begin{pmatrix} \frac{\partial L_{j,k}}{\partial x_k} \\ \frac{\partial L_{j,k}}{\partial y_k} \end{pmatrix} = \begin{pmatrix} \frac{x_{j,k}-x_{j,k+1}}{L_{j,k}} \\ \frac{y_{j,k}-y_{j,k+1}}{L_{j,k}} \end{pmatrix}. \tag{D.25}$$

$r_i = (x_i, y_i)$	Coordinates of vertex v_i
η_i	Drag over vertex v_i
∇_i	Gradient with respect to vertex i $\left(\frac{\partial}{\partial x_i}, \frac{\partial}{\partial y_i}\right)$
M	Total number of cells
M_j	Number of vertices of j^{th} cell, $j = 1, 2, \ldots, M$
$v_{j,k}$	k^{th} vertex of j^{th} cell, $j = 1, 2, \ldots, M$ and $k = 1, \ldots, M_j$, $v_{j,M_j+1} = v_{j,1}$
	Vertices are arranged in an anti-clockwise direction.
$x_{j,k}$	x coordinate for the k^{th} vertex of j^{th} cell
$y_{j,k}$	y coordinate for the k^{th} vertex of j^{th} cell
$L_{j,k}$	Length of edge between vertices $v_{j,k}$ and $v_{j,k+1}$
L_a	Length of edge between vertices v_i and v_a
L_b	Length of edge between vertices v_i and v_b
L_c	Length of edge between vertices v_i and v_c
h	Height of every cell
A_j	Area of j^{th} cell, $j = 1, 2, \ldots, M$
$A_{j,N}$	Target area of j^{th} cell, $j = 1, 2, \ldots, M$
C_j	Perimeter of j^{th} cell in 2-D, $j = 1, 2, \ldots, M$
$C_{j,N}$	Target perimeter of j^{th} cell in 2-D, $j = 1, 2, \ldots, M$ $\left(2\sqrt{\pi \cdot A_{j,N}}\right)$
U	Free energy of the system
U_D	Deformation energy of the system
U_S	Membrane surface energy of the system
U_{Adh}	Cell-cell adhesion energy of the system
$U_{j,D}$	Deformation energy of j^{th} cell, $j = 1, 2, \ldots, M$
$U_{j,S}$	Membrane surface energy of j^{th} cell, $j = 1, 2, \ldots, M$
$U_{j,Adh}$	Cell-cell adhesion energy of j^{th} cell, $j = 1, 2, \ldots, M$
$N_{j,k}$	Neighbouring cell of j^{th} cell through the edge $\overline{v_{j,k}\ v_{j,k+1}}$
α_j	Type of j^{th} cell
TS	Number of times steps
T_{α_i,α_j}	T1 swaps between i^{th} and j^{th} cells
$\gamma_{j,k}$	Adhesion constant, function of α_j and $\alpha_{N_{j,k}}$

Table D.6: Notation of equations of motion for vertices in the cell-vertex model.

References

P. Abrahamsson, O. Salo, J. Ronkainen, and J. Warsta. Agile software development methods. *Vtt Publications*, 478(3):167–168, 2002.

E. Acuña and C. Rodríguez. *The treatment of missing values and its effect in the classifier accuracy*. Springer, 2004.

S. Aeberhard, D. Coomans, and O. de Vel. Comparison of classifiers in high dimensional settings. Technical Report 92–02, Dept. of Computer Science and Dept. of Mathematics and Statistics, James Cook University of North Queensland, 1992.

A. Alon, N. Barkai, D. Notterman, K. Gish, S. Ybarra, D. Mack, and A. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proc Natl Acad Sci USA*, 96:6745–6750, 1999.

A.R.A. Anderson, M.A.J. Chaplain, K.A. Rejniak, and J.A. Fozard. Single-cell-based models in biology and medicine. *Math Med Biol*, 25(2):185–186, 2008.

N.M. Anstey and R.N. Price. Improving case definitions for severe malaria. *PLoS Med*, 4(8): e267, Aug 2007.

K. Arnki, T. Ogata, M. Kobayashi, and R. Yatani. A morphological study on the histogenesis of human colorectal hyperplastic polyps. *Eur J Gastroen Hepat*, 8(1), 1996.

A. Asunción and D. Newman. University of California, Irvine (UCI) machine learning repository, 2007. URL `http://archive.ics.uci.edu/ml/`.

F. Azuaje and J. Dopazo. *Data Analysis and Visualization in Genomics and Proteomics*. John Wiley & Sons, 2005.

K. Baggerly and K. Coombes. An introduction to high-throughput bioinformatics data. *Cancer*, (March):1–34, 2006.

B Bah. Diffusion maps: analysis and applications, 2008.

M. Barni, V. Cappellini, and A. Mecocci. Comments on a possibilistic approach to clustering. *IEEE Trans Fuzzy Syst*, 4(3):393–396, 1996.

E. Batlle, H. J.T. Henderson, Beghtel, M.M.W. van den Born, E. Sancho, G. Huls, J. Meeldijk, J. Robertson, M. van de Wetering, T. Pawson, and H. Clevers. Beta-catenin and TCF mediate cell positioning in the intestinal epithelium by controlling the expression of EphB/EphrinB. *Cell*, 111(2):251–263, 2002.

M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Comput*, 15:1373–1396, 2003.

A. Ben-Hur, D. Horn, H. Siegelmann, and V. Vapnik. Support vector clustering. *J Mach Learn Res*, 2:125–137, Mar 2002.

M. Berthold and D. Hand, editors. *Intelligent Data Analysis: An Introduction*. Springer-Verlag New York, Inc., 1st edition, 1999.

J.C. Bezdek, R. Ehrlich, and W. Full. FCM: the Fuzzy C-means clustering algorithm. *Comput Geosc*, 10(2-3):191–203, 1983.

G.W. Brodland. Computational modeling of cell sorting, tissue engulfment, and related phenomena: A review. *Appl Mech Rev*, 57(1):47–76, 2004.

P.O. Brown and D. Botstein. Exploring the new world of the genome with DNA microarrays. *Nature Genetics*, 21(1 Suppl):33–7, 1999.

R. Butterworth. On feature selection through clustering. *Proc 5th IEEE Int Conf Data Mining*, 2005.

R. Campello and E. Hruschka. A fuzzy extension of the silhouette width criterion for cluster analysis. *Fuzzy Sets Syst*, 157(21):2858–2875, 2006.

L.S. Chan and O.J. Dunn. The treatment of missing values in discriminant analysis – I. The sampling experiment. *J Amer Stat Assoc*, 67(338):473–477, 1972.

S.C. Chi and C.C. Yang. A two-stage clustering method combining ant colony SOM and K-means. *J Inf Sci Eng*, 24(5):1445–1460, 2008.

Y.T. Chien. *Interactive Pattern Recognition*. Marcel Dekker, Inc., 1978.

J.G. Cleland. Controversies in the management of heart failure. *Edimburgh. Churchill Livingstone*, pages vii–viii, 1997.

H. Clevers and E. Batlle. EphB/EphrinB receptors and Wnt signaling in colorectal cancer. *Cancer Res*, 66(1):2–5, Jan 2006.

R. Coifman and S. Lafon. Diffusion maps. *Appl Comp Harm Anal*, (21):5–30, 2006.

O. Cominetti, A. Matzavinos, S. Samarasinghe, D. Kulasiri, S. Liu, P.K. Maini, and R. Erban. DifFUZZY: A fuzzy clustering algorithm for complex data sets. *Int J Comput Intell Bioinf Syst Bio*, 1(4):402–417, 2010.

C. Cortina, S. Palomo-Ponce, M. Iglesias, J.L. Fernandez-Masip, A. Vivancos, G. Whissell, M. Huma, N. Peiro, L. Gallego, S. Jonkheer, A. Davy, J. Lloreta, E. Sancho, and E. Batlle. EphB-ephrin-B interactions suppress colorectal cancer progression by compartmentalizing tumor cells. *Nat Genet*, 39(11):1376–1383, Nov 2007.

M. Dash and H. Liu. Feature selection for clustering. In *Proc 4th Pacific-Asia Conf Know Discovery and Data Mining*, pages 110–121. Springer-Verlag, 2000.

A.G. De Brevern, S. Hazout, and A. Malpertuy. Influence of microarrays experiments missing values on the stability of gene groups by hierarchical clustering. *BMC Bioinformatics*, 5:114, 2004.

W. de Nooy, A. Mrvar, and V. Batagelj. *Exploratory Social Network Analysis with Pajek (Structural Analysis in the Social Sciences)*. Cambridge University Press, 2005.

D. Dembélé and P. Kastner. Fuzzy C-means method for clustering microarray data. *Bioinformatics*, 19(8):973–980, May 2003.

A. Denton. Kernel-density-based clustering of time series subsequences using a continuous random-walk noise model. In *5th IEEE Int Conf Data Mining*, page 8, Nov 2005.

A. Deutsch and S. Dormann. *Cellular Automaton Modeling of Biological Pattern Formation: Characterization, Applications, and Analysis*. Birkhauser, 2004.

G. Di Battista, P. Eades, R. Tamassia, and I. Tollis. Algorithms for drawing graphs: an annotated bibliography. *Comp Geom-Theor Appl*, 5(4), 1994.

G. Di Battista, P. Eades, R. Tamassia, and I. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice-Hall, 1999.

S. Ding. A tutorial on spectral clustering II: Advanced/related topics, 2004.

W. Donath and A. Hoffman. Lower bounds for the partitioning of graphs. *IBM J Res Dev*, 17 (5):420–425, 1973.

J. Dunn. A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters. *Cybern Syst*, 3:32–57, 1973.

R. Erban, J. Chapman, and P. Maini. A practical guide to stochastic simulations of reaction-diffusion processes, 2007a. URL `http://arxiv.org/abs/0704.1908`.

R. Erban, T. Frewen, X. Wang, T. Elston, R. Coifman, B. Nadler, and I. Kevrekidis. Variable-free exploration of stochastic models: a gene regulatory network example. *J Chem Phys*, 126(15):155103, Apr 2007b.

J. Ernst, G.J. Nau, and Z. Bar-Joseph. Clustering short time series gene expression data. *Bioinformatics*, 21(suppl 1):i159–i168, 2005.

B.S. Everitt, S. Landau, M. Leese, and D. Stahl. Cluster analysis. *Wiley Ser Probab Statist*, 2011.

T. Fawcett. Roc graphs: Notes and practical considerations for researchers. Technical Report HPL-2003-4, HP Labs, 2004.

T. Fawcett. Roc analysis in pattern recognition. *Pattern Recogn Lett*, 27(8):861–874, June 2006.

M. Filippone, F. Camastra, F. Masulli, and S. Rovetta. A survey of kernel and spectral methods for clustering. *Pattern Recogn*, 41(1):176–190, 2008.

R. Fisher. The use of multiple measurements in taxonomic problems. *Annals Eugen*, 7: 179–188, 1936.

S. French, M. Rosenberg, and M. Knuiman. The clustering of health risk behaviours in a Western Australian adult population. *Health Promot J Austr*, 19(3):203–209, Dec 2008.

K. Fukuda and T. Matsui. Finding all the perfect matchings in bipartite graphs. *Appl Math Lett*, 7:15–18, 1989.

G. Gan, C. Ma, and J. Wu. Data clustering: Theory, algorithms, and applications. *ASA-SIAM Ser Stat Appl Probab*, 20, 2007.

N. Gehlenborg, S. Donoghue, N. Baliga, A. Goesmann, M. Hibbs, H. Kitano, O. Kohlbacher, H. Neuweger, R. Schneider, and D. Tenenbaum. Visualization of omics data for systems biology. *Methods*, 7(3):S56–68, 2010.

T. Golub, D. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. Mesirov, H. Coller, M. Loh, J. Downing, M. Caligiuri, C. Bloomfield, and E. Lander. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, 286(5439): 531–537, Oct 1999.

B. Good, Y. de Montjoye, and A. Clauset. The performance of modularity maximization in practical contexts. *Phys Rev E*, 81:046106, 2010.

N.N. Gopal and M Karnan. Diagnose brain tumor through MRI using image processing clustering algorithms such as Fuzzy C Means along with intelligent optimization techniques. *ICCIC 2010 IEEE Int Conf Comp Intell Comp Res*, pages 1–4, 2010.

S. Guha, R. Rastogi, and K. Shim. Cure: an efficient clustering algorithm for large databases. *Inf Syst*, 26(1):35–58, 2001.

I. Guyon, R. Williamson, and U. van Luxburg. Clustering: Science or Art? *NIPS Workshop on Clustering*, pages 1–11, 2009.

L. Hammouche, M. Diaf, and J.G. Postaire. A clustering method based on multidimensional texture analysis. *Pattern Recogn*, 39(7):1265–1277, 2006.

J. Handl, J. Knowles, and D.B. Kell. Computational cluster validation in post-genomic data analysis. *Bioinformatics*, 21(15):3201–3212, Aug 2005.

D. Harel and Y. Koren. On clustering using random walks. *FST TCS 2001: Foundations of Software Technology and Theoretical Computer Science*, 2245:18–41, Nov 2001.

R. Hathaway and J.C. Bezdek. Recent convergence results for the Fuzzy C-means clustering algorithms. *J Classif*, 5(2):237–247, 1988.

Z. He, X. Xu, and S. Deng. Clustering mixed numeric and categorical data: A cluster ensemble approach. *CoRR*, 2005.

T. Hocking, A. Joulin, F. Bach, and J. Vert. Clusterpath: An algorithm for clustering using convex fusion penalties. *Proc 28th Int Conf Mach Learn*, pages 745–752, Jun 2011.

J. Horvath. Image segmentation using Fuzzy C-means. *Proc 6th IEEE Int Symp Appl Mach Intell Inf*, 2006.

M.E. Houle. The relevant-set correlation model for data clustering. *Stat Anal Data Min*, 1 (3):157–176, Nov 2008.

D. Howe, M. Costanzo, P. Fey, T. Gojobori, L. Hannick, W. Hide, D. Hill, R. Kania, M. Schaeffer, S. St Pierre, S. Twigger, O. White, and S. Yon Rhee. Big data: The future of biocuration. *Nature*, 455(7209):47–50, Sept 2008.

M. Huang and P. Eades. A fully animated interactive system for clustering and navigating huge graphs. In S. Whitesides, editor, *Graph Drawing*, volume 1547 of *Lecture Notes in Computer Science*, pages 374–383. Springer Berlin/Heidelberg, 1998.

L. Hubert and P. Arabie. Comparing partitions. *J Classif*, (2):193–218, 1985.

A.K. Jain. Data clustering: 50 years beyond K-means. *Lect Notes Artif Int*, 31(8):3–29, 2008.

L.C. Jain, M. Sato-Ilic, M. Virvou, G.A. Tsihrintzis, and V.E. Balas. *Computational Intelligence Paradigms: Innovative Applications*, volume 137 of *SCI*. Springer-Verlag, 2008.

C.S. Jensen, D. Lin, and B.C. Ooi. Continuous clustering of moving objects. *IEEE Trans Knowl Data Eng*, 19(9):1161–1174, Sept. 2007.

D. Jiang, J. Pei, and A. Zhang. DHC: A density-based hierarchical clustering method for time series gene expression data, 2003.

S. Johnson. Hierarchical clustering schemes. *Psychometrika*, 32(3):241–254, Sep 1967.

I.T. Jolliffe. *Principal Component Analysis*. Springer, Oct 2002.

J. Kao, K. Salari, M. Bocanegra, Y. Choi, L. Girard, J. Gandhi, K.A. Kwei, T. Hernandez-Boussard, A.F. Wang, P.and Gazdar, J.D. Minna, and J.R. Pollack. Molecular profiling of breast cancer cell lines defines relevant tumor models and provides a resource for cancer gene discovery. *PLoS ONE*, 4(7):e6146, Jul 2009.

N. Karthikeyani and K. Thangavel. Impact of normalization in distributed K-means clusterings. *Int Journal of Soft Computing*, 4:168–172, 2009.

J.C. Kasemeier-Kulesa, R. Bradley, E.B. Pasquale, F. Lefcort, and P.M. Kulesa. Eph/ephrins and N-cadherin coordinate to control the pattern of sympathetic ganglia. *Development*, 133(24):4839–4847, 2006.

D. Kauchak and S. Dasgupta. An iterative improvement procedure for hierarchical clustering. *Adv Neural Inf Process Syst*, 16:481–488, 2004.

L. Kaufman and P. Rousseeuw. *Finding groups in data. An introduction to cluster analysis*. Wiley Series in Probability and Mathematical Statistics. Applied Probability and Statistics, 1990.

M.Y. Kiang, U.R. Kulkarni, M. Goul, R.T. Chi, and E. Turban. Improving the effectiveness of self-organizing map networks using a circular Kohonen layer. In *Int Conf Syst Sci*, pages 521–529, 1997.

S. Kim, J. Lee, and J. Bae. Effect of data normalization on fuzzy clustering of DNA microarray data. *BMC Bioinformatics*, 7:134, 2006.

S.Y. Kim, T.M. Choi, and J.S. Bae. Fuzzy types clustering for microarray data. *Int J Comput Intell*, pages 12–15, 2005.

J. Kleinberg. An impossibility theorem for clustering. *Adv Neural Inf Process Syst*, (15): 446–453, 2002.

S.G. Kobourov. Spring embedders and force directed graph drawing algorithms. *CoRR*, 2012.

I. Kohane, A. Kho, and A. Butte. *Microarrays for an Integrative Genomics*. MIT Press, 2003.

T. Kosmerl and D. Bavcar. Determination of ash content in Slovenian wines by empirical equations, Sept 2003.

R. Krishnapuram and J. Keller. A possibilistic approach to clustering. *IEEE Trans Fuzzy Syst*, 1(2):98–110, May 1993.

H.W. Kuhn. The Hungarian method for the assignment problem. *Nav Res Log*, 2:83–97, 1955.

R. Lambiotte. Multi-scale modularity in complex networks. *Methods*, pages 546–553, 2010.

R. Lambiotte, J.C. Delvenne, and M. Barahona. Laplacian dynamics and multiscale modular structure in networks. *Arxiv preprint arXiv08121770*, 812:1–29, 2008.

J.L. Ledford, J. Teixeira, and M.E. Tyler. *Google Analytics*. John Wiley & Sons, 2010.

J. Leskovec. Dimensionality reduction. PCA, SVD, MDS, ICA, and friends. *Machine Learning recitation*, Apr 2006.

J. Leskovec, K.J. Lang, and M. Mahoney. Empirical comparison of algorithms for network community detection. *Proc 19th Int Conf WWW*, pages 631–640, 2010.

H. Li. Lagrange multipliers and their applications. *Electr Eng*, (10), 2008.

Y.Q. Li, S.A. Roberts, U. Paulus, M. Loeffler, and C.S. Potten. The crypt cycle in mouse small intestinal epithelium. *J Cell Sci*, (107):3271–3279, 1994.

W. Liang, A. Maddukuri, T. Teslovich, C. De la Fuente, E. Agbottah, S. Dadgar, K. Kehn, S. Hautaniemi, A. Pumfery, D. Stephan, and F. Kashanchi. Therapeutic targets for HIV-1 infection in the host proteome. *Retrovirology*, 2(1):20, 2005.

T.W. Liao. Clustering of time series data - a survey. *Pattern Recogn*, 38:1857–1874, 2005.

U. Luxburg. A tutorial on spectral clustering. *Stat Comput*, 17(4):395–416, 2007.

J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proc 4th Berkeley Symp Math Stat Prob*, volume 1, pages 281–297, 1967.

K. Maitland, S. Nadel, A.J. Pollard, T.N. Williams, C.R. Newton, and M. Levin. Management of severe malaria in children: proposed guidelines for the United Kingdom. *BMJ*, 331 (7512):337–343, 8 2005.

J.I. Maletic and A. Marcus. Data cleansing: Beyond integrity checking. In *Proc 2000 ICIQ Conf*, pages 200–209, Boston, MA, USA, 2000.

C.D. Manning, P. Raghavan, and H. Schutze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

K. Marsh, D. Forster, C. Waruiru, I. Mwangi, M. Winstanley, V. Marsh, C. Newton, P. Winstanley, P. Warn, N. Peshu, G. Pasvol, and R. Snow. Indicators of life-threatening malaria in African children. *N Engl J Med*, 332(21):1399–1404, 1995.

A. Martínez and W. Martínez. *Exploratory Data Analysis with MATLAB. Chapter 5 – Finding Clusters*. Routledge, 2004.

M. Matteucci. A tutorial on clustering algorithms. `http://home.dei.polimi.it/matteucc/Clustering/tutorial_html/index.html`, 2004.

F.A. Meineke, C.S. Potten, and M. Loeffler. Cell migration and organization in the intestinal crypt using a lattice-free model. *Cell Prolif*, 34(4):253–266, 2001.

G. Menschaert, T. Vandekerckhove, B. Landuyt, E. Hayakawa, L. Schoofs, W. Luyten, and W. Van Criekinge. Spectral clustering in peptidomics studies helps to unravel modification profile of biologically active peptides and enhances peptide identification rate. *Proteomics*, 9(18):4381–4388, 2009.

A. Mohan, S.K. Sharma, and S. Bollineni. Acute lung injury and acute respiratory distress syndrome in malaria. *J Vector Borne Dis*, 45(3):179–193, 2008.

L. Morey and A. Agresti. The measurement of classification agreement: an adjustment to the rand statistic for chance agreement. *Educ Psychol Meas*, (44):33–37, 1984.

D.J Mundfrom and A. Whitcomb. Imputing missing values: The effect on the accuracy of classification. *MLRV*, 25(1):13–19, 1998.

J. Murray. *Mathematical biology: I. An introduction*. Springer, 2005.

B. Nadler and M. Galun. Fundamental limitations of spectral clustering. In *Adv Neural Inf Process Syst 19*, pages 1017–1024, 2007.

T. Nagai and H. Honda. A dynamic cell model for the formation of epithelial tissues. *Phil Mag B*, 81(7):699–719, 2001.

A. Newman and J. Cooper. AutoSOME: a clustering method for identifying gene expression modules without prior knowledge of cluster number. *BMC Bioinformatics*, pages 11–117, 2010.

M.E.J. Newman. Modularity and community structure in networks. *Proc Natl Acad Sci USA*, 103(23):8577–8582, Jun 2006.

M.E.J. Newman. *Networks: An Introduction*. Oxford University Press, 2010.

A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Adv Neural Inf Proces Syst 14*, pages 849–856. MIT Press, 2002.

P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Comput*, 15:1373–1396, 2003.

M.A. Nowak, N.L. Komarova, A. Sengupta, P.V. Jallepalli, I. Shih, B. Vogelstein, and C. Lengauer. The role of chromosomal instability in tumor initiation. *Proc Natl Acad Sci USA*, 99(25):16226–16231, 2002.

S. Oba, M. Sato, I. Takemasa, M. Monden, K. Matsubara, and S. Ishii. A Bayesian missing value estimation method for gene expression profile data. *Bioinformatics*, 19(16):2088–2096, 2003.

J. Osborne, A. Walter, S.K. Kershaw, G.R. Mirams, A.G. Fletcher, P. Pathmanathan, D.J. Gavaghan, O.E. Jensen, P.K. Maini, and H.M. Byrne. A hybrid approach to multi-scale modelling of cancer. *Phil Trans R Soc A Math Phys Eng Sci*, 368(1930):5013–5028, 2010.

N. Pal and J. Bezdek. Corrections to "On cluster validity for the Fuzzy C-means model". *IEEE Trans Fuzzy Syst*, 5(1):152–153, Feb 1997.

N. Pal, K. Pal, and J Bezdek. A mixed c-means clustering model. *Proc 6th IEEE Int Conf Fuzzy Syst*, 1:11–21, Jul 1997.

N. Pal, J. Keller, M. Popescu, J. Bezdek, Mitchell, and J. Huband. Gene ontology-based knowledge discovery through fuzzy cluster analysis. *Neur Parall Scient Comp*, 13:337–362, 2005.

G. Palla, I. Derenyi, I. Farkas, and T. Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435(7043):814–818, Jun 2005.

J. Pitt-Francis, M.O. Bernabeu, J. Cooper, A. Garny, L. Momtahan, J. Osborne, P. Pathmanathan, B. Rodriguez, J.P. Whiteley, and D.J. Gavaghan. Chaste: using agile programming techniques to develop computational biology software. *Philos Transact A Math Phys Eng Sci*, 366(1878):3111–3136, Sep 2008.

J. Pitt-Francis, P. Pathmanathan, M.O. Bernabeu, R. Bordas, J. Cooper, A.G. Fletcher, G.R. Mirams, P. Murray, J.M. Osborne, A. Walter, S.J. Chapman, A. Garny, I.M.M. van Leeuwen, P.K. Maini, B. Rodriguez, S.L. Waters, J.P. Whiteley, H.M. Byrne, and D.J. Gavaghan. Chaste: A test-driven approach to software development for biological modelling. *Comput Phys Commun*, 180(12):2452–2471, 2009.

A. Poliakov, M. Cortina, and D.G. Wilkinson. Diverse roles of Eph receptors and ephrins in the regulation of cell migration and tissue assembly. *Dev Cell*, 7(4):465–480, Oct 2004.

J.M. Pujol, J. Béjar, and J. Delgado. Clustering algorithm for determining community structure in large networks. *Phys Rev E Stat Nonlin Soft Matter Phys*, 74(1 Pt 2):016107, 2006.

J. Quackenbush. Computational analysis of microarray data. *Nat Rev Genet*, 2:418, 2001.

W.M. Rand. Objective criteria for the evaluation of clustering methods. *J Amer Stat Assoc*, 66(336):846–850, 1971.

P.J. Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *J Comput Appl Math*, 20:53–65, 1987.

E. Ruspini. Numerical methods for fuzzy clustering. *Inform Sci*, 2(3):319–350, 1970.

M.F. Saad and A.M. Alimi. Modified fuzzy possibilistic C-means. *Proc Int MultiConf Engi Comp*, 1(4), 2009.

S. Samarasinghe. *Neural Networks for Applied Sciences and Engineering: From Fundamentals to Complex Pattern Recognition*. Auerbach Publications, 2006.

A.X. Sarkar and E.A. Sobie. Regression analysis for constraining free parameters in electrophysiological models of cardiac cells. *PLoS Comput Biol*, 6(9):e1000914, Sept 2010.

S.E. Schaeffer. Graph clustering. *Computer Science Review*, 1(1):27–64, 2007.

A. Shah, M. Singhal, K. Klicker, E. Stephan, H. Wiley, and K. Waters. Systems biology, enabling high-throughput data management for systems biology: The Bioinformatics Resource Manager. *Bioinformatics*, 23(7):906–909, 2007.

Z.S. Shokri and K. Alsultan. A simulated annealing algorithm for the clustering problem. *Pattern Recognition*, 24(10):1003–1008, 1991.

G. Simon, J. Lee, and M. Verleysen. Unfolding preprocessing for meaningful time series clustering. *Neural Networks*, 19(6-7):877–888, Jul 2006.

R.M. Simon. *Design and analysis of DNA microarray investigations*. Springer, SBH, 2003.

A. Singhal and D.E. Seborg. Clustering multivariate time-series data. *Journal of Chemometrics*, 19(8):427–438, 2005.

D. Smith. Network visualization and energy optimization. *In preparation*, 2012.

M.S. Steinberg. Differential adhesion in morphogenesis: A modern view. *Curr Opin Genet Dev*, 17(4):281–286, 2007.

N. Street, W.H. Wolberg, and O.L. Mangasarian. Nuclear feature extraction for breast tumor diagnosis. *IS&T/SPIE I S Elect Imaging Sci Technol*, pages 861–870, 1993.

J. Stuart, E. Segal, D. Koller, and S. Kim. A gene-coexpression network for global discovery of conserved genetic modules. *Science*, 302(5643):249–255, Oct 2003.

M. Su, T. Liu, and H. Chang. Improving the self organizing feature map algorithm using an efficient initialization scheme. *Tamkang J Sci Eng*, 5:35–48, 2002.

M. Swat, A. Kel, and H. Herzel. Bifurcation analysis of the regulatory modules of the mammalian G1/S transition. *Bioinformatics*, 20(10):1506–1511, 2004.

P.N. Tan, M. Steinbach, and V. Kumar. *Introduction to data mining*. Pearson Addison Wesley, 2006.

Y. Tan, L. Shi, W. Tong, G.T. Hwang, and C. Wang. Multi-class tumor classification by discriminant partial least squares using microarray gene expression data and assessment of classification models. *Comput Biol Chem*, 28(3):235–244, Jul 2004.

T. Taylor, C. Olola, C. Valim, T. Agbenyega, P. Kremsner, S. Krishna, D. Kwiatkowski, C. Newton, M. Missinou, M. Pinder, and D. Wypij. Standardized data collection for multi-center clinical studies of severe malaria in African children: establishing the SMAC network. *Trans R Soc Trop Med Hyg*, 100(7):615–622, Aug 2006.

B. Thomas, G. Raju, and W. Sonam. A modified fuzzy c-means algorithm for natural data exploration. *WASET*, 49:478–481, 2009.

A. Topchy, A.K. Jain, and W. Punch. Combining multiple weak clusterings. *IEEE Int Conf Data Mining*, pages 331–338, Nov 2003.

M. Trivedi and J. Bezdek. Low-level segmentation of aerial images with fuzzy clustering. *IEEE Trans Syst Man Cybern*, SMC-16:589–598, 1986.

M. Trujillo, I. Shakra, and A. El Saddik. Haptic: the new biometrics-embedded media to recognizing and quantifying human patterns. *Proc 13th ACM Int Conf Multim*, pages 387–390, 2005.

S. Tsubouchi. Theoretical implications for cell migration through the crypt and the villus of labelling studies conducted at each position within the crypt. *Cell Tissue Kinet*, 16: 441–456, 1983.

J. Tuikkala, L.L. Elo, O.S. Nevalainen, and T. Aittokallio. Missing value imputation improves clustering and interpretation of gene expression microarray data. *BMC Bioinformatics*, 9 (202):202, 2008.

J.J. Tyson and B. Novak. Regulation of the eukaryotic cell cycle: Molecular antagonism, hysteresis, and irreversible transitions. *J Theor Biol*, 210(2):249–263, 2001.

T.C. Van Der Pouw Kraan, F.A. Van Gaalen, T.W.J. Huizinga, E. Pieterman, F.C. Breedveld, and C.L. Verweij. Discovery of distinctive gene expression profiles in rheumatoid synovium using cDNA microarray technology: evidence for the existence of multiple pathways of tissue destruction and repair. *Genes Immun*, 4(3):187–196, 2003.

M. van Hulle and J. Davis. Data mining course notes. *Laboratorium voor Neuro en Psychofysiologie, K.U.Leuve*, 1998.

I. van Leeuwen, H. Byrne, O. Jensen, and J. King. Elucidating the interactions between the adhesive and transcriptional functions of β-catenin in normal and cancerous cells. *J Theor Biol*, 247(1):77–102, Jul 2007.

R. Varshavsky, D. Horn, and M. Linial. Global considerations in hierarchical clustering reveal meaningful patterns in data. *PLoS ONE*, 3(5):e2247, May 2008.

K.L. Wagstaff and V.G. Laidler. Making the most of missing values: Object clustering with partial data in astronomy. *XIV Proc Astron Data Anal Soft Syst*, 347:172–176, 2005.

A. Walter. *A comparison of continuum and cell-based models of colorectal cancer*. PhD thesis, University of Nottingham, 2009.

H. Wang, Z. Wang, X. Li, B. Gong, L. Feng, and Y. Zhou. A robust approach based on Weibull distribution for clustering gene expression data. *Algorithm Mol Biol*, 6(1), 2011.

J.Z. Wang. *Integrated Region-Based Image Retrieval*. The Kluwer Int Series Inf Retrieval. Kluwer Academic Publishers, 2001.

X. Wang, M. Wu, Z. Li, and C. Chan. Short time-series microarray analysis: Methods and challenges. *BMC Syst Biol*, 2(1):58, 2008.

R. Wehrens. Chemometrics I. Study Guide, Oct 2008.

W. Wei-Yi, L. Zhan-Ming, and Z. Guo-Quan. Novel fuzzy clustering-based image segmentation with simultaneous uneven illumination estimation. *J Inform Technol*, 10:607–610, Oct 2010.

WHO. Severe falciparum malaria. *Trans R Soc Trop Med Hyg*, (94):Suppl S1–90, Apr 2000.

WHO. World malaria report. *World Health Organisation*, 2011.

D.G. Wilkinson. How attraction turns to repulsion. *Nat Cell Biol*, 5(10):851–853, Oct 2003.

H.G. Wilson, B. Boots, and A.A. Millward. A comparison of hierarchical and partitional clustering techniques for multispectral image classification. *In Proc IEEE Int Geosc Rem Sens Symp*, 3:1624–1626, Jun 2002.

D.S. Wishart, R. Yang, D. Arndt, P. Tang, and J. Cruz. Dynamic cellular automata: an alternative approach to cellular simulation. *In Sil Biol*, 5(2):139–161, 2005.

T. Wittman. Time-series clustering and association analysis of financial data, Dec 2002.

S.Y. Wong, K.H. Chiam, C.T. Lim, and P. Matsudaira. Computational model of cell positioning: directed and collective migration in the intestinal crypt epithelium. *J R Soc Interface*, 7(Suppl 3):S351, 2010.

C. Wright, T. Burns, P. James, J. Billings, S. Johnson, M. Muijen, S. Priebe, I. Ryrie, J. Watts, and I. White. Assertive outreach teams in London: models of operation. *Br J Psychiatry*, 183(2):132–138, 2003.

B. Yao and S. Li. ANMM4CBR: a case-based reasoning method for gene expression data classification. *Algorithms Mol Biol*, 5(14):14, 2010.

H. Zare, P. Shooshtari, A. Gupta, and R. Brinkman. Data reduction for spectral clustering to analyze high throughput flow cytometry data. *BMC Bioinformatics*, 11(1):403, 2010.

C. Zhang, K. Chou, and G. Maggiora. Predicting protein structural classes from amino acid composition: application of fuzzy clustering. *Protein Eng*, 8:425–435, 1995.

C. Zhang, X. Zhang, M.Q. Zhang, and Y. Li. Neighbor number, valley seeking and clustering. *Pattern Recogn Lett*, 28(2):173–180, 2007.

S. Zhang, J. Zhang, X. Zhu, Y. Qin, and C. Zhang. Missing value imputation based on data clustering. *Trans Comput Sci J*, 1:128–138, 2008.

X. Zhang and L. Jiang. An image segmentation algorithm based on Fuzzy C-means clustering. In *Digital Image Processing, 2009 Int Conf on*, pages 22–26, March 2009.

H. Zhao, A. Langerod, Y. Ji, K. Nowels, J. Nesland, R. Tibshirani, I. Bukholm, R. Karesen, D. Botstein, A. Borresen-Dale, and S. Jeffrey. Different gene expression patterns in invasive lobular and ductal carcinomas of the breast. *Mol Biol Cell*, 15(6):2523–2536, Jun 2004.

Y. Zhao, E. Levina, and J. Zhu. On consistency of community detection in networks. *Ann Stat*, Nov 2011.